الجمهورية العربية السورية
المعهد العالي للعلوم التطبيقية والتكنولوجيا
قسم المعلوماتية
العام الدراسي 2024/2025

مشروع سنة رابعة

بناء نظام كشف هجومات باستخدام خوارزميات التَّعلُّم الآلي والتَّعلُّم العميق

تقديم
حسين سلوم

إشراف
د. سميح جمّول
م. محمد بشار دسوقي

202/8/10

الخلاصة

مع الانتشار المتسارع لشبكات إنترنت الأشياء (IoT) وما يصاحبها من تحديات أمنية، أصبحت هذه الأجهزة هدفاً رئيسياً لتشكيل الروبوتات الشبكية (Botnets) التي تُستخدم لشن هجمات واسعة النطاق. يعالج هذا المشروع هذه المشكلة من خلال تصميم وتنفيذ وتقييم نظام ذكي لكشف التسلل (IDS) يعتمد على تعلم الآلة والتعلم العميق، بهدف تحديد حركة المرور الخبيثة في شبكات IoT في الزمن الفعلي. تم الاعتماد على مجموعة بيانات dataset IoT-23، حيث تم استخلاص البيانات من 16 سيناريو مختلف وخضعت لعملية معالجة مسبقة شاملة تضمنت تنظيف البيانات، وهندسة السمات، وتحويل المسألة إلى تصنيف ثنائي بين حركة المرور الحميدة والخبيثة. تم بعد ذلك تدريب وتقييم أربعة نماذج تصنيف مختلفة: ثلاثة نماذج تعلم آلة (Random Forest, XGBoost, LightGBM) ونموذج تعلم عميق (Neural Network)، وذلك باستخدام مقاييس أداء دقيقة مثل F1-Score.

لإثبات الجدوى العملية، تم بناء بيئة محاكاة متكاملة باستخدام جهازين افتراضيين، حيث يقوم نظام الكشف باستخدام أداة Zeek لمراقبة الشبكة وتحليلها فورياً، وتمرير البيانات عبر قناة (Pipe) إلى محرك تنبؤ يستخدم نموذج LightGBM المدرّب مسبقاً، مع عرض النتائج على لوحة معلومات تفاعلية. أظهرت النتائج تفوقاً كبيراً للنماذج القائمة على الأشجار، التي حققت دقة تجاوزت 99%، وتم اختيار نموذج LightGBM للنشر بفضل توازنه المثالي بين الدقة العالية والكفاءة الحسابية. يخلص هذا المشروع إلى إثبات فعالية وجدوى تطبيق تعلم الآلة في بناء أنظمة كشف تسلل قوية واستباقية قادرة على تأمين بيئة إنترنت الأشياء ضد التهديدات الحديثة.

الكلمات المفتاحية: *إنترنت الأشياء، أمن الشبكات، نظام كشف التسلل، تعلم الآلة، تعلم عميق، الروبوتات الشبكية (Botnets)، مجموعة بيانات IoT-23.*

Abstract

With the rapid spread of Internet of Things (IoT) networks and their accompanying security challenges, these devices have become a primary target for the formation of botnets used to launch large-scale attacks. This project addresses this problem by designing, implementing, and evaluating an intelligent Intrusion Detection System (IDS) based on machine learning and deep learning, with the goal of identifying malicious traffic in IoT networks in real-time.

The methodology was based on the IoT-23 dataset, where data was extracted from 16 scenarios and underwent comprehensive preprocessing, including data cleaning, feature engineering, and transformation into a binary classification problem (benign vs. malicious). Four different classification models were subsequently trained and evaluated: three machine learning models (Random Forest, XGBoost, LightGBM) and one deep learning model (Neural Network), using precise performance metrics such as the F1-Score.

To demonstrate practical feasibility, an integrated simulation environment was built using two virtual machines. The detection system utilizes the Zeek tool for real-time network monitoring, streaming data via a pipe to a prediction engine that employs the pre-trained LightGBM model, with results displayed on an interactive dashboard. The results showed the significant superiority of the tree-based models, which achieved accuracy metrics exceeding 99%. The LightGBM model was selected for deployment due to its optimal balance between high accuracy and computational efficiency. This project concludes by demonstrating the effectiveness and feasibility of applying machine learning to build robust and proactive intrusion detection systems capable of securing the IoT environment against modern threats.

Keywords*: Internet of Things, Network Security, Intrusion Detection System, Machine Learning, Deep Learning, Botnets, IoT-23 Dataset.*

المحتويات

قائمة الأشكال

قائمة الجداول

قائمة الاختصارات

الاختصار	المعنى باللغة الإنكليزية	المعنى باللغة العربية
APIDS	Application Protocol-based Intrusion Detection System	أنظمة كشف التسلل المستندة إلى بروتوكول التطبيق
AUC	Area Under the Curve	المساحة تحت المنحني
C&C	Command-and-Control	القيادة والتحكم
CSV	Comma-Separated Values	قيم مفصولة بفاصلة
DDoS	Distributed Denial-of-Service	حجب الخدمة الموزع
DoS	Denial-of-Service	حجب الخدمة
FN	False Negative	سلبي خاطئ (هجوم لم يتم كشفه)
FP	False Positive	إيجابي خاطئ (إنذار كاذب)
HIDS	Host-based Intrusion Detection System	نظام كشف التسلل على مستوى المضيف
HTTP	HyperText Transfer Protocol	بروتوكول نقل النص الفائق
ICMP	Internet Control Message Protocol	بروتوكول رسائل التحكم في الإنترنت
IDS	Intrusion Detection System	نظام كشف التسلل
IoT	Internet of Things	إنترنت الأشياء
IP	Internet Protocol	بروتوكول الإنترنت
IPS	Intrusion Prevention System	نظام منع التسلل
IRC	Internet Relay Chat	بروتوكول الدردشة عبر الإنترنت

LightGBM	Light Gradient Boosting Machine	آلة تعزيز التدرج الخفيفة
ML	Machine Learning	تعلم الآلة
NIDS	Network-based Intrusion Detection System	نظام كشف الاختراق على مستوى الشبكة
NN	Neural Network	الشبكة العصبونية
PDoS	Permanent Denial-of-Service	حجب الخدمة الدائم
PIDS	Protocol-based Intrusion Detection System	نظام كشف التسلل المعتمد على البروتوكول
P2P	Peer-to-Peer	الند للند
ReLU	Rectified Linear Unit	وحدة التصحيح الخطي
ROC	Receiver Operating Characteristic	خصائص التشغيل للمستقبِل
TCP	Transmission Control Protocol	بروتوكول التحكم بالإرسال
TN	True Negative	سلبي حقيقي
TP	True Positive	إيجابي حقيقي
UDP	User Datagram Protocol	بروتوكول حزم بيانات المستخدم
UI	User Interface	واجهة المستخدم
VM	Virtual Machine	آلة افتراضية
XGBoost	eXtreme Gradient Boosting	تعزيز التدرج الشديد

مقدمة عامة

أدى التطور الهائل والمستمر لأجهزة إنترنت الأشياء (IoT) إلى ترسيخ دورها كعنصر أساسي في حياتنا اليومية، حيث أصبحت جزءاً لا يتجزأ من القطاعات الحيوية كالصحة والصناعة، فضلاً عن انتشارها الواسع في المنازل الذكية. إلا أن هذا الانتشار الواسع كشف في الوقت نفسه عن تحدٍ أمني كبير؛ فهذه الأجهزة، التي تتميز غالباً بمحدودية مواردها الحاسوبية وبساطة أنظمتها، قد خلقت سطح هجوم واسع وغير مسبوق، مما جعلها هدفاً مثالياً للمهاجمين.

لقد استغل المهاجمون هذه الثغرات ببراعة، حيث قاموا بتحويل ملايين الأجهزة حول العالم إلى جيوش من الروبوتات الشبكية (Botnets)، والتي يتم توظيفها لشن بعض من أعنف الهجمات الإلكترونية وأكثرها تأثيراً، كهجمات حجب الخدمة الموزع (DDoS)وعمليات الاستطلاع المنهجي مثل مسح المنافذ الأفقي (Horizontal Port Scan) . في مواجهة هذا التهديد المتطور والديناميكي، أثبتت أنظمة الدفاع التقليدية، وخصوصاً تلك المعتمدة على التواقيع (Signature-based)، عجزها الواضح عن مواكبة سرعة وتعقيد هذه الهجمات الجديدة.

من هذا المنطلق، يهدف هذا المشروع إلى تجاوز هذا القصور عبر تبني نهج استباقي وذكي يعتمد على تقنيات تعلم الآلة والتعلم العميق. وذلك من خلال تصميم وتنفيذ وتقييم نظام متكامل لكشف التسلل (IDS) قادر على تحليل سلوك شبكات IoT بدقة .تم تحقيق ذلك عبر تدريب ومقارنة أربعة من أقوى نماذج التصنيف (Random Forest, XGBoost, LightGBM, Neural Network) على مجموعة بيانات IoT-23 الواقعية .والأهم من ذلك، أن هذا المشروع لا يكتفي بالتحليل النظري، بل يقدم إثباتاً عملياً للمفهوم عبر بناء بيئة محاكاة متكاملة، يتم فيها نشر النموذج الأفضل أداءً (LightGBM) في خط أنابيب للكشف الفوري، والذي يستخدم أداة Zeek لمراقبة الشبكة والتنبؤ بالهجمات في الزمن الفعلي.

إن الهدف النهائي لهذا العمل هو تقديم بنية نظام فعالة وموثوقة، تساهم في تطوير جيل جديد من الحلول الأمنية القادرة على حماية منظومات إنترنت الأشياء، وتضمن بذلك مستقبلاً أكثر أمناً لهذه التقنية الحيوية.

الفصل الأوّل

تعريف بالمشروع

نتعرف في هذا الفصل على الهدف من المشروع مع توضيح خطوات العمل.

مقدمة

مع التوسع الهائل لإنترنت الأشياء (IoT) وتحولها إلى عصب أساسي في القطاعات الصحية والصناعية وحتى في بيئاتنا المنزلية، برز تحدٍ أمني غير مسبوق. فهذه الأجهزة، التي تتميز غالباً ببساطة تصميمها ومحدودية مواردها الحاسوبية، أصبحت تمثل نقطة ضعف رئيسية في البنى التحتية للشبكات، مما خلق سطح هجوم واسع استغله المهاجمون ببراعة. لقد أصبحت هذه الأجهزة المنتشرة بالملايين الأداة المثالية لاستغلالها وتحويلها الى روبوتات شبكيه(Botnet) ، والذي يُستخدم لشن أعنف وأخطر الهجمات الإلكترونية. في مواجهة هذا الواقع، أثبتت أنظمة الدفاع التقليدية المعتمدة على التواقيع (signature based) قصورها في مواكبة تطور هذه التهديدات وتعقيدها. ومن هذا المنطلق، برزت الحاجة الماسة إلى تبني نهج أكثر ذكاءً واستباقية، حيث تأتي خوارزميات تعلم الآلة (Machine Learning) لتقدم حلاً فعالاً قادراً على تحليل الكم الهائل من البيانات الناتجة عن هذه الشبكات، وتعلم أنماط سلوكها الطبيعي، وبالتالي كشف أي انحراف أو هجوم محتمل في الوقت الفعلي.

هدف المشروع

هو تدريب نظام ذكاء صنعي قادر على كشف هجومات الروبوتات الشبكية (botnets) على شبكات انترنت الأشياء IoT مثل هجومات Horizontal Port Scan عبر خوارزميات التعلم الآلي وخوارزميات التعلم العميق.

توصيف المشروع

هذا المشروع عبارة عن تصميم وتنفيذ وتقييم نظام متكامل لكشف التسلل (Intrusion Detection System - IDS) مخصص لشبكات إنترنت الأشياء(IoT) ، وذلك باستخدام نماذج تعلم الآلة (Machine Learning) والتعلم العميق (Deep Learning).حيث أن النظام هو لمواجهة التهديدات الأمنية المتزايدة التي تستهدف أجهزةIoT ، وخصوصاً هجمات الروبوتات الشبكية .(Botnets) تم إنجاز المشروع عبر دورة حياة كاملة، بدءاً من معالجة مجموعة بيانات واقعية(IoT-23) ، مروراً بتدريب ومقارنة عدة نماذج تصنيف، وانتهاءً ببناء بيئة محاكاة عملية لاختبار قدرة النظام على كشف الهجمات في الزمن الفعلي.

خطوات العمل

ينقسم سير عمل المشروع إلى ثلاث مراحل رئيسية ومتسلسلة: مرحلة إعداد البيانات، مرحلة تدريب النماذج، ومرحلة التنفيذ والمحاكاة في الزمن الفعلي.

المرحلة الأولى: إعداد البيانات وتجهيزها

هذه المرحلة التأسيسية ركزت على تحويل البيانات الأولية إلى مجموعة بيانات نظيفة ومنظمة وجاهزة لعملية تدريب النماذج.

1. **اختيار مجموعة البيانات وتحديد السيناريوهات:** قمنا باختيار مجموعة بيانات **IoT-23** لواقعيتها وتنوعها. من بين 23 سيناريو متاح، تم انتقاء 16 سيناريو محدداً يحتوي على خليط من حركة المرور الخبيثة والحميدة لضمان بناء نموذج قوي.
2. **استخراج البيانات:** قمنا بمتابة سكربت بلغة بايثون لقراءة ملفات conn.log.labeled الـ 16، وهي ملفات نصية ينتجها إطار عمل Zeek. قام السكربت بتحليل هذه الملفات واستخراج السمات والقيم منها وتحويلها إلى 16 ملفاً منفصلاً بصيغة csv.
3. **المعالجة المسبقة للبيانات:** من أجل كل ملف csv. قمنا بعملية تنظيف ومعالجة شاملة عبر سكربت بايثون مخصص قام بالمهام التالية:
   - تحويل المسألة إلى تصنيف ثنائي: تم ترميز جميع أنواع الهجمات الخبيثة بالقيمة 1 والحركة الحميدة بالقيمة 0.
   - حذف السمات غير المفيدة: تم التخلص من السمات التي لا تساهم في عملية التنبؤ، مثل الطوابع الزمنية (ts)، ومعرفات الاتصال (uid)، وعناوين IP، لزيادة قدرة النموذج على التعميم.
   - معالجة القيم المفقودة (NaN): تم اتباع استراتيجية دقيقة لملء القيم المفقودة في السمات الرقمية (duration, orig_bytes, resp_bytes). للاتصالات من نوع S0 (محاولة اتصال لم تكتمل)، تم إسناد القيمة 0، بينما تم استخدام الوسيط (Median) لباقي حالات الاتصال لتجنب التأثير على توزيع البيانات.
   - تحويل أنواع البيانات: تم تحويل السمات النصية التي تحمل قيماً رقمية إلى النمط الرقمي المناسب (float).
4. **دمج البيانات:** بعد معالجة كل ملف على حدة، قمنا بدمج الـ 16 ملفاً النظيفاً في مجموعة بيانات رئيسية واحدة تحت اسم combined_dataset.csv، لتكون جاهزة لعملية التدريب.

المرحلة الثانية: تدريب النماذج وتقييمها

في هذه المرحلة، قمنا باستخدام مجموعة البيانات المجمعة لبناء وتقييم عدة نماذج بهدف اختيار الأفضل.

1. **تقسيم البيانات:** قمنا بتقسيم مجموعة البيانات النهائية إلى مجموعتي **تدريب (Training)** و**اختبار (Testing)** بنسبة 70-30%.

2. **تدريب النماذج:** قمنا بتدريب أربعة أنواع مختلفة من المصنفات لتقييم أساليب متنوعة:
   - تعلم الآلة (Machine Learning): Random Forest, XGBoost, LightGBM.
   - التعلم العميق (Deep Learning): Neural Network

لكل نموذج، تم اختيار وتبرير المعلمات الفائقة (Hyperparameters).

3. **تقييم الأداء:** قمنا بتقييم أداء كل نموذج مُدرّب على مجموعة بيانات الاختبار (unseen data) باستخدام مقاييس تقييم شاملة يمكن ايجادها من مصفوفة الارتباك، وتشمل: Accuracy, Precision, Recall, F1-Score, و AUC.
4. **حفظ النموذج:** تم حفظ النماذج المدربة في ملفات (مثل pkl. أو joblib.) لإعادة استخدامها لاحقاً في بيئة المحاكاة.

المرحلة الثالثة: التنفيذ والمحاكاة في الزمن الفعلي

هذه المرحلة النهائية تهدف إلى إثبات الجدوى العملية للنظام وقدرته على كشف الهجمات بشكل فوري.

1. **إعداد بيئة المحاكاة:** تم بناء بيئة معزولة باستخدام **VMware Workstation** تحتوي على جهازين افتراضيين بنظام Ubuntu:
   - آلة الكشف (Detection Machine): تستضيف نظام كشف التسلل.
   - آلة الهجوم (Attacking Machine): تستخدم لتوليد حركة مرور خبيثة.
2. **بناء خط أنابيب الكشف (Detection Pipeline):** تم تصميم نظام متكامل على آلة الكشف لمعالجة البيانات فورياً:
   - التقاط وتحليل الشبكة (Zeek): يقوم Zeek بمراقبة حركة المرور، وباستخدام سكربت مخصص (iot_detection.zeek) يقوم باستخلاص السمات الـ 12 المطلوبة وترميزها رقمياً في الزمن الفعلي، ثم يكتبها في ملف سجل io_t_detection.log.
   - نقل البيانات الفوري (Pipe): يُستخدم الأمر tail -f لنقل أي سطر جديد من ملف السجل إلى أنبوب (/tmp/zeek_pipe) في الذاكرة، مما يضمن وصول البيانات إلى محرك التنبؤ دون أي تأخير.
   - كود التنبؤ (predict.py): يقوم سكربت بايثون بتحميل النموذج المدرّب مسبقاً (مثل LightGBM)، ويقرأ البيانات الجديدة من الأنبوب، ويقوم بالتنبؤ بما إذا كان الاتصال "حميداً" أم "خبيثاً".
   - واجهة العرض (dashboard.py): سكربت بايثون آخر يستقبل مخرجات التنبؤ ويعرضها على شاشة المستخدم في لوحة معلومات نصية (terminal dashboard) يتم تحديثها باستمرار، مع إحصائيات وترميز لوني لتسهيل المراقبة.

3. **محاكاة الهجوم**: تم استخدام أداة Nmap من آلة الهجوم لشن هجوم مسح TCP SYN على آلة الكشف.
4. **التحقق من النتائج**: تم التأكد من أن النظام نجح في التقاط الهجوم المصطنع وتصنيفه بشكل صحيح وفوري على أنه "خبيث" وعرض التنبيه على لوحة المعلومات.

المخطط الصندوقي للنظام

يوضح المخطط في الأسفل معمارية نظام كشف تسلل (IDS) يعتمد على تعلم الآلة، وهو مقسم إلى مرحلتين أساسيتين:

- **المرحلة الأولى: التدريب (Phase 1: Training)** في هذه المرحلة، يتم تحضير النموذج. تبدأ العملية بسجلات بيانات المصدر (conn.log.labeled files) التي يتم استخراجها وتحويلها إلى ملفاتCSV . تخضع هذه الملفات للمعالجة المسبقة والتنظيف، ثم يتم دمجها في مجموعة بيانات نهائية. بعد ذلك، تُستخدم هذه المجموعة لتدريب أربعة نماذج تعلم آلة مختلفة (LightGBM, XGBoost, Random Forest, Neural Net)، ويتم في النهاية حفظ النموذج الأفضل أداءً.
- **المرحلة الثانية: المحاكاة (Phase 2: Simulation)** في هذه المرحلة، يتم استخدام النموذج في بيئة حية. حيث تقوم آلة الهجوم بإرسال حركة مرور خبيثة إلى آلة الكشف. تقوم آلة الكشف بالتقاط هذه الحركة باستخدام سكربت Zeekالذي يكتبها في ملف سجل. يتم نقل البيانات الجديدة من السجل فورياً عبر قناة (Pipe) إلى سكربت التنبؤ (predict.py)حيث يقوم هذا السكربت بتحميل النموذج المدرّب مسبقاً من المرحلة الأولى لتصنيف حركة المرور، وأخيراً، يقوم سكربت لوحة العرض (dashboard.py) بتقديم النتائج والتنبيهات للمستخدم.

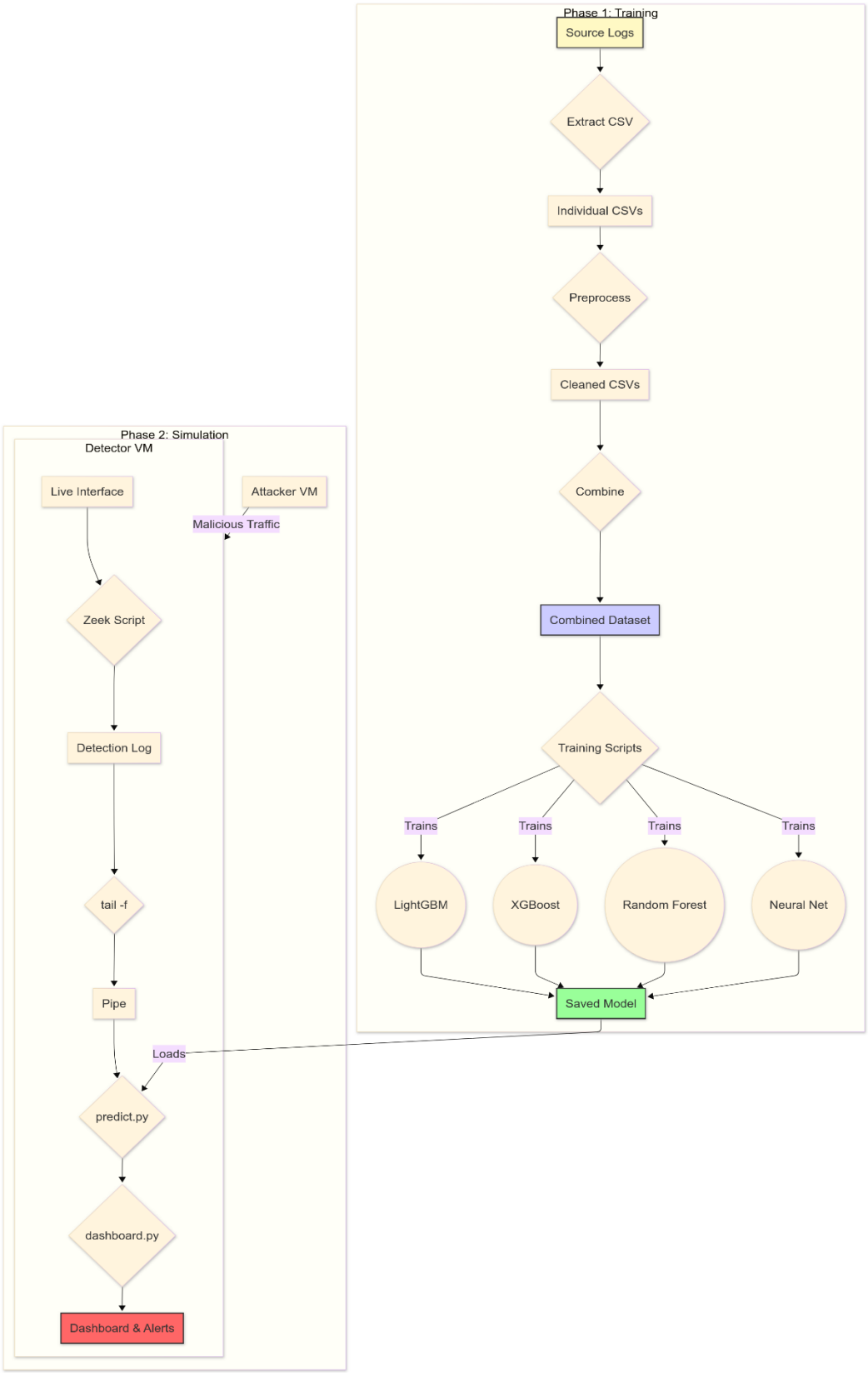


Figure1 System Block Diagram

الخاتمة

قمنا بهذا الفصل بوضع أساس للمشروع، حيث قمنا بتعريفه بشكل واضح وشامل. انطلاقاً من تحديد الهدف المتمثل في بناء نظام ذكي لكشف هجمات الروبوتات الشبكية على بيئة إنترنت الأشياء، و استعراض توصيف متكامل للمشروع وتفصيل خطوات العمل المنهجية المكونة من ثلاث مراحل رئيسية: إعداد البيانات، تدريب النماذج، والمحاكاة في الزمن الفعلي. كما تم عرض المخطط الصندوقي الذي يوضح بنية النظام المقترح.

بعد التعريف بالمشروع، سنتحدث في الفصل التالي عن انترنت الاشياء وأنظمة كشف التسلل.

الفصل الثاني

انترنت الأشياء (IoT)

نتعرف في هذا الفصل على بنية انترنت الاشياء والية عملها والهجمات عليها، بالاضافة الى التعرف على أنظمة كشف التسلل.

مقدمة

يُعرَّف مفهوم إنترنت الأشياء (IoT) بأنه ربط الكيانات المادية اليومية بشبكة الإنترنت، مما يمنحها القدرة على المعالجة الرقمية والتواصل، لتصبح بمثابة شبكة واسعة النطاق يمكن فيها لأي شيء، من آلة صنع القهوة إلى محرك طائرة، أن يقوم بجمع البيانات وتبادلها. تعمل هذه المنظومة وفق دورة متكاملة؛ حيث تبدأ بالمستشعرات (Sensors) والمشغلات (Actuators) المدمجة في الأجهزة بجمع المعلومات من بيئتها المحيطة، مثل درجة الحرارة أو الموقع أو الحركة. بعد ذلك، يتم إرسال هذه البيانات عبر إحدى تقنيات الاتصال الشبكي مثل الواي فاي (Wi-Fi) أو البلوتوث (Bluetooth) أو الجيل الخامس (5G) إلى السحابة (Cloud). وفي البيئة السحابية، تقوم برمجيات متقدمة بمعالجة هذه البيانات وتحليلها لاستخلاص رؤى معمّقة أو إطلاق إجراء معين، والذي يمكن بعد ذلك إرساله مرة أخرى إلى الجهاز أو عرضه للمستخدم عبر تطبيق مخصص. وقد أصبحت هذه التقنية جزء لا يتجزأ من الحياة المعاصرة من خلال تطبيقات شائعة مثل المنازل الذكية، حيث يتم ضبط منظمات الحرارة والإضاءة تلقائيا وفقاً لعادات المستخدم والأجهزة القابلة للارتداء، كالساعات الذكية التي تراقب المؤشرات الصحية أنيّا. على مر السنين تعرضت شبكات إنترنت الأشياء إلى العديد من الهجمات وهذا ما جعل صانعي هذه الأنظمة أكثر إدراكاً لأمن أجهزة إنترنت الأشياء. في هذا القسم سنتحدث عن آلية عمل إنترنت الأشياء وعن بعض أنواع الهجمات ومدى تأثيرها على المستخدمين والأجهزة المختلفة ومن ثم سنتحدث عن أهم الأنظمة المتبعة لتفادي هذه الهجمات.

الية عمل انترنت الاشياء

تعمل منظومة إنترنت الأشياء (IoT) وفق دورة رباعية المراحل، تحوّل من خلالها الإجراءات المادية إلى رؤى رقمية ذات قيمة.

أولاً، تقوم **الأجهزة المزودة بمستشعرات (sensors)** بجمع البيانات من بيئتها المادية المحيطة. تعمل هذه المستشعرات بمثابة حواس رقمية، حيث تلتقط معلومات متنوعة مثل درجة الحرارة، أو الحركة، أو مستويات الإضاءة، أو الموقع الجغرافي. على سبيل المثال، يقوم مستشعر في منظم حرارة ذكي بقياس درجة حرارة الغرفة باستمرار. ثانياً، يتم نقل هذه البيانات الأولية (الخام) إلى

موقع مركزي باستخدام إحدى **تقنيات الاتصال** مثل الواي فاي (Wi-Fi)، أو البلوتوث (Bluetooth)، أو الشبكات الخلوية كالجيل الخامس (G5).

ثالثاً، بمجرد وصول البيانات إلى منصة سحابية أو خادم طرفي (Edge Server)، فإنها تخضع للمعالجة. تقوم برمجيات متقدمة بتحليل هذه المعلومات ومقارنتها بالبيانات التاريخية أو بالقواعد التي حددها المستخدم، لتحويلها إلى معلومات ذات معنى. على سبيل المثال، قد تستنتج الخدمة السحابية أن قراءة درجة الحرارة الواردة من منظم الحرارة أقل من الإعداد المفضل للمستخدم. وأخيراً، تُستخدم هذه المعلومات المعالجة لإطلاق إجراء معين أو عرضها على المستخدم. قد يكون هذا الإجراء مؤتمتاً، كأن ترسل السحابة أمراً إلى منظم الحرارة لتشغيل نظام التدفئة (باستخدام مكون يسمى المشغِّل "Actuator")، أو قد يكون على شكل تنبيه يُرسل إلى تطبيق المستخدم على هاتفه الذكي، مما يسمح له باتخاذ القرار بنفسه، أي يتم تطبيق الاجراء يدويا أو اوتوماتيكيا.[1]

بنية انترنت الأشياء

تتمتع منظومة إنترنت الأشياء (IoT) بهيكلية متعددة الطبقات (multi-layered architecture)، حيث تؤدي كل طبقة دوراً محدداً في تحويل القياسات المادية إلى إجراءات أو رؤى ذات قيمة. وعلى الرغم من وجود نماذج مختلفة (مثل النماذج ثلاثية أو خماسية الطبقات)، فإن الهيكلية رباعية الطبقات تعد إطار عمل مقبولاً وعملياً على نطاق واسع لفهم آلية عملها، حيث تتمثل الطبقات في الشكل التالي:

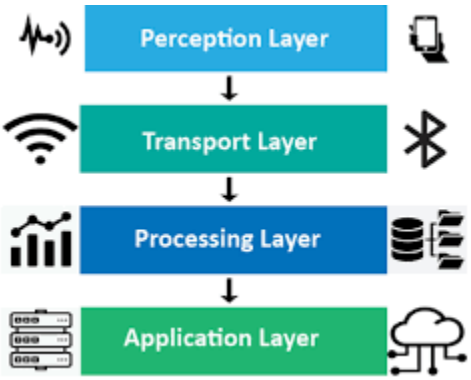


Figure2 IoT Layers

1. طبقة الاستشعار (Sensing Layer)

تمثل هذه الطبقة أساس هيكلية إنترنت الأشياء، وهي النقطة التي تُولد فيها البيانات. تتكون من الأشياء المادية التي تتفاعل مباشرة مع البيئة المحيطة. حيث تقوم بجمع البيانات من العالم المادي أو تنفيذ إجراءات فيه. وتتكون من:

- **المستشعرات (Sensors)**: أجهزة تكتشف الخصائص الفيزيائية وتقيسها، محولة إياها إلى إشارات رقمية. من أمثلتها مستشعرات درجة الحرارة، وكاشفات الحركة، ووحدات GPS، والكاميرات، والميكروفونات.
- **المشغّلات (Actuators)**: أجهزة تستقبل الأوامر وتنفذ إجراء فيزيائي. على سبيل المثال، القفل الذكي هو مشغّل يقوم بقفل أو فتح الباب، والصمام الذكي هو مشغّل يوقف تدفق المياه.

2. طبقة الشبكة (Network Layer)

بمجرد جمع البيانات، لا بد من إرسالها إلى وجهة ما. تتولى طبقة الشبكة مسؤولية النقل الآمن لجميع تلك المعلومات من طبقة الاستشعار إلى مراكز المعالجة. حيث تقوم بربط الأشياء بالإنترنت ونقل البيانات. تعمل هذه الطبقة كنظام نقل للبيانات. وتتكون من:

- تقنيات الاتصال: تشمل بروتوكولات اتصال متنوعة مثل الواي فاي (Wi-Fi)، والبلوتوث (Bluetooth)، وزيجبي (Zigbee)، و LoRaWAN، والشبكات الخلوية (G4/G5). يعتمد اختيار التقنية على عوامل مثل المدى، واستهلاك الطاقة، وحجم البيانات.
- البوابات (Gateways): تعمل هذه الأجهزة كوسطاء، حيث تقوم بتجميع البيانات من مستشعرات متعددة (والتي قد تستخدم بروتوكولات قصيرة المدى كالبلوتوث) وإعادة توجيهها إلى الشبكة الرئيسية أو السحابة، وغالباً ما تقوم بترجمة البروتوكولات في أثناء ذلك.

3. طبقة معالجة البيانات (Data Processing Layer)

هنا يتم تحويل البيانات الأولية (الخام) إلى معلومات قيمة. حيث يوصفونها بالدماغ المركزي لنظام إنترنت الأشياء، فإن هذه الطبقة مسؤولة عن التخزين والتحليل واتخاذ القرارات، حيث تقوم بمعالجة وتحليل الكميات الهائلة من البيانات الواردة من طبقة الشبكة. تقوم بالتصفية والتجميع وتطبيق القواعد أو نماذج التعلم الآلي للكشف عن الأنماط واستخلاص الرؤى. وتتكون من:

- المنصات السحابية (Cloud Platforms): توفر الخدمات السحابية القوية مثل (AWS IoT) من أمازون، و(Azure IoT) من مايكروسوفت، و(Google Cloud IoT) إمكانيات التخزين والحوسبة الهائلة اللازمة لتحليل البيانات على نطاق واسع.
- الحوسبة الطرفية (Edge Computing): بالنسبة للتطبيقات الحساسة للوقت (مثل نظام الكبح في مركبة ذاتية القيادة)، يمكن إجراء بعض المعالجة مباشرة على الجهاز أو بالقرب منه عبر بوابة طرفية. هذا يقلل من زمن الاستجابة (latency) بتجنب الحاجة إلى إرسال البيانات إلى السحابة والعودة منها.

4. طبقة التطبيقات (واجهة المستخدم – Application Layer)

وهي الطبقة العليا في الهيكلية، والتي يتفاعل معها المستخدم النهائي مباشرة. تقوم بعرض البيانات المعالجة بصيغة قابلة للاستخدام وتسمح للمستخدم بالتحكم في نظام إنترنت الأشياء. حيث يقوم بتقديم التطبيق أو الخدمة المحددة للمستخدم. إنها تجعل البيانات القادمة من طبقة المعالجة مرئية وقابلة للتنفيذ. وتتكون من:

- واجهات المستخدم (UI): تشمل تطبيقات الهواتف المحمولة، ولوحات المعلومات على الويب، ولوحات التحكم. على سبيل المثال، تطبيق على هاتف ذكي يعرض البث المباشر من كاميرا المراقبة ويتيح قفل باب عن بعد.
- ذكاء الأعمال (Business Intelligence): في البيئات الصناعية، قد تغذي هذه الطبقة برمجيات المؤسسات (مثل أنظمة تخطيط موارد المؤسسات ERP) بأتمتة سلاسل التوريد أو إنشاء تقارير عن كفاءة المصنع.

[2]

تصنيف الهجمات على منظومات إنترنت الأشياء (IoT)

إن الطبيعة التصميمية لأجهزة إنترنت الأشياء وانتشارها الواسع يخلقان **سطح هجوم (Attack Surface)** كبيراً ومعقداً. فالعديد من هذه الأجهزة تُصنَّع بتكلفة منخفضة مع تركيز أقل على الجانب الأمني، وغالباً ما تفتقر إلى القدرة على تلقي تحديثات أمنية، مما يجعلها عرضة لمجموعة واسعة من التهديدات. يمكن تقسيم الهجمات التي تستهدف هذه المنظومات بشكل أساسي إلى فئتين رئيسيتين، لكل منهما أساليبه وأهدافه، وهم هجومات فيزيائية (Physical Attacks) و هجومات الكترونية (cyber Attacks).

الهجوم الفيزيائي (Physical Attack)

تستهدف هذه الفئة من الهجمات الوصول المباشر إلى المكونات المادية للجهاز نفسه. ونظراً لأن أجهزة إنترنت الأشياء غالباً ما تكون موزعة في أماكن عامة أو غير خاضعة للمراقبة (مثل المستشعرات الزراعية أو كاميرات المراقبة الخارجية)، فإنها تكون معرضة بشكل كبير لهذا النوع من الهجوم. من أساليبه:

- **التلاعب بالهاردوير :(Hardware Tampering)** فتح الجهاز وتعديل دوائره الإلكترونية لتعطيله أو تغيير وظيفته.
- **استخلاص البيانات :(Data Extraction)** الوصول إلى شرائح الذاكرة (Memory Chips) لاستخراج معلومات حساسة مباشرة، مثل مفاتيح التشفير أو كلمات مرور الشبكة.
- **الهندسة العكسية :(Reverse Engineering)** تفكيك الجهاز وتحليل مكوناته لفهم طريقة عمله واكتشاف ثغرات يمكن استغلالها على نطاق أوسع ضد أجهزة مماثلة.

- **استنساخ الجهاز :(Device Cloning)** نسخ هوية جهاز شرعي لإنشاء جهاز خبيث يمكنه الانضمام إلى الشبكة دون اكتشافه.
[3]

الهجوم الالكتروني (Cyber Attack)

تُعد الهجمات الإلكترونية (Cyber Attacks) التهديد الأبرز الذي يواجه منظومات إنترنت الأشياء، حيث تستهدف استغلال الثغرات في بروتوكولات الاتصال والبرمجيات للوصول غير المشروع إلى البيانات أو للتحكم في الأجهزة. تنقسم هذه الهجمات إلى فئتين رئيسيتين بناءً على طبيعة تفاعل المهاجم مع النظام: **الهجمات السلبية (Passive Attacks)** التي تركز على المراقبة والتنصت، و**الهجمات النشطة (Active Attacks)** التي تتضمن تعديلاً أو تعطيلاً متعمداً لعمل النظام.

الهجوم السلبي (Passive Attack): المراقبة وجمع المعلومات

الهدف الأساسي للهجوم السلبي هو الحصول على معلومات حساسة من الشبكة دون تعديل أي بيانات أو التأثير على عملها، مما يجعل اكتشافه أمراً صعباً للغاية. ومن أنواعه:

- **التنصت (Eavesdropping)**:حيث يقوم المهاجم باعتراض والتقاط حزم البيانات أثناء إرسالها عبر الشبكات اللاسلكية (مثل Wi-Fi, Bluetooth, Zigbee). إذا كانت هذه البيانات غير مشفرة، يمكن للمهاجم قراءتها مباشرة. وهذا يؤدي الى سرقة معلومات بالغة الحساسية مثل بيانات تسجيل الدخول، أو تفاصيل شخصية للمستخدم، أو قراءات من مستشعرات طبية، أو حتى الأوامر المرسلة إلى الأجهزة. هذه المعلومات يمكن استخدامها لاحقاً لشن هجوم نشط.
- **تحليل حركة المرور (Traffic Analysis): حيث** حتى لو كانت البيانات مشفرة، لا يزال بإمكان المهاجم جمع معلومات قيمة عبر مراقبة البيانات الوصفية (Metadata) للاتصال. يقوم بتحليل أنماط حركة المرور، مثل: من يتصل بمن، ومتى، وكم مرة، وحجم البيانات المرسلة. وهذا يمكّن المهاجم من استنتاج سلوكيات المستخدمين. على سبيل المثال، يمكنه معرفة أوقات وجود الأشخاص في المنزل أو غيابهم بناءً على نشاط أجهزتهم الذكية، أو تحديد نوع الجهاز المستخدم، مما يساعده على تخطيط هجوم أكثر استهدافاً وفعالية.

الهجوم النشط (Active Attack): التخريب والتلاعب

في هذا النوع، يتفاعل المهاجم بشكل مباشر مع النظام بهدف تعطيل الخدمات، أو تعديل البيانات، أو تدميرها، أو السيطرة على الأجهزة. هذه الهجمات أكثر وضوحاً من الهجمات السلبية ولكنها أكثر تدميراً. من أنواعها:

- **هجمات حجب الخدمة (DoS Attacks)**: حيث يتم في هذا الهجوم إغراق جهاز مستهدف (كمستشعر أو بوابة) أو الشبكة بأكملها بعدد كبير من البيانات أو طلبات الاتصال الوهمية التي تفوق قدرته على المعالجة، مما يؤدي إلى

استهلاك كامل لموارده (كالبطارية، المعالج، الذاكرة) وتوقفه عن العمل. وهو خطير للغاية على شبكات انترنت الاشياء، حيث يمكنه استنزاف بطارية مستشعر بعيد وجعله عديم الفائدة، أو تعطيل كاميرا أمنية في وقت حرج، أو جعل الأجهزة الطبية غير مستجيبة.

- **هجمات الرجل في المنتصف (Man-in-the-Middle - MITM): حيث** يقوم المهاجم بوضع نفسه بين جهازين يتواصلان (مثلاً، مستشعر والسحابة)، ويعترض الاتصال، ويمكنه قراءة البيانات أو تعديلها قبل إعادة إرسالها إلى وجهتها. مما يمكّن المهاجم من التلاعب بالبيانات (Data Tampering)، كأن يغير قراءة درجة حرارة في مصنع من مرتفعة إلى طبيعية مما يسبب كارثة، أو يمنع وصول أمر إيقاف إلى جهاز صناعي.
- **التزييف وهجمات الانتحال (Spoofing and Sybil Attacks)**
  - **Spoofing (التزييف):** يقوم المهاجم بانتحال هوية جهاز شرعي عبر تزييف عنوانه الفريد (مثل عنوان MAC). يسمح له ذلك بإرسال بيانات ضارة تبدو وكأنها قادمة من مصدر موثوق.
  - **Sybil Attack (هجوم سيبيل):** يقوم المهاجم بإنشاء عدد كبير من الهويات المزيفة داخل الشبكة. يستخدم هذا الهجوم للتلاعب بأنظمة التصويت في الشبكات الموزعة أو للتأثير على بروتوكولات التوجيه.
- **هجمات الثقب والتشويش (Hole Attacks and Jamming)**
  - **Sinkhole Attack:** يعلن فيه المهاجم (عقدة خبيثة) على أنه يمتلك أفضل مسار للوصول إلى البوابة الرئيسية، فتنجذب إليه حركة البيانات من العقد المجاورة، مما يمكنه من تحليل كل البيانات أو تجاهلها.
  - **Jamming (التشويش):** هجوم على الطبقة الفيزيائية، حيث يقوم المهاجم ببث إشارات ضوضاء قوية على نفس التردد الذي تستخدمه أجهزة إنترنت الأشياء، مما يؤدي إلى قطع الاتصال بالكامل في منطقة معينة.
- **المدخلات الضارة والتلاعب بالبيانات (Data Tampering & Malicious Inputs): حيث** تتضمن إرسال بيانات مُصممة بشكل خبيث إلى جهاز IoT لاستغلال ثغرة في كيفية معالجته للمدخلات، أو تعديل البيانات الموجودة بالفعل على الجهاز أو أثناء نقلها (كما في هجوم MITM). يمكن أن يؤدي إلى تعطل جهاز انترنت اشياء، أو تنفيذ أوامر غير مصرح بها، أو اتخاذ قرارات خاطئة بناءً على بيانات مغلوطة، مما قد يسبب أضراراً مادية كبيرة.

[4]

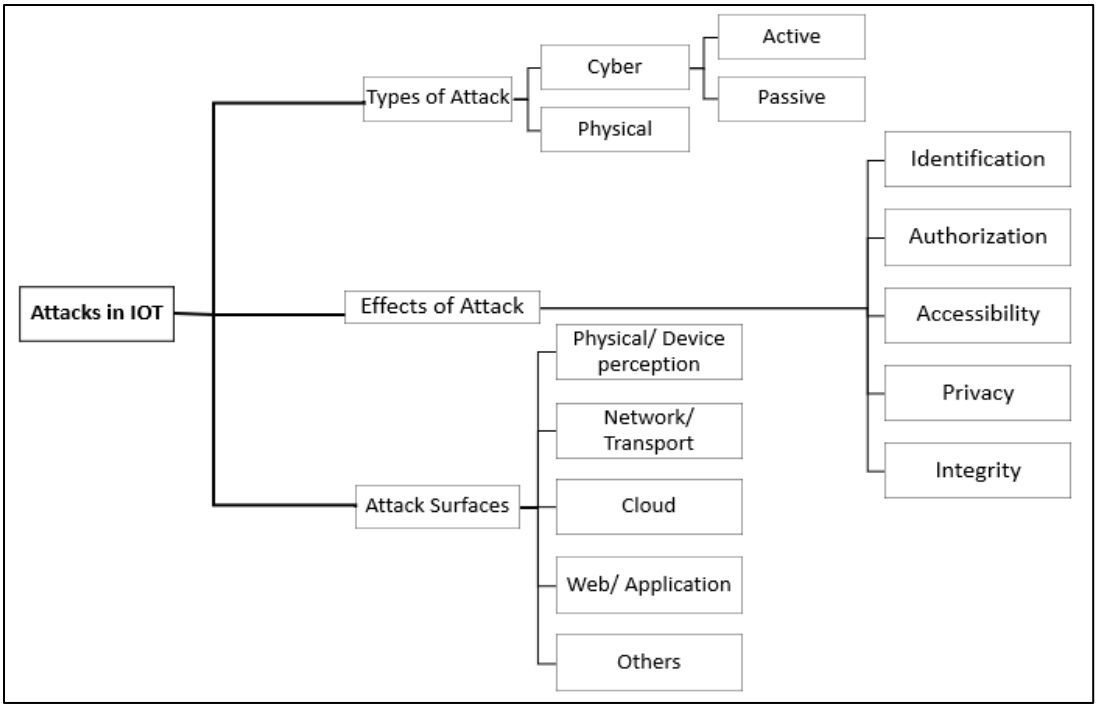


Figure3 Attacks in IoT

الروبوتات الشبكية Botnets

تمثل شبكات الروبوت (البوت نت Botnets) بنية تحتية هجومية متطورة، تتألف بشكل أساسي من مجموعة من الأجهزة المخترقة (البوتات) التي تعمل تحت إمرة خادم للقيادة والتحكم (C&C) يديره مهاجم يُعرف بـ "botmaster". إن الهيكلية التشغيلية لهذه الشبكات تبدأ بمرحلة الانتشار، حيث تقوم البرمجية الخبيثة بمسح نطاقات واسعة من الشبكة بشكل منهجي بحثاً عن أجهزة تعاني من ثغرات أمنية معروفة أو تستخدم بيانات اعتماد ضعيفة (مثل كلمات المرور الافتراضية). فور نجاح عملية الاختراق، يدمج البوت نفسه في نظام تشغيل الجهاز ويدخل في حالة انتظار، منتظراً التعليمات من خادمه المركزي لتنفيذ أنشطة ضارة بشكل منسق. وتتجلى قوة هذه الشبكات في قدرتها على شن هجمات الإغراق التعاوني(collaborative flooding)، التي تهدف إلى إغراق جهاز مستهدف بفيض من الطلبات غير الشرعية، مما يستنزف موارده ويمنعه من معالجة الطلبات المشروعة، وينتهي به الأمر إلى هجوم حجب الخدمة الموزع (DDoS) .بالإضافة إلى ذلك، تشمل أنشطتها الخبيثة الأخرى التنقيب عن العملات المشفرة، أو سرقة البيانات عبر تسجيل ضغطات المفاتيح، أو شن حملات البريد العشوائي.

على الرغم من أن أول بوت نت لإنترنت الأشياء، المعروف باسم **Linux.Hydra**، قد تم اكتشافه في عام 2008، إلا أن الخطر الحقيقي لهذه الظاهرة لم يظهر بوضوح للمجتمع الأمني حتى ظهور بوت نت **Mirai** في سبتمبر 2016، والذي شكل نقطة تحول محورية. استهدف هجوم ميراي الأول مدونة Krebs on Security بحركة مرور هائلة بلغت 620 جيجابت في الثانية، وهو رقم قياسي في ذلك الوقت. لكن التداعيات الأكثر خطورة أتت من إتاحة شفرة المصدر الخاصة بميراي للعلن، مما

أدى إلى ظهور عشرات المتغيرات وألهم جيلاً جديداً من شبكات الروبوت. وبالفعل، في الشهر التالي، تمكن متغير من ميراي من شل حركة مزود الخدمةDyn ، في هجوم اعتبر الأضخم في التاريخ آنذاك. لقد كانت ميراي مجرد بداية، حيث شهدت السنوات اللاحقة تطوراً مقلقاً في قدرات هذه الشبكات؛ ففي عام 2017، ظهر **BrickerBot** الذي لم يكتفِ بتعطيل الأجهزة مؤقتاً، بل كان قادراً على تدميرها بشكل دائم عبر هجوم حجب الخدمة الدائم (PDoS). وفي عام 2018، برز بوت نت **JenX**الذي أظهر تطوراً في منهجية الانتشار، حيث استخدم خوادم قوية لتسريع عملية مسح الشبكة بحثاً عن ضحايا جدد، مما يعكس تسارعاً مستمراً في تطور هذه التهديدات وتزايد خطورتها على البنى التحتية الرقمية.

[5]
تصنَّف شبكات الروبوت (Botnets) بشكل أساسي بناءً على هيكلية القيادة والتحكم - (Command and Control C&C)، وهي الطريقة التي يتواصل بها المهاجم (Botmaster) مع بوتاته من الأجهزة المخترِقة (Bots) تحدد هذه الهيكلية مدى سرعة استجابة الشبكة، وقدرتها على الصمود في وجه محاولات التفكيك. هناك نموذجان أساسيان لهذه الهيكلية: المركزي (Centralaized)، واللامركزي (الند للند)(Peer to Peer).

الهيكلية المركزية (Centralized Architecture)

تعتبر هذه هي الطريقة التقليدية والأبسط لبناء شبكات الروبوت، حيث تعتمد على نموذج الخادم والعميل-Client) (Server، وتتبع ما يعرف بطوبولوجيا النجمة(Star Topology) حيث أنه في هذا النموذج، يوجد خادم مركزي واحد أو عدد قليل من الخوادم تمثل مركز الشبكة. تقوم جميع الروبوتات المصابة بالاتصال مباشرة بهذا الخادم بشكل دوري لتلقي الأوامر. يقوم الـ Botmaster بالوصول إلى هذا الخادم المركزي وإصدار أمر واحد، ويقوم الخادم بدوره بنشره فوراً إلى جميع الروبوتات المتصلة به لتنفيذه بشكل متزامن. ويعتمد على بروتوكول الاتصال IRC (Internet Relay Chat) الذي قديما كان الأكثر استخداماً بسبب بساطته. لكن في الشبكات الحديثة، تحول المهاجمون إلى استخدام بروتوكولات أكثر شيوعاً مثل HTTP/HTTPS، لأن حركة مرور البيانات الخاصة بها تبدو كأنها تصفح عادي للإنترنت، مما يجعل من الصعب على أنظمة الدفاع اكتشافها وحظرها. إنّ هذه الطريقه لها ميزة أساسية وهي سرعة وسهولة التحكم يمكن للمهاجم نشر الأوامر وتحديث البرمجيات الخبيثة بسرعة فائقة لجميع الروبوتات في وقت واحد. ولكنها بالطبع لها مشكاة وهي وجود نقطة ضعف مركزية (Single Point of Failure) بمجرد أن تتمكن السلطات أو شركات الأمن من تحديد موقع الخادم المركزي وتعطيله، تفقد الشبكة بأكملها قائدها وتصبح عديمة الفائدة.

الهيكلية اللامركزية – الند للند (Peer-to-Peer - P2P)

ظهر هذا النموذج كتطور للتغلب على ضعف الهيكلية المركزية. في هذا التصميم، لا يوجد خادم مركزي، بل تعمل الروبوتات كشبكة مترابطة ومكتفية ذاتياًحيث أنه كل روبوت في الشبكة لديه قائمة بعناوين عدد من الروبوتات الأخرى (الأقرانpeers). عندما يريد الـ Botmaster إصدار أمر، فإنه يقوم بحقنه في عدد قليل من الروبوتات التي يمكنه الوصول إليها. بعد ذلك، يقوم

كل روبوت تلقى الأمر بتمريره إلى أقرانه في القائمة الخاصة به، والذين يقومون بدورهم بتمريره إلى أقرانهم. تنتشر الأوامر عبر الشبكة حتى تصل إلى جميع الروبوتات، ويعتمد على **بروتوكولات الاتصال** الند للند المخصصة أو المعروفة (مثل التي تستخدم في برامج مشاركة الملفات) لتبادل المعلومات والأوامر بين الروبوتات. **يتمتع هذا النموذج بالمرونة والصلابة العالية** .نظراً لعدم وجود رأس مركزي، فإن تعطيل عدد قليل من الروبوتات لا يؤثر على الشبكة بأكملها. تفكيك مثل هذه الشبكة يتطلب جهداً هائلاً وتحديد عدد كبير جداً من الأجهزة المصابة. ولكن بالطبع لديه مشكلة وهي **بطء انتشار الأوامر وتعقيد التحكم**. حيث يستغرق وصول الأمر إلى جميع أجزاء الشبكة وقتاً أطول مقارنة بالنموذج المركزي. كما أن تصميم وإدارة مثل هذه الشبكة أكثر تعقيداً بالنسبة للمهاجم.

[6]

دورة الحياة التشغيلية لروبوتات إنترنت الأشياء (IoT Botnets lifecycle)

إن دورة حياة شبكات الروبوت الخاصة بإنترنت الأشياء هي عملية منهجية متعددة المراحل، يمكن تقسيمها إلى مرحلتين رئيسيتين :مرحلة إنشاء البوت نت وانتشاره، ومرحلة إطلاق الهجوم المنسق .توضح هذه الدورة كيف يتحول جهاز IoT شرعي إلى أداة في هجوم إلكتروني واسع النطاق.

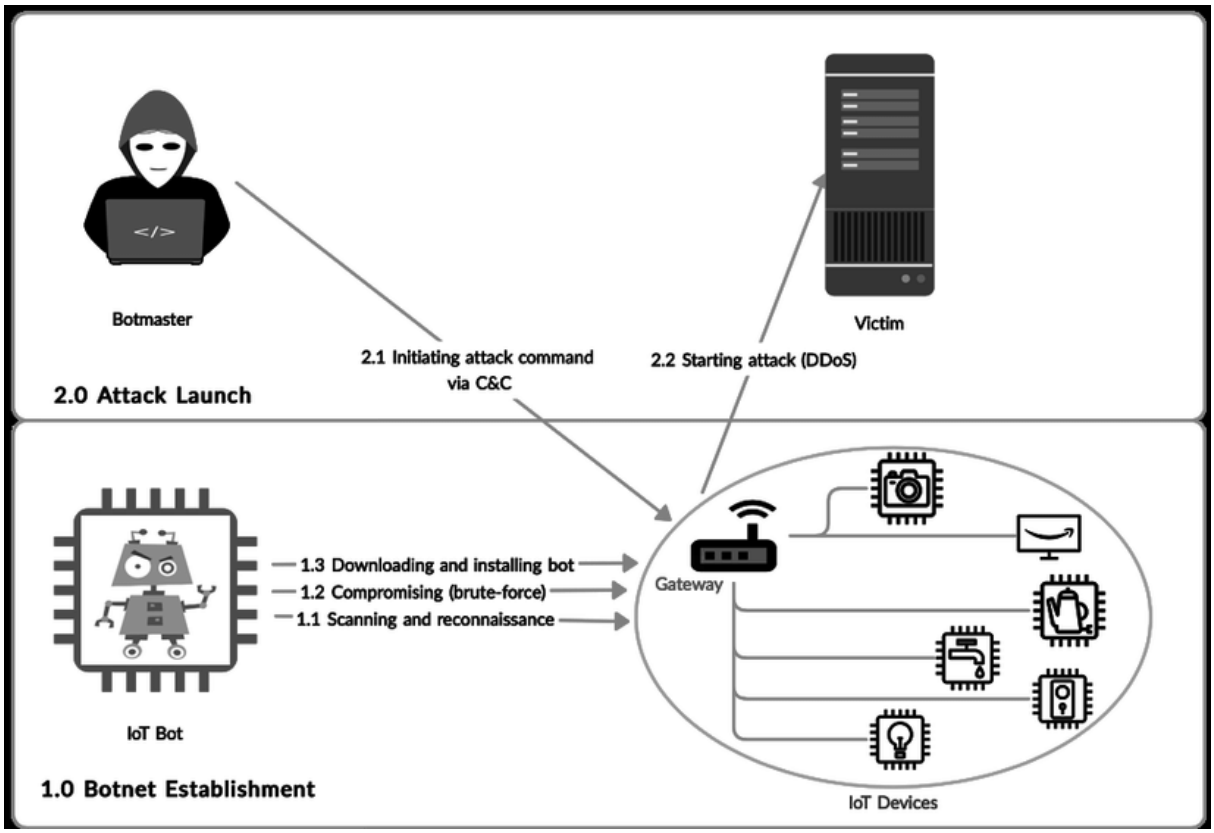


Figure4 IoT Botnet lifecycle

1. مرحلة إنشاء البوت نت وانتشاره (Botnet Creation & Propagation)

تُعد هذه المرحلة حجر الأساس لبناء جيش الروبوتات، وتتضمن عدة خطوات دقيقة لتحديد الأهداف وإصابتها:

أ- **الاستطلاع والعدوى(Reconnaissance and Infection)** :

تبدأ هذه المرحلة الأولية بقيام الروبوت (أو الماسح الأولي) بعملية **استطلاع (Scanning)** آلية واسعة النطاق عبر شبكة الإنترنت. يتم فحص عناوين IP بشكل عشوائي للبحث عن منافذ مفتوحة تشير إلى وجود أجهزة IoT، وخصوصاً المنافذ غير الآمنة مثل **Telnet (Port 23)**و **SSH (Port 22)**.بمجرد تحديد هدف محتمل، تبدأ محاولة العدوى بإحدى طريقتين رئيسيتين:

- **هجوم بيانات الاعتماد الضعيفة :**وهي الطريقة الأكثر شيوعاً ونجاحاً، حيث يقوم الروبوت بشن هجوم **(Brute-Force Attack)**، مستخدماً قائمة من كلمات المرور الافتراضية والشهيرة) .
- **استغلال الثغرات المعروفة :** يستهدف الروبوت الثغرات الأمنية الموثقة في البرامج الثابتة (Firmware)القديمة وغير المحدثة للأجهزة، ويقوم بإرسال شفرة استغلال (Exploit Code) للسيطرة على الجهاز.

ب- **التثبيت والتحصين(Installation and Fortification)**

بعد نجاح الاختراق، لا يتم تثبيت الروبوت الكامل فوراً. عادةً ما يتم تحميل أداة تنزيل صغيرة (Dropper) تقوم بتحديد بنية معالج الجهاز الضحية) مثل (ARM, MIPS, x86 ومن ثم تقوم بتنزيل وتثبيت النسخة المتوافقة من البرمجية الخبيثة الكاملة. بعد ذلك، تأتي خطوة التحصين الحاسمة، حيث يقوم الروبوت بتأمين الجهاز لصالحه عبر:

- **إغلاق الثغرة :**يقوم بإغلاق المنفذ أو الثغرة التي استخدمها للدخول.
- **قتل العمليات المنافسة :**يقوم بمسح ذاكرة الجهاز وإنهاء أي عمليات تابعة لبرمجيات خبيثة أخرى.

وهكذا يحتكر موارد الجهاز (قوة المعالج، عرض النطاق الترددي) وضمان الولاء الحصري لل Botmaster الحالي.

2. مرحلة إطلاق الهجوم المنسق

عندما يصبح لدينا عدد كافٍ من الروبوتات وتوصيلها بمركز القيادة، تصبح الشبكة جاهزة لتنفيذ الأوامر. ويتم عندها الاتي:

أ- **إصدار الأوامر :**يقوم الـ Botmaster بإرسال أوامره عبر خادم **القيادة والتحكم (C&C)** في الهياكل المركزية، يتم إرسال الأمر مباشرة إلى جميع الروبوتات. أما في هياكل **الند للند(P2P)** ، يتم حقن الأمر في عدد قليل من الروبوتات التي تقوم بنشره لبقية أقرانها في الشبكة كما شرحنا سابقا.

ب- **تنفيذ الهجوم :**عند تلقي الأوامر، تبدأ جميع الروبوتات بتنفيذ الهجوم بشكل متزامن. تتنوع طبيعة هذه الهجمات بشكل كبير، وتشمل:

- **هجمات حجب الخدمة الموزعة (DDoS):** إغراق الهدف بحركة مرور هائلة لجعله غير متاح.
- **هجمات حجب الخدمة الدائمة(PDoS)** : إرسال تحديثات برامج ثابتة ضارة تهدف إلى (Bricking)الجهاز، أي تدميره بشكل دائم.
- **التنقيب عن العملات المشفرة :(Cryptomining)** استغلال قوة المعالجة الجماعية للأجهزة لتعدين العملات الرقمية لصالح المهاجم.
- **أنشطة أخرى :**مثل سرقة البيانات، أو شن هجمات تصيد(Phishing) ، أو استخدام الأجهزة كبروكسي لإخفاء هوية المهاجم.

[7]

تحليل هجمات حجب الخدمة (DoS) وحجب الخدمة الموزع (DDoS)

تُعد هجمات حجب الخدمة (Denial-of-Service - DoS) إحدى أخطر التهديدات التي تستهدف مبدأ الإتاحة (Availability)، وهو أحد أركان ثلاثية أمن المعلومات (CIA) (CIA: Confidentiality Integrity Availability Triad).الهدف الأساسي لهذه الهجمات هو جعل خدمة أو مورد رقمي غير متاح للمستخدمين الشرعيين، إما بشكل مؤقت أو دائم، عبر استنزاف موارده بشكل كامل.

عندما يكون الهجوم من مصدر واحد، يُعرف بـ .DoS أما عندما يتم شنه بشكل منسق من مصادر متعددة، فإنه يتطور إلى هجوم حجب الخدمة الموزع(Distributed Denial-of-Service - DDoS) ، وهو النوع الأكثر شيوعاً وتدميراً اليوم، ويتم تنفيذه عادة عبر شبكات الروبوت (Botnets)

تكمن خطورة هجمات DDoS في صعوبة التخفيف من أثرها، حيث أن سيل الطلبات الخبيثة يأتي من آلاف الأجهزة الموزعة جغرافياً وذات العناوين المختلفة(IPs) ، مما يجعل التمييز بين حركة المرور الشرعية وحركة المرور الهجومية أمراً شبه مستحيل بالطرق التقليدية. وبالنسبة لأجهزة إنترنت الأشياء (IoT) على وجه الخصوص، فإن لهذه الهجمات أثراً إضافياً يتمثل في استنزاف البطارية، حيث أن إبقاء الجهاز في حالة نشاط ومعالجة مستمرة للطلبات يستهلك طاقته المحدودة بسرعة، مما قد يؤدي إلى تعطله بشكل دائم.

يمكن تصنيف هجمات DoS و DDoS بشكل عام إلى ثلاث فئات رئيسية بناءً على الطبقة التي تستهدفها في حزمة البروتوكولات الشبكية

.

- **هجمات الحجم(Volumetric Attacks)**

هذه هي الفئة الأكثر شيوعاً لهجماتDDoS ، وهدفها هو استهلاك كامل عرض حزمة الشبكة (Network Bandwidth) للهدف.حيث يقوم المهاجم بإرسال كمية هائلة من البيانات إلى الهدف، تفوق قدرة اتصال الإنترنت لديه على الاستيعاب. مثال عليهم UDP Floods و ICMP Floods، حيث يتم إغراق الهدف بحزم بيانات لا تتطلب مصادقة، مما يجبره على استهلاك كل نطاقه الترددي في معالجتها.

- **هجمات البروتوكول(Protocol Attacks)**

تستهدف هذه الهجمات استنزاف موارد النظام (System Resources) للأجهزة الشبكية الوسيطة، مثل جدران الحماية (Firewalls)وموازنات التحميل(Load Balancers) ،حيث يستغل المهاجم نقاط الضعف في بروتوكولات الشبكة مثل TCPفي الطبقة الرابعة (transport layer) ,ويتم إرسال طلبات اتصال مُصاغة بشكل خبيث تستهلك كل ذاكرة وقوة معالجة الجهاز المستهدف أثناء محاولته التعامل معها. مثال عليهم SYN Flood : وهو الهجوم الأشهر في هذه الفئة، حيث يرسل المهاجم عدداً كبيراً من طلبات الاتصال (SYN) ولكنه لا يكمل عملية المصادقة الثلاثية(Three-way Handshake) ، مما يترك المنافذ مفتوحة وفي حالة انتظار، ويستنزف موارد الخادم حتى يتوقف عن استقبال أي اتصالات جديدة.

- **هجمات طبقة التطبيقات(Application Layer Attacks)**

تعتبر هذه الهجمات الأكثر تطوراً وصعوبة في الكشف، حيث تستهدف موارد التطبيق (Application Resources) نفسه، مثل خادم الويب أو قاعدة البيانات. حيث يقوم المهاجم بإنشاء طلبات تبدو وكأنها طلبات شرعية من مستخدمين حقيقيين (مثل طلب تحميل صفحة ويب أو إجراء بحث في قاعدة بيانات). تكون هذه الطلبات بطيئة ومعقدة، وتُرسل بأعداد كبيرة لإرهاق موارد الخادم (CPU, RAM) وجعله غير قادر على خدمة المستخدمين الفعليين.

  - **أمثلةHTTP Flood :** حيث تقوم الروبوتات بإرسال آلاف طلبات HTTP GET أو POST إلى خادم ويب بشكل متزامن، مما يؤدي إلى انهياره.

[8]

أنواع الروبوتات الشبكية (Botnets)

فيما يلي سنتحدث عن بعض أنواع الروبوتات الشبكية المشهورة والموجودة في مجموعة البيانات المختارة في مشروعنا وهي IoT-23 dataset.

أ- روبوت Miria

شبكة ميراي (Mirai) تمثل لحظة محورية في تاريخ التهديدات السيبرانية. لقد تميزت هذه الشبكة بقدرتها على بناء عدد هائل من أجهزة إنترنت الأشياء (IoT) المخترقة، مثل مسجلات الفيديو الرقمية (DVRs) وكاميرات بروتوكول الإنترنت (IP cameras). تميزت آلية العدوى الخاصة بها بالبساطة الخادعة والفعالية التدميرية، حيث كانت تقوم بمسح شبكة الإنترنت بشكل مستمر بحثاً عن أجهزة إنترنت الأشياء التي ما زالت تستخدم بيانات الاعتماد الافتراضية للمصنع (أسماء المستخدمين وكلمات المرور). فور نجاحها في الوصول، تقوم ميراي بإصابة الجهاز وتحويله إلى زومبي (zombie) ينتظر الأوامر. في عام 2016، تم توظيف هذه الشبكة كسلاح لشن بعض أضخم هجمات حجب الخدمة الموزع (DDoS) التي شوهدت في ذلك الوقت، حيث قامت بشل موقع الصحفي المتخصص في الأمن السيبراني (براين كريبسBrian Krebs) بحركة مرور بلغت 620 جيجابت في الثانية، ولاحقاً قامت بتعطيل خدمات إنترنت رئيسية مثل تويتر ونتفليكس وسبوتيفاي عبر استهداف مزود خدمة DNS، شركة Dyn. وقد ترسخ الإرث الدائم لميراي عندما قام منشئها بنشر شفرتها المصدرية للعلن، مما مكّن عدداً لا يحصى من المهاجمين الآخرين من إنشاء متغيراتهم الخاصة، وضمنَ أن تأثيرها التخريبي سيستمر لسنوات قادمة.

[9]

ب- روبوت Okiru

تُعد شبكة بوت نت أوكيرو (Okiru) أحد المتغيرات الهامة وواسعة الانتشار لشبكة ميراي (Mirai). تم تحديدها لأول مرة في أوائل عام 2018، وتستهدف أوكيرو بشكل خاص المنظومة الواسعة وغير الآمنة في كثير من الأحيان من أجهزة إنترنت الأشياء التي تعمل بمعالجات قائمة على بنية ARC. توجد هذه المعالجات بشكل شائع في مليارات الأجهزة المدمجة مثل أجهزة التوجيه (الراوترات)، ومسجلات الفيديو الرقمية، وكاميرات بروتوكول الإنترنت. اتباعاً لنفس الاستراتيجية البسيطة والفعالة لسلفها، تنتشر أوكيرو عبر مسح شبكة الإنترنت بشكل نشط بحثاً عن الأجهزة التي تحتوي على منافذ Telnet مفتوحة، ثم تحاول تسجيل الدخول باستخدام قائمة من بيانات الاعتماد الافتراضية للمصنع أو كلمات المرور الضعيفة. بمجرد اختراق الجهاز، يتم تجنيده في شبكة البوت نت ليصبح جاهزاً للاستخدام كسلاح في هجمات حجب الخدمة الموزع (DDoS) واسعة النطاق. إن ظهور أوكيرو ومتغيراتها يؤكد على التهديد المستمر الذي يمثله نشر شفرة المصدر الخاصة بميراي للعلن، حيث يوضح مدى سهولة تكييف شبكات الروبوت الجديدة لاستهداف معماريات أجهزة مختلفة وتوسيع جيش الزومبي العالمي من أجهزة إنترنت الأشياء.

[10]

ت- روبوت Torii

إن شبكة بوت نت توري(Torii) ، التي تم اكتشافها لأول مرة في عام 2018، تمثل قفزة نوعية في درجة تعقيد البرمجيات الخبيثة التي تستهدف إنترنت الأشياء. فعلى عكس سابقاتها مثل ميراي، التي كانت تركز بشكل شبه حصري على شن هجمات حجب الخدمة الموزع(DDoS) ، فقد تم تصميم توري لتحقيق البقاء طويل الأمد على الأجهزة، وتسريب البيانات، وتنفيذ أوامر أكثر تعقيداً .وتبرز هذه الشبكة بفضل ميزاتها المتقدمة، حيث يمكنها إصابة مجموعة واسعة من معماريات المعالجات مثل (x86, ARM, MIPS)، وتستخدم طبقات متعددة من الاتصالات المشفرة لإخفاء حركة مرور بيانات القيادة والتحكم

(C&C)، وتعتمد على عملية عدوى متعددة المراحل لتأسيس موطئ قدم حصين على الجهاز. وبدلاً من الاعتماد على هجمات القوة الغاشمة على كلمات المرور الضعيفة، لوحظ أن توري تحاول استغلال الثغرات الأمنية المعروفة، مما يظهر نهجاً أكثر استهدافاً. إن هذا التركيز على التخفي والبقاء الدائم يوحي بأن توري لم تُبنَ كأداة بسيطة لشن هجماتDDoS ، بل كمنصة متعددة الاستخدامات للتجسس أو لبناء جيش زومبي عالي المرونة والقوة لشن هجمات أكثر تعقيداً في المستقبل.

[11]

تأثير الهجمات على إنترنت الأشياء

أدت الهجمات المختلفة على شبكات إنترنت الأشياء لحدوث أضرار جسيمة طالت أجهزة إنترنت الأشياء ومستخدمي الشبكة، فلم تقتصر الأضرار على منع وصول الخدمة إلى المستخدمين الحقيقيين بل امتد تأثيرها لتصبح قادرة على العبث بالبيانات المنقولة عبر الشبكة والتعدي على خصوصية المستخدمين. إن لهذا التأثير أبعاد حافلة بالمخاطرة على جميع المستويات التقنية والصحية والاقتصادية وذلك لكون أجهزة إنترنت الأشياء أصبحت محرك أساسي لجميع التقانات في جميع القطاعات المختلفة. يقدم الشكل التالي قائمة كاملة بأشكال مختلفة من الهجمات، بما في ذلك تأثيرها على أجهزة إنترنت الأشياء.

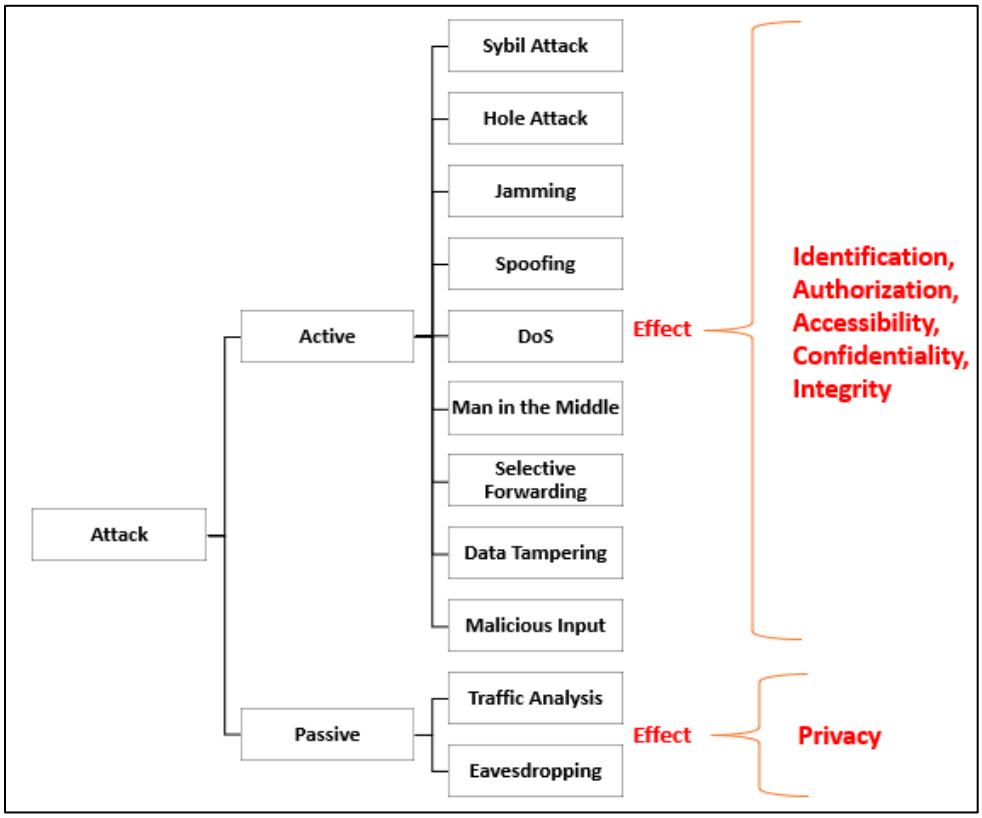


Figure5 attacking IoT effects

متطلبات الأمن الأساسية في أنظمة إنترنت الأشياء

عند تصميم أي بنية أمنية لأنظمة إنترنت الأشياء، يجب أن ترتكز على تحقيق مجموعة من الأهداف الأساسية لضمان حماية النظام من الهجمات المتنوعة. هذه المتطلبات مستوحاة من المبادئ الكلاسيكية لأمن المعلومات ولكن مع تكييفها لتناسب تحديات بيئة IoT الفريدة وهم:

1- **الإتاحة (Availability)** : وهو ضمان استمرارية عمل النظام وقدرة المستخدمين المصرح لهم على الوصول إلى الخدمات والبيانات في أي وقت يحتاجون إليها دون انقطاع أو تعطيل. في عالم إنترنت الأشياء، يكتسب هذا المبدأ أهمية حرجة وتحديات فريدة، لأن توقف جهاز واحد عن العمل قد لا يعني مجرد انقطاع خدمة رقمية، بل قد يؤدي إلى عواقب مادية خطيرة. إن التهديد الأبرز للإتاحة في بيئة IoT يأتي من هجمات حجب الخدمة(DoS) ، حيث يقوم المهاجم بإغراق جهاز ذي موارد محدودة (معالج ضعيف وذاكرة صغيرة) بسيل من الطلبات الوهمية التي لا يستطيع معالجتها، مما يؤدي إلى استنزاف طاقته وتوقفه عن العمل. بالإضافة إلى ذلك، تعتبر هجمات التشويش(Jamming) ، التي تبث ضوضاء على ترددات الاتصال اللاسلكي، تهديداً خطيراً آخر، حيث تمنع الجهاز من التواصل مع الشبكة بشكل كامل. [12]

2- **السرية (Confidentiality)** و**الخصوصية (Privacy)** :على الرغم من أن مفهومي السرية (Confidentiality)والخصوصية (Privacy) غالباً ما يُستخدمان بالتبادل، إلا أنهما يمثلان مطلبين أمنيين متمايزين وإن كانا متكاملين، وتتجلى أهميتهما بشكل خاص في بيئة إنترنت الأشياء. تُعنى السرية بالجانب التقني لحماية البيانات، وهي تضمن أن المعلومات غير قابلة للقراءة أو الوصول إليها إلا من قبل الأطراف المصرح لها بذلك. يتم تحقيق ذلك بشكل أساسي عبر آليات التشفير (Encryption) القوية التي تحمي البيانات سواء أثناء نقلها عبر الشبكة أو أثناء تخزينها على الأجهزة أو في السحابة، مما يجعلها منيعة ضد هجمات التنصت(Eavesdropping) .
أما الخصوصية، فهي مفهوم أوسع وأكثر ارتباطاً بحقوق المستخدم، حيث تتعلق بالتحكم فيمن يُسمح له بجمع البيانات الشخصية، وما هي البيانات التي يتم جمعها، وكيف يتم استخدامها، ومع من تتم مشاركتها. بعبارة أخرى، يمكن لنظام إنترنت الأشياء أن يحقق سرية مثالية (عبر تشفير كل شيء) ولكنه قد ينتهك الخصوصية بشكل صارخ (عبر بيع بيانات سلوك المستخدمين، حتى لو كانت مجهولة الهوية، لشركات إعلانية دون موافقة صريحة). في عالم الأجهزة القابلة للارتداء التي تسجل مؤشراتنا الحيوية والمنازل الذكية التي تراقب عاداتنا اليومية، فإن حماية البيانات من القراءة غير المصرح بها (السرية) هي مجرد خطوة أولى، بينما يظل ضمان استخدام هذه البيانات بشكل أخلاقي وقانوني ومحدود (الخصوصية) هو التحدي الأعمق والأكثر تعقيداً. [12]

3- **سلامة البيانات(أو النزاهة) (Data Integrity)**: يُعنى بضمان دقة وموثوقية المعلومات طوال دورة حياتها، والتأكد من أنها لم تتعرض لأي تعديل أو تغيير أو حذف، سواء بشكل متعمد أو عن طريق الخطأ، أثناء نقلها أو تخزينها. في سياق إنترنت الأشياء، لا يقتصر هذا المبدأ على مجرد حماية البيانات، بل هو أساس الثقة في النظام بأكمله، لأن القرارات والإجراءات المادية تعتمد بشكل مباشر على صحة هذه البيانات. على سبيل المثال، يمكن أن يؤدي تغيير بسيط في قراءة مستشعر من طرف مهاجم (كهجوم الرجل في المنتصف)، مثل تعديل درجة حرارة في مصنع ذكي من خطرة إلى طبيعية، إلى منع اتخاذ إجراءات وقائية ضرورية والتسبب في أضرار فادحة. ولضمان سلامة البيانات، يتم استخدام تقنيات التشفير مثل دوال التجزئة (Hashing) ورموز مصادقة الرسائل(MACs) ، والتي تعمل بمثابة ختم رقمي للرسالة. عند استلام البيانات، يمكن للمستقبِل التحقق من هذا الختم للتأكد فوراً من أن البيانات التي وصلته هي النسخة الأصلية والدقيقة التي تم إرسالها من المصدر الموثوق، دون أي عبث أو تحريف. [12]

4- **المصادقة (Authentication)**: وهو يُعد حجر الزاوية الذي تُبنى عليه الثقة داخل أي منظومة لأمن إنترنت الأشياء، وهو يُعنى بالعملية الحاسمة للتحقق من هوية جميع الكيانات المشاركة في الاتصال والتأكد من أنها هي من تدعي أنها هي بالفعل. على عكس الأنظمة التقليدية التي تركز على مصادقة المستخدمين فقط، فإن المصادقة في بيئة IoT هي عملية متبادلة (Mutual Authentication) ومعقدة، حيث يجب على كل جهاز (كمستشعر أو كاميرا) أن يثبت هويته للبوابة أو للخادم السحابي، وفي المقابل، يجب على الخادم أن يثبت هويته للجهاز قبل أن يتم تبادل أي بيانات حساسة. إن غياب آلية مصادقة قوية يفتح الباب على مصراعيه أمام هجمات الانتحال والتزييف (Spoofing Attacks)، حيث يمكن لمهاجم أن ينشئ جهازاً خبيثاً ينتحل هوية جهاز شرعي ويرسل بيانات مزيفة إلى الشبكة، أو الأسوأ من ذلك، أن ينشئ خادماً مزيفاً لخداع الأجهزة الشرعية وجعلها ترسل بياناتها إليه مباشرة. ولتحقيق ذلك، يتم تزويد الأجهزة ببيانات اعتماد فريدة وآمنة، مثل الشهادات الرقمية (Digital Certificates) أو المفاتيح المشتركة مسبقاً(Pre-Shared Keys) ، والتي تستخدم لإثبات الهوية بشكل مشفر، مما يضمن أن جميع الاتصالات اللاحقة تتم بين أطراف موثوقة فقط. [12]

5- **التفويض (Authorization)**: وهو يُمثل الخطوة المنطقية التالية لعملية المصادقة، وهو يُعنى بتحديد وإنفاذ الصلاحيات الدقيقة الممنوحة لكل كيان مصادق (authenticated) عليه داخل شبكة إنترنت الأشياء. فبعد التحقق من هوية جهاز أو مستخدم، يجيب التفويض على سؤال: "ما الذي يُسمح لهذا الكيان بفعله؟". يعتمد هذا المبدأ بشكل أساسي على مبدأ الامتياز الأقل(Principle of Least Privilege) ، الذي ينص على أنه يجب منح كل كيان الحد الأدنى من الصلاحيات اللازمة لأداء وظيفته المحددة فقط لا غير. على سبيل المثال، يجب أن يكون مستشعر درجة الحرارة مصرّحاً له فقط بإرسال قراءات الحرارة إلى الخادم، ولكن لا ينبغي أبداً أن يكون لديه تفويض لإصدار أوامر لقفل ذكي أو الوصول إلى بيانات من أجهزة أخرى. وبالمثل، قد يُصرّح لمستخدم عادي بمشاهدة بث

كاميرا المراقبة، بينما يُحصر التفويض بتغيير إعداداتها أو حذف التسجيلات على حساب المسؤول فقط. إن التطبيق الصارم لسياسات التفويض يقلل بشكل كبير من سطح الهجوم ويحد من الأضرار المحتملة في حال تم اختراق أحد الأجهزة، حيث يمنع المهاجم من الانتقال أفقياً عبر الشبكة واستغلال صلاحيات غير ممنوحة له، مما يضمن أن كل مكون في النظام يعمل ضمن نطاقه الوظيفي المحدد بدقة. [12]

تحدثنا في هذا القسم عن أنواع الهجمات على شبكات إنترنت الأشياء وتأثيرها على الشبكة والمستخدم وخصوصيته. إن هذا التأثير أدى إلى الحاجة الماسة إلى أنظمة متطورة لتحذير المسؤول عن الشبكة عن وجود تهديدات للشبكة أو انتهاكات للسياسة الأمنية ليقوم بدوره باتخاذ التدابير المناسبة وفقاً للمعلومات المتاحة له. أحد أهم هذه الأنظمة هي أنظمة كشف التسلل أو IDS وأنظمة منع الاختراق IPS. حيث تعتبر أنظمة كشف التسلل تكنولوجيا سلبية تتلخص وظيفتها في الفحص والتنبيه والمراقبة ويكون الأمر متروك في النهاية للمسئولين إذا حدث خرق للشبكة، أما أنظمة منع الاختراق فهي تمنع التهديدات لحظة التعرف عليها وفقاً للبيانات المتاحة لها وذلك يتم بأكثر من طريقة ولكننا في مشروعنا هذا سنهتم بالنوع الأول من الأنظمة وهو أنظمة كشف التسلل.

نظام كشف التسلل (Intrusion Detection Systems)

يُعد نظام كشف التسلل (Intrusion Detection System - IDS) تقنية أمنية تعمل بمثابة جهاز إنذار للشبكات أو الأنظمة الحاسوبية. تتمثل وظيفته الأساسية في المراقبة السلبية لحركة مرور البيانات وأنشطة النظام، وتحليل هذه المعلومات بحثاً عن أي علامات تدل على أنشطة خبيثة أو انتهاكات للسياسة الأمنية، ومن ثم إصدار تنبيه عند اكتشاف تهديد محتمل.

إن الكلمة المفتاحية هنا هي مراقبة سلبية (Passive) فنظام كشف التسلل لا يقوم بحظر التهديد المكتشف بشكل نشط، بل يقتصر دوره على تحديد التهديد وإبلاغ المسؤول الأمني عنه، والذي يجب عليه بعد ذلك اتخاذ الإجراء المناسب .وهذا هو الفارق الجوهري بينه وبين نظام منع التسلل(IPS) ، المصمم لإيقاف التهديدات بشكل فوري ونشط.

المكونات الأساسية لنظام كشف التسلل

يتألف نظام كشف التسلل النموذجي من أربعة مكونات منطقية رئيسية تعمل معاً لاكتشاف التهديدات والإبلاغ عنها.

1- **المستشعرات أو الوكلاء(Sensors/Agents)** : وظيفتها الوحيدة هي جمع البيانات من البيئة التي تتم مراقبتها.ولها هدة أنواع حيث أنه في نظام كشف التسلل على مستوى الشبكة(NIDS) ، تكون المستشعرات عبارة عن أجهزة شبكية تلتقط جميع حزم البيانات التي تتدفق عبر قطاع معين من الشبكة. أما في نظام كشف التسلل على مستوى المضيف(HIDS) ، فتكون المستشعرات عبارة عن برامج وكيلة (Agents) مثبتة على الحواسيب الفردية، وتقوم بمراقبة سجلات النظام، وتغييرات الملفات، والعمليات قيد التشغيل.

2- **محرك التحليل(Analysis Engine)** : وهو يستقبل البيانات الأولية من المستشعرات ويقوم بتحليلها لتحديد ما إذا كان هناك تسلل قد حدث .يستخدم المحرك عادةً إحدى الطريقتين التاليتين أو كلتيهما من أجل الكشف:

- الكشف المعتمد على التوقيع(Signature-based Detection) : يقوم بمقارنة البيانات المجمّعة بقاعدة بيانات من أنماط الهجوم المعروفة (أو التواقيع). هذه الطريقة فعالة جداً في اكتشاف التهديدات المعروفة ولكنها لا تستطيع تحديد الهجمات الجديدة غير الموثقة.
- الكشف المعتمد على الشذوذ (Anomaly-based Detection): يقوم أولاً ببناء نموذج أساسي للسلوك الطبيعي للشبكة أو النظام. بعد ذلك، يراقب أي نشاط ينحرف بشكل كبير عن هذا الخط الأساسي، ويصنفه على أنه تهديد محتمل. يمكن لهذه الطريقة اكتشاف هجمات جديدة، ولكنها قد تولد المزيد من الإنذارات الكاذبة(False Positives) .

3- **نظام التنبيه وإعداد التقارير(Alerting / Reporting System)** : حيث أنه عندما يكتشف محرك التحليل تسللاً محتملاً، يكون هذا النظام مسؤولاً عن إصدار تنبيه لموظفي الأمن. ويمكنه إرسال التنبيهات بطرق مختلفة، مثل البريد الإلكتروني أو الرسائل القصيرة إلى المسؤول، أو إشعار في لوحة معلومات أمنية مثل(SIEM) ، أو تسجيل إدخال في ملف سجل(Log File)لمراجعته لاحقاً.

4- **وحدة التحكم الإدارية(Management Console)** : وهي تمثل لوحة التحكم المركزية أو واجهة المستخدم. تسمح للمسؤولين بتكوين نظام كشف التسلل، وإدارة المستشعرات، وتحديث قاعدة بيانات التواقيع الهجومية، وعرض وتحليل التنبيهات، وإنشاء تقارير حول الأحداث الأمنية.

[13]

أنواع نظام كشف التسلل

نظام كشف التسلل على مستوى الشبكة (NIDS)

يُعد نظام كشف التسلل على مستوى الشبكة (Network-based Intrusion Detection System - NIDS) مكوناً استراتيجياً في البنية الأمنية للشبكات، حيث يعمل كحارس يراقب حركة مرور البيانات بشكل شامل. يتم وضع مستشعرات هذا النظام في نقاط استراتيجية داخل الشبكة (مثل نقاط التوزيع الرئيسية أو خلف الجدران النارية)، حيث تقوم بالتقاط وتحليل نسخ من جميع حزم البيانات التي تمر عبرها في الوقت الفعلي. على عكس أنظمة HIDS التي تركز على جهاز واحد، فإن NIDSيمتلك رؤية واسعة للشبكة بأكملها، مما يمكنه من اكتشاف الهجمات التي تستهدف عدة أجهزة في وقت واحد، أو محاولات المسح والاستطلاع للشبكة، أو الهجمات التي تنتشر بين الأجهزة. يقوم محرك التحليل الخاص به بمقارنة حزم البيانات التي تم التقاطها مع قاعدة بيانات من تواقيع الهجمات المعروفة أو بمراقبة أي انحراف عن سلوك الشبكة الطبيعي. عند اكتشاف

نشاط مشبوه، يقوم النظام بإصدار تنبيه فوري للمسؤولين. ومع ذلك، فإن من أبرز تحدياته هو عدم قدرته على تحليل حركة المرور المشفرة، والتي تبدو له كبيانات عشوائية، مما قد يجعله أعمى أمام الهجمات التي تتم عبر قنوات اتصال آمنة. [14]

نظام كشف التسلل على مستوى المضيف (HIDS)

يُمثل نظام كشف التسلل على مستوى المضيف (Host-based Intrusion Detection System - HIDS) خط دفاع عميق ومخصص، حيث يعمل كوكيل أمني (Agent) يتم تثبيته مباشرة على نظام التشغيل للجهاز المراد حمايته، سواء كان خادماً أو حاسوباً شخصياً. على عكس أنظمة NIDS التي تراقب الشبكة بشكل عام، فإن HIDS يمتلك رؤية تفصيلية ودقيقة لما يحدث داخل الجهاز نفسه. يقوم بمراقبة مجموعة واسعة من الأنشطة الداخلية مثل التغييرات في ملفات النظام الحرجة، أو محاولات تعديل سجلات النظام(System Logs) ، أو العمليات (Processes) قيد التشغيل، أو استدعاءات النظام System) (Calls. تمنحه هذه الرؤية العميقة القدرة على اكتشاف الهجمات التي قد تفلت من مراقبة الشبكة، مثل البرمجيات الخبيثة التي تم تحميلها عبر وسائط مادية كـ (USB) أو الهجمات التي يستغل فيها مستخدم شرعي صلاحياته بشكل خبيث. عند اكتشاف نشاط مشبوه، يقوم النظام بمقارنته بقواعد بيانات معروفة أو بتحليل مدى انحرافه عن السلوك الطبيعي للجهاز، ومن ثم يصدر تنبيهاً. على الرغم من فعاليته، فإن من أبرز تحدياته هو استهلاكه لجزء من موارد الجهاز الذي يعمل عليه، والحاجة إلى تثبيته وإدارته بشكل فردي على كل مضيف في الشبكة. [14]

نظام كشف التسلل المعتمد على البروتوكول (PIDS)

يُمثل نظام كشف التسلل المعتمد على البروتوكول (Protocol-based Intrusion Detection System - PIDS) نوعاً متخصصاً من أنظمة كشف التسلل، يركز على مراقبة وتحليل بروتوكول اتصال معين بين كيانين، وعادة ما يكون بين العميل (Client)والخادم (Server) على عكس نظام NIDS الذي يراقب كل حركة المرور على الشبكة، أو نظام HIDS الذي يراقب الأنشطة داخل مضيف واحد، فإن PIDS يوضع عادة على الخادم لمراقبة وفك تشفير وفهم بروتوكول محدد بشكل عميق، وأشهر مثال على ذلك هو بروتوكول HTTP/HTTPSلخوادم الويب. يقوم هذا النظام بتحليل الطلبات والاستجابات ضمن هذا البروتوكول المحدد للكشف عن أي سلوك غير طبيعي أو استغلال للثغرات الخاصة بالبروتوكول نفسه، مثل هجمات حقن SQL أو Cross-Site Scripting (XSS) التي قد تكون مخفية داخل حركة مرور HTTPS المشفرة. وبهذا، فإنه يوفر طبقة حماية مركزة ومتعمقة لا تستطيع أنظمة NIDS توفيرها (بسبب التشفير) وتكون أكثر تخصصاً من أنظمة HIDS العامة. [14]

نظام كشف التسلل المستند إلى بروتوكول التطبيق (APIDS)

يُعد نظام كشف التسلل المستند إلى بروتوكول التطبيق (Application Protocol-based Intrusion Detection System - APIDS) أحد أكثر أنواع أنظمة كشف التسلل تخصصاً وعمقاً، حيث يركز بشكل حصري على مراقبة وتحليل حركة مرور البيانات الخاصة ببروتوكول تطبيق معين، مثل بروتوكول نقل النص الفائق (HTTP) لخوادم الويب، أو بروتوكول نقل الملفات (FTP). على عكس أنظمة NIDS التي تفحص حركة المرور بشكل عام، فإن APIDS يمتلك فهماً عميقاً لقواعد ومنطق البروتوكول الذي يراقبه، مما يمكنه من اكتشاف الهجمات المعقدة والمخفية داخل ما يبدو أنها اتصالات شرعية، كهجمات حقن لغة الاستعلامات المهيكلة (SQL Injection) أو البرمجة عبر المواقع(Cross-Site Scripting) ، والتي قد تمر دون أن يلاحظها نظام كشف التسلل الشبكي. يتم تثبيت هذا النظام عادةً على الخادم الذي يستضيف التطبيق مباشرةً، ليوفر طبقة حماية مركزة ودقيقة تستهدف نقاط الضعف الخاصة بالتطبيقات نفسها. [14]

نظام كشف التسلل الهجين (Hybrid IDS)

يُمثل نظام كشف التسلل الهجين (Hybrid Intrusion Detection System) نهجاً متطوراً ومتكاملاً يهدف إلى تحقيق أقصى درجات الدقة والشمولية في اكتشاف التهديدات، وذلك عبر دمج نوعين أو أكثر من تقنيات كشف التسلل المختلفة للاستفادة من نقاط القوة لكل منها وتعويض نقاط ضعف الأخرى. إن الشكل الأكثر شيوعاً لهذا النظام هو الذي يجمع بين الكشف المعتمد على التوقيع (Signature-based Detection) والكشف المعتمد على الشذوذ .(Anomaly-based Detection) في هذا النموذج، يقوم النظام باستخدام قاعدة بيانات التواقيع لاكتشاف الهجمات المعروفة والموثقة بدقة عالية، وفي الوقت نفسه، يستخدم تحليل الشذوذ لتحديد الهجمات الجديدة وغير المعروفة (Zero-Day Attacks) عبر مراقبة أي انحراف عن السلوك الطبيعي للشبكة. كما يمكن أن يشير المصطلح أيضاً إلى نظام يدمج بيانات من مصادر مختلفة، مثل دمج معلومات من نظام كشف التسلل الشبكي (NIDS) مع معلومات من نظام كشف التسلل على مستوى المضيف(HIDS) ، مما يوفر رؤية أمنية شاملة تربط بين الأنشطة المشبوهة على الشبكة وتأثيرها الفعلي داخل الأجهزة، وبالتالي تقليل الإنذارات الكاذبة (False Positives)وزيادة فعالية الاستجابة. [14]

طرق الكشف في أنظمة كشف التسلل (Detection Methods)

الكشف المعتمد على التوقيع (Signature-Based Detection)

تُعد هذه الطريقة المنهجية التقليدية في أنظمة كشف التسلل، وهي تعمل بشكل مشابه لبرامج مكافحة الفيروسات من خلال الاعتماد على قاعدة بيانات ضخمة من التواقيع التي تمثل بصمات رقمية لهجمات وبرمجيات خبيثة معروفة مسبقاً. يقوم النظام

بمطابقة حركة مرور البيانات والأنشطة المراقبة بشكل مستمر مع هذه القاعدة، وفي حال حدوث تطابق مع أي توقيع، يتم إطلاق تنبيه فوري. وتمتاز هذه الطريقة بدقتها العالية ومعدل إنذاراتها الكاذبة المنخفض للغاية عند التعامل مع الهجمات الموثقة مسبقاً، مما يجعلها أداة موثوقة لتحديد التهديدات المعروفة. إلا أن نقطة ضعفها الجوهرية تكمن في عجزها التام عن مواجهة الهجمات الجديدة أو ما يعرف بهجمات اليوم صفر(Zero-Day Attacks) ، حيث أنها لا تستطيع التعرف إلا على ما هو موجود بالفعل في قاعدة بياناتها، مما يجعلها تتطلب تحديثاً مستمراً لتبقى فعالة. [14]

الكشف المعتمد على الشذوذ (Anomaly-Based Detection)

ظهرت هذه الطريقة كحل للتغلب على قصور الطريقة المعتمدة على التوقيع، حيث تركز على تحديد أي سلوك ينحرف عن الوضع الطبيعي بدلاً من البحث عن تهديد معروف. تعمل هذه الطريقة عبر مرحلتين: أولاً، مرحلة التعلم التي تستخدم فيها تقنيات التعلم الآلي (Machine Learning) لإنشاء نموذج إحصائي أو خط أساس (Baseline) يمثل السلوك الطبيعي والمشروع للشبكة أو النظام. ثانياً، مرحلة الكشف التي تتم فيها مقارنة الأنشطة الحية مع هذا النموذج، وأي انحراف كبير يُعتبر شذوذاً وهجوماً محتملاً. وتكمن القوة الأساسية لهذا المنهج في قدرته على اكتشاف التهديدات غير المعروفة وهجمات اليوم صفر التي لا تمتلك تواقيع، مما يوفر قدرة دفاعية استباقية. ولكن، يواجه هذا المنهج تحدياً رئيسياً يتمثل في احتمالية توليد عدد أكبر من الإنذارات الكاذبة(False Positives) ، حيث أن أي نشاط شرعي ولكنه غير معتاد قد يتم تصنيفه بالخطأ على أنه تهديد، كما أن فعاليته تعتمد بشكل كبير على وجود بيئة نظيفة أثناء مرحلة التعلم. [12]

بعد الحديث عن بنية انترنت الاشياء وتأثير الهمات عليها وأنظمة كشف التسلل، سنتحدث في الفصل التالي عن تعلم الالة والخوارزميات المتبعة في مشروعنا.

الفصل الثالث

تعلّم الآلة والتعلم العميق

في هذا الفصل سنعرض الدراسة التي أجريناها في مجال تعلم الآلة والتعلم العميق وذكر المصنفات.

المقدمة

إن تطور التهديدات السيبرانية من هجمات بسيطة ومتوقعة إلى استغلالات معقدة ومتعددة الأشكال وغير مرئية من قبل، قد كشف عن القيود الجوهرية الكامنة في الأنظمة الأمنية التقليدية .هذه الأنظمة القديمة، وتحديداً أنظمة كشف التسلل المعتمدة على التوقيع(Signature-based IDS) ، تعمل وفق نموذج رد فعل(Reactive Model) ، حيث تقوم بتحديد التهديدات عبر مطابقة حركة مرور الشبكة مع قاعدة بيانات محددة مسبقاً لأنماط الهجوم المعروفة. وعلى الرغم من فعاليتها ضد البرمجيات الخبيثة الموثقة، إلا أن هذا النهج يظل عاجزاً بشكل أساسي عن اكتشاف الهجمات الجديدة أو ما يعرف بـ هجمات اليوم صفر(Zero-Day Attacks) ، مما يجعله متأخراً بخطوة دائمة عن الخصوم المتطورين. هذا الموقف التفاعلي استلزم تحولاً نموذجياً نحو نموذج أمني استباقي قائم على البيانات(Data-Driven Security) . فالحجم الهائل للبيانات المتدفقة عبر الشبكات الحديثة، الذي كان يُنظر إليه في السابق على أنه تحدي، يُعتبر الآن مورداً غنياً لتدريب الأنظمة الذكية. وهذا هو السبب تحديداً الذي جعل تعلم الآلة (Machine Learning) حجر الزاوية في الأمن السيبراني الحديث، فبدلاً من الاعتماد على قواعد ثابتة، فإنه يستفيد من الخوارزميات التي يمكنها تعلم الأنماط المعقدة للسلوك الطبيعي للشبكة من البيانات نفسها. ومن خلال إنشاء هذا الخط الأساسي للسلوك الطبيعي، يمكن لنماذج تعلم الآلة تحديد الانحرافات والشذوذات الدقيقة التي تشير إلى تسلل محتمل، حتى لو لم يتم رؤيته من قبل. وعليه، يهدف هذا الفصل إلى تقديم استكشاف نظري للنماذج المتقدمة التي تقف في طليعة هذا التحول، حيث يتم فحص المبادئ التأسيسية لأساليب التجميع مثل الغابة العشوائية(Random Forest) ، وتقنيات التعزيز المتدرج مثل XGBoost وLightGBM ، وقدرات تعلم السمات المعقدة للشبكات العصبية(Neural Networks) ، كل ذلك في سياق بناء دفاعات أمنية أكثر تكيفاً وذكاءً.

المفاهيم الأساسية في تعلم الآلة

ينقسم مجال تعلم الآلة بشكل أساسي إلى نموذجين أساسيين: التعلم الخاضع للإشراف (Supervised Learning) والتعلم غير الخاضع للإشراف (Unsupervised Learning).

في **التعلم الخاضع للإشراف (Supervised Learning)**، تتعلم الخوارزمية من مجموعة بيانات تم تصنيفها مسبقاً بالنتائج الصحيحة. حيث يتم تزويد النموذج ببيانات الإدخال مع الإجابات الصحيحة المقابلة لها، وتتعلم كيفية ربط أحدهما بالآخر. غالباً ما يكون هدفه الأساسي هو التصنيف (Classification)، مثل تحديد حركة مرور الشبكة إما كـ "خبيثة" أو "سليمة". على العكس من ذلك تماماً، يعمل **التعلم غير الخاضع للإشراف (Unsupervised learning)** على بيانات غير مصنفة (unlabeled)، حيث يجب على الخوارزمية اكتشاف الأنماط والهياكل المخفية بنفسها. إن التطبيق الأكثر صلة بهذا النموذج في مجال الأمن هو كشف الشذوذ (Anomaly Detection)، حيث يكون الهدف هو تحديد نقاط البيانات النادرة أو المشبوهة التي تنحرف بشكل كبير عن القاعدة المعمول بها، مما يجعله ذا قيمة لا تقدر بثمن في اكتشاف التهديدات الجديدة. كلا النموذجين حاسمان لاستراتيجية أمنية شاملة: فالتعلم الخاضع للإشراف يتفوق في تحديد الهجمات المعروفة بدقة عالية، بينما يوفر التعلم غير الخاضع للإشراف القدرة على اكتشاف التهديدات الجديدة وغير المتوقعة. [16] [15]

يتبع التطبيق العملي لأي نموذج من نماذج تعلم الآلة عملية منظمة ومتعددة المراحل تُعرف بمسار عمل تعلم الآلة (Machine Learning Workflow). يضمن هذا المسار النظري بناء النماذج وتقييمها بطريقة قوية وقابلة للتكرار. تبدأ العملية بجمع البيانات (Data Collection)، حيث يتم تجميع البيانات الأولية، مثل سجلات الشبكة أو أحداث النظام. نادراً ما تكون هذه البيانات قابلة للاستخدام في حالتها الأولية ويجب أن تخضع للمعالجة المسبقة للبيانات (Data Preprocessing)، وهي خطوة حاسمة تتضمن تنظيف البيانات(dataset cleaning)، والتعامل مع القيم المفقودة(handling missing values)، وتطبيع السمات إلى مقياس مشترك(feature scaling). بعد ذلك، يتم تنفيذ هندسة السمات (Feature Engineering) لاختيار أو إنشاء المتغيرات الأكثر إفادة والتي ستساعد النموذج على إجراء تنبؤات دقيقة. يكمن جوهر مسار العمل في مرحلة تدريب النموذج (Model Training)، حيث يتم ادخال البيانات المعدة (مجموعة التدريب) لخوارزمية (مثل الغابة العشوائية أو الشبكة العصبية) لتعلم الأنماط الأساسية. وبشكل حاسم، يتم بعد ذلك التحقق من أداء النموذج في مرحلة تقييم النموذج (Model Evaluation)، حيث يتم اختباره على مجموعة بيانات منفصلة وغير مرئية مسبقاً (مجموعة الاختبار) لضمان قدرته على التعميم على بيانات جديدة. أخيراً، يكون النموذج الذي تم التحقق من صحته جاهزاً للنشر (Deployment) في بيئة حية. [18] [17]

يتطلب الفهم الواضح لتعلم الآلة الإلمام بمصطلحاته الأساسية. تُعرف متغيرات الإدخال التي يستخدمها النموذج بالسمات (Features)، وهي الخصائص الفردية القابلة للقياس للبيانات التي يتم تحليلها على سبيل المثال، حجم حزمة الشبكة، أو مدة الاتصال. في التعلم الخاضع للإشراف، تقابل كل مجموعة من السمات عنواناً أو تصنيفاً (Label)، وهو النتيجة أو التصنيف

الذي نحاول التنبؤ به، مثل فئة "هجوم" أو "طبيعي". تسمى مهمة التنبؤ بهذه الفئات المحددة مسبقاً بالتصنيف (Classification). وعلى العكس من ذلك، فإن كشف الشذوذ (Anomaly Detection)، الذي غالباً ما يكون مهمة غير خاضعة للإشراف، لا يعتمد على الlabels. بدلاً من ذلك، يركز على تحديد نقاط البيانات النادرة أو التي تختلف اختلافاً كبيراً عن غالبية البيانات. في سياق أمني، يُستخدم التصنيف لتصنيف التهديدات المعروفة بناءً على سماتها، بينما يعد كشف الشذوذ التقنية الأساسية المستخدمة للإبلاغ عن الأنشطة الجديدة والمشبوهة التي لم يتم رؤيتها من قبل. [18] [19]

التعلم التجميعي (Ensemble Learning)

تُعد أساليب التجميع (Ensemble Learning) تقنية قوية تعمل على مبدأ أن دمج تنبؤات نماذج فردية متعددة، تُعرف بـ النماذج الضعيفة (weak learners)، يؤدي إلى تنبؤ واحد أكثر دقة ومتانة مما يمكن لأي نموذج فردي تحقيقه بمفرده. تماثل الفكرة الجوهرية طلب المشورة من مجموعة متنوعة من الخبراء بدلاً من الاعتماد على خبير واحد؛ حيث تميل حكمة الجمهور الجماعية إلى أن تكون أفضل من الآراء الفردية. هناك عائلتان رئيسيتان من أساليب التجميع: تقنيات التكييس (Bagging)، مثل الغابة العشوائية (Random Forest)، والتي تقوم بتدريب نماذج متعددة بالتوازي على مجموعات فرعية مختلفة من البيانات ومن ثم حساب متوسط تنبؤاتها بهدف تقليل التباين (reduce variance)؛ وتقنيات التعزيز (Boosting)، مثل XGBoost، والتي تقوم بتدريب النماذج بشكل تسلسلي، حيث يركز كل نموذج جديد على تصحيح الأخطاء التي ارتكبها النموذج السابق بهدف تقليل التحيز (reduce bias). من خلال تجميع وجهات نظر متعددة، تكون أساليب التجميع فعالة للغاية في تحسين الأداء التنبؤي وتعتبر حجر الزاوية في تعلم الآلة الحديث.[16]

مصنف الغابة العشوائية (Random Forest Classifier)

يُعد مصنّف الغابة العشوائية (Random Forest) طريقة تعلّم تجميعي (Ensemble Learning) قوية وشائعة الاستخدام، حيث أنها تعمل من خلال بناء عدد كبير من أشجار القرار (Decision Trees) في مرحلة التدريب لمعالجة مشكلة الارتباط الكبير ببيانات التدريب (Overfitting)التي تعاني منها أشجار القرار الفردية(Decision tree classifier) .

ينتمي هذا المصنف إلى فئة أوسع من الخوارزميات تُعرف باسم Bootstrap Aggregating أو Bagging ، ولكن مع تحسين جوهري يعزز أداءه بشكل كبير. يعتمد المبدأ الأساسي للغابة العشوائية على إدخال مصدرين مختلفين من العشوائية لتوليد مجموعة من الأشجار غير المترابطة (Decorrelated).

أولاً، يتم تدريب كل شجرة قرار فردية على عينة فرعية عشوائية مختلفة من بيانات التدريب، يتم سحبها مع الاستبدال وهو ما يُعرف بالعينة الاستنساخية أو (Bootstrap Sample)، مما يضمن أن كل شجرة قرار تتعلم من عينة من لبيانات.

ثانياً، أيضا بمصنف الغابات العشوائية يوجد عشوائية حتى على عملية اختيار السمات (features) عند كل انقسام للعقدة (Node Split). فبدلاً من البحث عن أفضل انقسام بين جميع الميزات (الfeatures) المتاحين، تختار الخوارزمية مجموعة فرعية عشوائية من الميزات وتعتبرها فقط لتقسيم العقدة.

ومن أجل اعطاء التصنيف النهائي، يتم من خلال دمج تنبؤات جميع الأشجار الفردية – عادةً عبر تصويت الأغلبية (Majority Vote)في مهام التصنيف – يقوم النموذج النهائي بتجميع هذه النماذج المدرّبة الأساسية المتنوعة، ذات الانحياز المنخفض (low bias) والتباين المرتفع (high variance)، في نموذج واحد قوي (Robust) يتميز بانحياز منخفض وتباين منخفض بشكل ملحوظ. هذه الاستراتيجية المزدوجة للعشوائية (للعينات وللسمات) تفصل الارتباط بين الأشجار بفعالية، مما يجعل النموذج التجميعي أقل عرضة للoverfitting وأكثر قدرة على التعميم (generalization)على البيانات الجديدة التي لم يرها من قبل أي (الunseen data). كما في الشكل التالي:

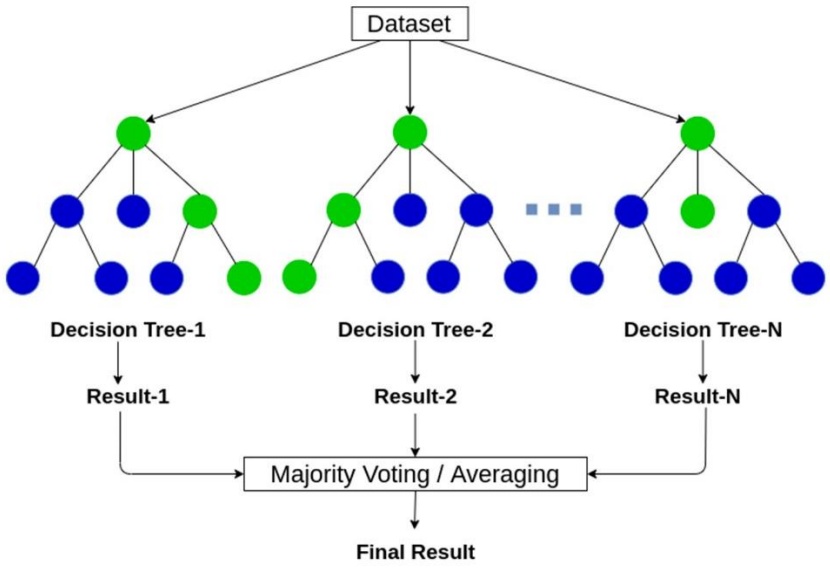


Figure6 Random Forest Classifier

يالاضافة الى ذلك، توفر الخوارزمية نواتج ثانوية قيمة، مثل مقياس لأهمية الميزات(Feature Importance) ، والذي يتم حسابه من خلال ملاحظة مقدار الانخفاض في دقة النموذج عند تبديل قيم ميزة معينة بشكل عشوائي عبر العينات خارج الكيس (Out-of-Bag Samples).

[20]

Random forest Hyperparameters

يعتمد الأداء التنبؤي لنموذج الغابة العشوائية بشكل كبير على إعدادات وسطائه الفائقة (Hyperparameters)، وهي وسطاء يتم تحديدها قبل البدء بعملية التدريب. يعد الضبط الصحيح لهذه المعلمات أمر ضروري للتحكم في مدى تعقيد النموذج، وإدارة المفاضلة بين الانحياز والتباين (Bias-Variance Tradeoff)، وفي النهاية تحسين قدرته على التعميم على البيانات الجديدة (Generalization to Unseen Data). ومنهم:

- n_estimators: تحدد هذه المعلمة العدد الإجمالي لأشجار القرار التي سيتم إنشاؤها في الغابة. بشكل عام، يؤدي استخدام عدد أكبر من الأشجار إلى نموذج أكثر قوة واستقرار، حيث إن تجميع تنبؤات عدد أكبر من الأشجار غير المترابطة يقلل من تباين النموذج الكلي. ومع ذلك، فإن الأداء عادةً ما يظهر تناقصاً بعد عدد معين، بينما تزداد التكلفة الحسابية للتدريب والاختبار بشكل خطي.
- max_features: يتحكم هذا الوسيط في حجم المجموعة الفرعية العشوائية من الميزات التي يتم النظر فيها عند كل انقسام للعقدة داخل الشجرة. تعد من أهم المعلمات لضبط تباين النموذج. إن استخدام قيمة أصغر للمعلمة max_features يزيد من عشوائية كل شجرة، مما يقلل بدوره من الارتباط بين الأشجار في الغابة، ويؤدي إلى انخفاض أكبر في تباين النموذج الإجمالي. أما القيمة الأكبر فتجعل الأشجار الفردية أكثر تشابهاً، حيث يزداد احتمال اختيار الأشجار لنفس الميزات للتقسيم. غالباً يتم استخدام sqrt(# of all features) لمهام التصنيف.
- max_depth: يحدد هذا الوسيط أقصى عمق يمكن أن تنمو إليه كل شجرة قرار فردية(Decision tree). وهي بمثابة طريقة مباشرة للتحكم في مدى تعقيد الوسطاء الأساسية. إذا لم يتم تحديدها، تستمر الأشجار في النمو حتى تصبح جميع العقد الطرفية (Leaf Nodes) نقية أو تحتوي على عدد عينات أقل من min_samples_split. يساعد تقييد العمق على تنظيم النموذج (Regularize)؛ فالأشجار الأقل عمقاً تكون أقل تعقيداً وأقل عرضة للoverfitting لأنها تمنع الانحياز للضجيج أو للنقاط الشاذة الخاصة بعينة التدريب.
- min_samples_split: يحدد هذا الوسيط الحد الأدنى لعدد العينات التي يجب أن تحتويها العقدة حتى يتم النظر في تقسيمها. من خلال تعيين هذه القيمة إلى رقم أكبر من القيمة الافتراضية، يمكن منع النموذج من إنشاء انقسامات بناءً على عدد قليل جداً من العينات التي تؤدي الى overfitting، ومنعه من زيادة تعقيده كثيرا.
- min_samples_leaf: يحدد الحد الأدنى لعدد العينات التي يجب أن تكون موجودة في العقدة الورقة (Leaf Node). لا يعتبر الانقسام صالحاً إلا إذا نتج عنه عقد تحتوي كل منها على هذا العدد من العينات أو اكثر. وبالتالي هي أيضا مهمة لمنع حدوث الoverfitting لأنه سيصبح أي تنبؤ فردي مدعوم بمجموعة كبيرة من العينات مما يجعل تنبؤات النموذج أكثرصحة.
- criterion: يحدد الدالة أو التابع المستخدم لقياس جودة الانقسام. بالنسبة لمهام التصنيف، الخياران الأساسيان هما 'gini' لـ (Gini Impurity) و 'entropy' لـ (Information Gain). تقيس gini impurity احتمالية التصنيف

الخاطئ لعنصر تم اختياره عشوائياً إذا تم تصنيفه وفقاً لتوزيع الفئات في المجموعة الفرعية. أما information gain فهو مشتق من مفهوم الإنتروبي في نظرية المعلومات. على الرغم من أن كلا المعيارين يخدمان الهدف نفسه، ولكن غالباً ما يكون gini function هو الخيار الافتراضي لأنه أسرع قليلاً في الحساب.

- (class weight): صُمم هذا الوسيط لمعالجة مشكلة مجموعات البيانات غير المتوازنة، وهي حالة شائعة يتجاوز فيها عدد عينات إحدى الصفوق عدد عينات الصفوف الاخرى بشكل كبير. من خلال تحديد وزن للصفوف، تقوم الخوارزمية بتعديل دالة الخسارة (Loss Function) لإعطاء أهمية أكبر للفئة ذات التمثيل الأقل (فئة الأقلية). هذا يعني أن التصنيف الخاطئ لعينة من فئة الأقلية سيتسبب في عقوبة أكبر أثناء التدريب، مما يجبر النموذج على تعلم ميزاتها بشكل أعمق بدلاً من الانحياز للفئة ذات الأغلبية لمجرد تحقيق دقة ظاهرية عالية.
- (n_jobs)عدد المهام المتوازية :هو وسيط يتعلق بالأداء الحسابي ولا يؤثر على النتائج التنبؤية النهائية للنموذج، ولكنها تتحكم في سرعة التدريب. تحدد هذه المعلمة عدد أنوية المعالج (CPU cores) التي يمكن للخوارزمية استخدامها بشكل متزامن لبناء الأشجار في الغابة. ونظراً لأن كل شجرة يتم تدريبها بشكل مستقل، فإن هذه المهمة قابلة للموازاة (Parallelizable) بدرجة عالية. إن ضبط هذه المعلمة لاستخدام أنوية متعددة يمكن أن يؤدي إلى تقليل كبير في الوقت اللازم لتدريب النموذج، خاصة مع مجموعات البيانات الكبيرة.
- (random_state) حالة العشوائية :تعد هذه المعلمة أساسية لضمان قابلية تكرار النتائج .(Reproducibility) تعتمد خوارزمية الغابة العشوائية على عدة عمليات عشوائية، مثل إنشاء عينات استنساخية (Bootstrap Samples) للبيانات واختيار مجموعات فرعية عشوائية من الميزات. من خلال تعيين random_stateإلى قيمة عددية ثابتة، يتم توفير بذرة (Seed) لمولد الأرقام العشوائية. هذا يضمن استخدام نفس تسلسل الأرقام العشوائية في كل مرة يتم فيها تدريب النموذج، مما ينتج عنه نموذج متطابق بأداء متطابق. وهذا أمر حاسم لتصحيح الأخطاء، ومقارنة النماذج المختلفة بشكل عادل، والسماح للباحثين الآخرين بالتحقق من صحة النتائج.

[17] [20]

مصنف XGBoost

يُعد مصنف XGBoost الذي هو اختصار ل (eXtreme Gradient Boosting)، مصنف عالي التطور ومُحسَّناً لإطار عمل التعزيز المتدرج (Gradient Boosting). تم تطويره بواسطة Tianqi Chenو Carlos Guestrin ، وأصبح خوارزمية أساسية للتعامل مع البيانات المهيكلة أو الجدولية(Tabular data)، حيث يشتهر بأدائه الاستثنائي في كل من الأوساط الأكاديمية والمسابقات التنافسية في مجال تعلم الآلة (machine learning). يُعتبر XGBoost طريقة تعلم تجميعي (Ensemble

Learning) حيث يبني سلسلة من أشجار القرار بشكل تسلسلي، حيث يتم تدريب كل شجرة جديدة على تصحيح الأخطاء التي ارتكبتها الأشجار السابقة.

المبدأ الأساسي لـ XGBoost هو في تقنية التعزيز المتدرج. فعلى عكس خوارزمية الغابة العشوائية التي تبني أشجاراً مستقلة بشكل متوازٍ (Bagging)، فإن التعزيز (Boosting) هو عملية تجميعية إضافية (Additive Process). تبدأ الخوارزمية بتدريب مُصنف ضعيف أولي (شجرة قرار بسيطة decision tree)، ثم تقوم بحساب الأخطاء، أو ما يُعرف بالبواقي (Residuals)، بين تنبؤات هذه الشجرة والقيم الحقيقية. لا يتم تدريب الشجرة التالية على التصنيفات الأصلية، بل على هذه البواقي. من خلال إضافة تنبؤات الشجرة الجديدة إلى تنبؤات الشجرة السابقة (مع ضربها في معدل التعلم أو Learning Rate)، يحسن النموذج دقته تدريجياً. يتم حساب هذه العملية على أنها مسألة أمثلية (Optimization Problem) يستخدم فيها النموذج خوارزمية الانحدار التدريجي (Gradient Descent) لتقليل دالة خسارة (Loss Function) محددة مع كل شجرة جديدة تُضاف إلى النموذج التجميعي. كما في الشكل التالي:

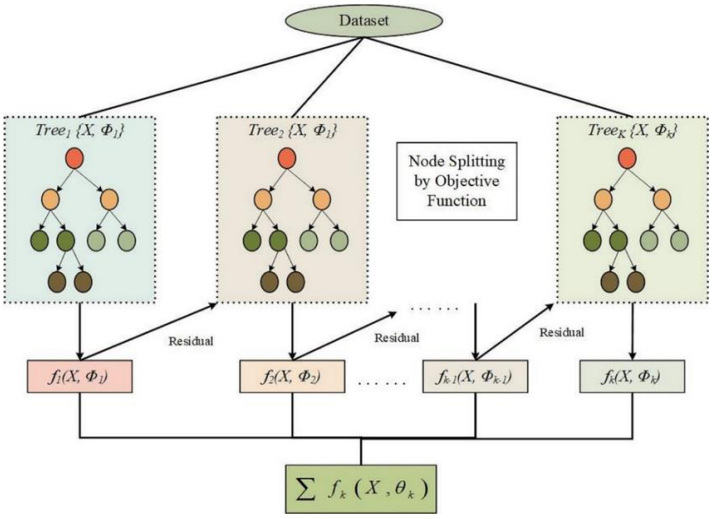


Figure7 XGBosot Algorithm

ان الذي يجعل مصنف XGBoost شديد (eXtreme) هو العديد من التحسينات الخوارزمية والتحسينات على مستوى النظام:

1. هدف تعلم مُنظَّم (Regularized Learning Objective): إن نماذج التعزيز المتدرج gradient boosting القياسية عرضة للارتباط الكبير ببيانات التدريب (Overfitting). يعالج مصنف XGBoost هذه المشكلة مباشرةً عن طريق تضمين حدي التنظيم L1 (Lasso) و L2 (Ridge) في دالة الهدف الخاصة به. تفرض هذه العملية عقوبة على مدى تعقيد النموذج (مثل عدد الأوراق الطرفية وحجم نتائجها)، مما يساعد على منع الoverfitting وتحسين قدرة النموذج على التعميم على البيانات الجديدة(generalization to unseen data).

2. إيجاد الانقسام مع مراعاة البيانات المتفرقة (Sparsity-Aware): في البيانات الواقعية، تعد القيم المفقودة شائعة. بدلاً من معالجتها المسبقة (preprocessing data)، يمتلك XGBoost حل مدمج للتعامل معها. عند كل عقدة في الشجرة، تتعلم الخوارزمية اتجاها افتراضيا للعينات ذات القيم المفقودة، مما يمكنها من تعلم أفضل استراتيجية لتعويض القيم المفقودة من البيانات نفسها بفعالية.
3. تحسينات النظام من أجل قابلية التوسع (Scalability): صُمم XGBoost ليكون فعالاً وقابلاً للتوسع. فهو يستخدم خوارزمية مبتكرة لإيجاد الأشجار تستخدم بنية بيانات تسمى الكتلة (Block) لتخزين البيانات في تنسيق عمودي مضغوط. هذه البنية مع أنماط الوصول الواعية لذاكرة التخزين المؤقت (Cache-aware)، تسمح بالحساب المتوازي والفعال لانقسامات الميزات(features splitting) أثناء بناء الشجرة.
4. التقريب من الدرجة الثانية (Second-Order Approximation): لتحسين دالة الخسارة، يستخدم XGBoost مفكوك تايلور من الدرجة الثانية second-order Taylor expansion، والذي يدمج معلومات من كل من المشتقة الأولى (التدرج أو Gradient) والمشتقة الثانية (Hessian). يوفر هذا معلومات أكثر حول اتجاه الأمثلية ويمكن أن يؤدي إلى تقارب أسرع.

[21]

XGBoost Hyperparameters

تكمن قوة خوارزمية XGBoost في مرونتها، والتي يتم التحكم فيها من خلال مجموعة واسعة من المعلمات الفائقة (Hyperparameters). يعد الضبط الصحيح لهذه المعلمات أمر حاسم لتحقيق التوازن بين الانحيازbias والتباينvariance (Bias-Variance Tradeoff)، والتحكم في مدى تعقيد النموذج، والوصول إلى الأداء الأمثل. يمكن تصنيف هذه المعلمات(الhyperparameters) بشكل عام إلى وسطاء تتحكم في عملية التعزيز (Boosting) الكلية، ووسطاء تتحكم في بنية الأشجار الفردية(decision trees).

Boosting hyperparameters:

- n_estimators: تحدد هذه المعلمة العدد الإجمالي للأشجار التسلسلية التي سيتم بناؤها، وهي تعادل عدد جولات التعزيز. يمكن أن يؤدي استخدام عدد أكبر من المقدِّرات (Estimators) إلى أداء أفضل، ولكن فقط إلى حد معين، قد يبدأ النموذج بعده في المعاناة من فرط التخصيص (الOverfitting). غالباً ما يتم ضبط هذه المعلمة بالتزامن مع معدل التعلم، وعادةً ما يتم إيجاد قيمتها المثلى باستخدام التوقف المبكر (Early Stopping) على مجموعة بيانات التحقق.
- learning_rate: تقوم هذه المعلمة بتقليص حجم مساهمة كل شجرة جديدة في التنبؤ النهائي للنموذج التجميعي. إن استخدام معدل تعلم أصغر يقلل من تأثير كل شجرة على حدة، مما يتطلب عدد أكبر من جولات التعزيز

(n_estimators) لبناء نموذج قوي. هذه العملية، التي تُعرف بـ "الانكماش" (Shrinkage)، تجعل عملية التعلم أكثر تحفظاً، مما يساعد على منع الoverfitting. تتراوح القيم لهذا الوسيط بين 0.01 و 0.3.

Decision trees hyperparameters:

- max_depth: تحدد هذه المعلمة أقصى عمق يمكن أن تصل إليه شجرة القرار الفردية. يمكن للأشجار الأكثر عمقاً التقاط أنماط أكثر تعقيداً وتحديداً في البيانات، ولكنها أيضاً أكثر عرضة لفرط التخصيص(overfitting). يعد تقييد العمق أحد أكثر الطرق شيوعاً للتحكم في مدى تعقيد النموذج ومنع حدوث الoverfitting.
- min_child_weight: تحدد هذه المعلمة الحد الأدنى لمجموع أوزان العينات (Hessian) المطلوب في العقدة الابن. إذا أدت خطوة تقسيم الشجرة إلى عقدة طرفية ذات مجموع أوزان أقل من min_child_weight، فإن عملية البناء ستتوقف عن المزيد من التقسيم. بعبارة أبسط، هي تتحكم في الحد الأدنى لعدد العينات المطلوبة في العقدة الطرفية، وتعتبر معلمة تنظيم (Regularization) قوية تُستخدم لمنع فرط التخصيص على مجموعات صغيرة ومحددة من العينات.
- gamma (أو min_split_loss): تحدد هذه المعلمة الحد الأدنى للانخفاض في الخسارة المطلوب لإجراء تقسيم إضافي على عقدة طرفية. لن يتم إجراء الانقسام إلا إذا أدى إلى انخفاض إيجابي في دالة الخسارة أكبر من قيمة gamma. إن استخدام قيمة gamma أعلى يجعل الخوارزمية أكثر تحفظاً، مما يؤدي إلى عدد أقل من الانقسامات وأشجار أبسط، وبالتالي ممكن أن تؤدي الى عدم حدوث overfitting ولكن بالطبع اختبار قيمة كبيره جدا ستؤدي الى underfitting.
- subsample: تتحكم هذه المعلمة في جزء بيانات التدريب (الصفوف) الذي يتم أخذ عينات منه عشوائياً قبل بناء كل شجرة جديدة. يؤدي تعيينها إلى قيمة أقل من 1.0 إلى إدخال العشوائية في عملية التعزيز، مما يساعد على منع فرط التخصيص من خلال ضمان بناء كل شجرة على مجموعة فرعية مختلفة قليلاً من البيانات.
- colsample_bytree: تشبه subsample، ولكنها خاصة بالميزات (الأعمدة). تحدد هذه المعلمة جزء الميزات الذي سيتم أخذ عينات منه عشوائياً عند بناء كل شجرة. تعد هذه طريقة فعالة أخرى لمنع فرط التخصيص ويمكنها أيضاً تسريع عملية التدريب بسبب عدم أخذ جميع السمات في كل مرة تدريب.

معلمات التنظيم (Regularization Parameters)

- reg_alpha (تنظيم L1) و reg_lambda (تنظيم L2): تتوافق هاتان المعلمتان مع حدي التنظيم L1 (Lasso) و L2 (Ridge) على أوزان العقد الطرفية. تؤدي زيادة هذه القيم إلى جعل النموذج أكثر تحفظاً عن طريق فرض عقوبة على قيم الأوزان الكبيرة، مما يساعد على جعل التنبؤات النهائية أكثر سلاسة ومنع فرط التخصيص.

[21]

LightGBM Classifier

LightGBM أو(Light Gradient Boosting Machine) ، هو إطار عمل للتعزيز المتدرج (Gradient Boosting) مفتوح المصدر وعالي الأداء، تم تطويره بواسطة شركة مايكروسوفت (Microsoft company). على الرغم من أنه يعتمد على نفس المبادئ التسلسلية لتصحيح الأخطاء الموجودة في طرق التعزيز المتدرج الأخرى مثلXGBoost ، إلا أن LightGBM قد تم تصميمه خصيصاً لمعالجة اختناقات الأداء في تلك الخوارزميات، مع إعطاء الأولوية لسرعة التدريب وكفاءة استخدام الذاكرة دون انخفاض الدقة بشكل كبير. لتحقيق ذلك، يقدم LightGBM العديد من التقنيات المبتكرة التي تغير بشكل أساسي كيفية بناء أشجار القرار.

أهم ابتكارين في هذه الخوارزمية هما تقنية أخذ العينات أحادية الجانب القائمة على التدرج-GOSS Gradient-based One) (Side Samplingوتقنية تجميع الميزات الحصرية (EFB Exclusive Feature Bundling) فمن خلال تقنيةGOSS ، تقوم الخوارزمية بأخذ عينات من البيانات المستخدمة لبناء كل شجرة بذكاء؛ فبدلاً من استخدام جميع نقاط البيانات، تحتفظ الخوارزمية بجميع العينات ذات التدرجات الكبيرة (أي تلك التي لم يتم تدريبها بشكل جيد) وتقوم بأخذ عينات عشوائية من العينات ذات التدرجات الصغيرة. هذا النهج يركز عملية التدريب على الأخطاء الأكثر إفادة، مما يقلل بشكل كبير من حجم مجموعة البيانات في كل تكرار. أما مع تقنيةEFB ، فيقوم LightGBM بتقليل عدد الميزات عن طريق تجميع الميزات المتنافية وهي تلك التي نادراً ما تأخذ قيم غير صفرية في نفس الوقت مثل الميزات الناتجة عن الترميز الأحادي (one-hot encoding) في ميزة واحدة، مما يسرّع الحسابات بشكل كبير.

اضافة الى ذلك، يتخلى LightGBM عن استراتيجية نمو الشجرة التقليدية القائمة على المستوى (level-wise) لصالح نهج قائم على الورقة(leaf-wise) ، حيث يقوم دائماً بتقسيم الورقة التي ستحقق أكبر انخفاض في الخسارة كما في الشكل في الأسفل .هذا يسمح للنموذج بالتقارب بشكل أسرع بكثير، على الرغم من أنه قد يكون أكثر عرضة لفرط التخصيص (Overfitting)على مجموعات البيانات الصغيرة .هذه التحسينات، جنباً إلى جنب مع خوارزمية عالية الكفاءة قائمة على المدرج التكراري (histogram-based) لإيجاد نقاط الانقسام، تجعل من LightGBM خيار سريع وفعال للغاية من حيث استهلاك الذاكرة للمهام واسعة النطاق في تعلم الآلة . [22]

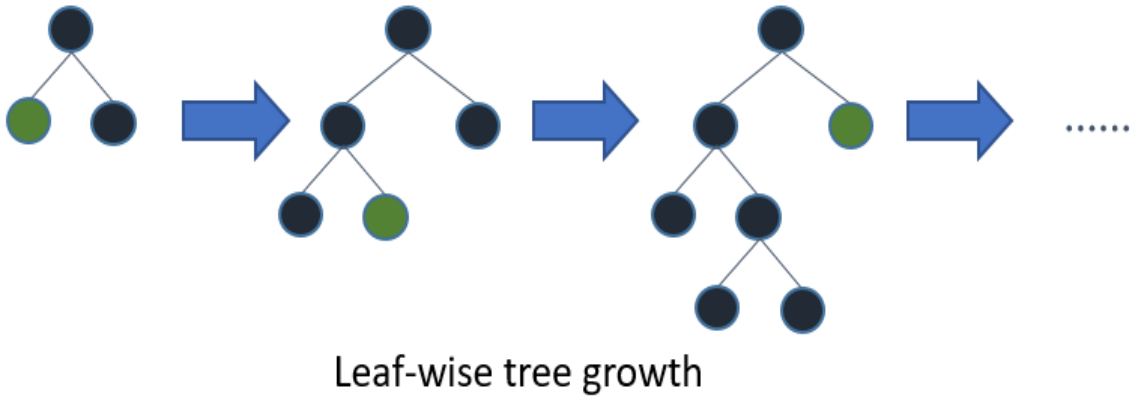


Figure8 Leaf-Wise growth used by LightGBM

LighGBM Hyperparameters

تعتمد الكفاءة والقدرة التنبؤية لنموذج LightGBM على مجموعة من المعلمات الفائقة (Hyperparameters) التي تضبط آلياته الأساسية، خاصة استراتيجياته المتعلقة بنمو الشجرة القائم على الورقة (leaf-wise) وأخذ العينات من البيانات. وبالتالي يجب ضبط هذه الوسطاء لزيادة الدقة إلى أقصى حد مع منع فرط التخصيص(Overfitting) .

معلمات التحكم في تعقيد الشجرة (decision tree complexity)

- num_leaves: تعد هذه المعلمة الأكثر أهمية للتحكم في مدى تعقيد الأشجار الفردية. فهي تحدد العدد الأقصى للأوراق الطرفية في الشجرة الواحدة. على عكس max_depth التي تقيد الشجرة عموديا، تتحكم num_leaves في العدد الإجمالي للعقد الطرفية. ونظراً لأن LightGBM ينمو الأشجار بطريقةleaf-wise ، فإن استخدام قيمة أعلى لـ num_leaves يسمح للنموذج بتكوين حدود قرار أكثر تعقيداً وتفصيلاً. ومع ذلك، فإن القيمة العالية جداً يمكن أن تؤدي بسهولة إلى فرط التخصيص. وغالبا يُنصح بإبقاء قيمة num_leaves أقل من.(max_depth)^2
- max_depth: تحدد هذه المعلمة أقصى عمق يمكن أن تنمو إليه الشجرة. على الرغم من أن LightGBM ينمو بطريقةleaf-wise ، إلا أن تحديد max_depth يعمل كإجراء وقائي لمنع الأشجار من أن تصبح عميقة بشكل مفرط، وهو سبب رئيسي للoverfitting. وبالتالي هو لمنع التعقيد الكبير للشجرة.
- (min_child_samplesأو min_data_in_leaf): تحدد هذه المعلمة الحد الأدنى لعدد نقاط البيانات التي يجب أن تكون موجودة في العقدة الطرفية. وهي معلمة تنظيم (Regularization) حاسمة لنمو الشجرة القائم على الورقة. إن تعيينها على قيمة كبيرة يمنع النموذج من إنشاء انقسامات مدعومة فقط بعدد قليل من نقاط البيانات، وبالتالي تجنب التقاط الضوضاء الخاصة بمجموعة التدريب وتحسين قدرة النموذج على التعميم ومنع الoverfitting.

معلمات عملية التعزيز(Boosting)

- n_estimators: تحدد هذه المعلمة العدد الإجمالي لجولات التعزيز، أو ما يعادل عدد الأشجار التي سيتم بناؤها بشكل تسلسلي. يؤدي استخدام عدد أكبر من الأشجار بشكل عام إلى تحسين أداء النموذج، ولكن هذا التأثير يتضاءل بمرور الوقت ويزيد من خطر فرط التخصيص والتكلفة الحسابية. عادةً ما يتم تحديد قيمتها المثلى باستخدام آلية التوقف المبكر(Early Stopping) .
- learning_rate: وهي تقوم بتقليص حجم مساهمة كل شجرة في التنبؤ النهائي. إن استخدام learning_rate أصغر يجعل عملية التعزيز أكثر تحفظاً، مما يتطلب عددًا أكبر من n_estimators لتحقيق أداء جيد، ولكنه يؤدي في النهاية إلى نموذج أكثر قوة وقابلية للتعميم.

معلمات أخذ العينات والعشوائية

- feature_fraction (or colsample_bytree): تحدد هذه المعلمة جزء الميزات الذي سيتم النظر فيه عشوائياً لكل شجرة. على سبيل المثال، تعني القيمة 0.8 أن LightGBM سيختار 80% من الميزات عشوائياً قبل بناء كل شجرة. هذا يُدخل عشوائية تساعد على فك الارتباط بين الأشجار وتقليل فرط التخصيص لأن كل شجرة اصبحت مختلفة عن غيرها نوعا ما.
- bagging_fraction (subsample): تتحكم هذه المعلمة في جزء البيانات الذي سيتم استخدامه لتدريب كل شجرة. حيث يتم اختيار جزء من البيانات عشوائياً بدون إرجاع. هذه التقنية، المعروفة باسم (Bagging) ، يمكن أن تسرّع التدريب وهي طريقة فعالة أخرى لمنع فرط التخصيص. لكي تكون هذه المعلمة نشطة، يجب تعيين bagging_freqإلى عدد صحيح موجب.
- bagging_freq:تحدد هذه المعلمة عدد مرات تكرار عملية الbagging. القيمة k تعني أن عملية الbagging ستتم كل k تكرار. القيمة 0 تعطل هذه العملية.

معلمات التنظيم(Regularization)

- lambda_l1 (L1 regularization) and lambda_l2 (L2 regularization): تطبق هاتان المعلمتان تنظيم L1و L2 على أوزان الأوراق الطرفية على التوالي. تؤدي زيادة هذه القيم إلى إضافة عقوبة على تعقيد النموذج، مما يفرض أن تكون الأوزان أصغر ويجعل النموذج أكثر تحفظاً، وهي تقنية لمنع فرط التخصيص (overfitting).

[22] [23]

الشبكات العصبونية (Neural Network)

يُعد مصنف الشبكة العصبونية نموذج حسابي قوي في مجال تعلم الآلة، وهو مستوحى من بنية ووظيفة الدماغ البيولوجي. تم تصميم هذا النموذج لتعلم العلاقات المعقدة وغير الخطية(non-linear) داخل البيانات بهدف أداء مهام التصنيف(classification). الوحدة الأساسية لبناء الشبكة هي العصبون الاصطناعي (artificial neuron) (أو العقدة)، الذي يستقبل مدخلاً واحداً أو أكثر، ويقوم بإجراء عملية جمع مرجح (weighted sum) للأوزان، ثم يضيف قيمة انحياز (bias)، وبعد ذلك يمرر النتيجة عبر تابع تنشيط (activation function) غير خطية لإنتاج مخرجاته.

يتم تنظيم هذه العصبونات في سلسلة من الطبقات :طبقة الإدخال (input layer) التي تستقبل بيانات الميزات الأولية (raw feature data)؛ وطبقة مخفية (hidden layer) واحدة أو أكثر، وهي المسؤولة عن تعلم أنماط وتمثيلات أكثر تجريد بشكل تدريجي من البيانات؛ وطبقة الإخراج (output layer) التي تنتج تنبؤ التصنيف النهائي .تمتلك الاتصالات بين العصبونات في الطبقات المتجاورة أوزاناً (weights) مرتبطة بها، وهي المعلمات الأساسية التي يتعلمها النموذج أثناء التدريب. كما في الشكل:

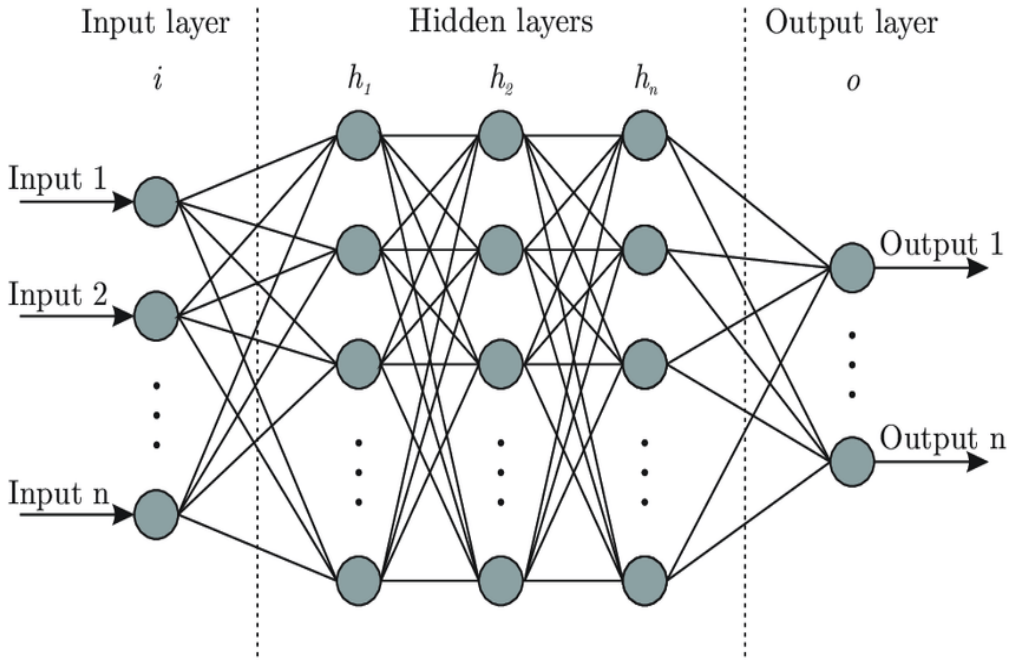


Figure9 Neural Network layers

تتضمن عملية التدريب، المعروفة باسم التعلم الخاضع للإشراف (supervised learning) ، تغذية الشبكة بمجموعة بيانات كبيرة ومصنفة. لكل دخل، تُنتج الشبكة تنبؤ يتم مقارنته بالتصنيف الحقيقي باستخدام دالة خسارة (loss function) مثل الإنتروبيا المتقاطعة (cross-entropy) لتحديد مقدار الخطأ .بعد ذلك، يتم نشر هذا الخطأ بشكل عكسي عبر الشبكة باستخدام خوارزمية تسمى الانتشار الخلفي (backpropagation) ، والتي تحسب تدرج (gradient) دالة الخسارة بالنسبة لكل وزن .ثم تستخدم خوارزمية أمثلية (optimization algorithm) ، مثل الانحدار التدريجي (Gradient Descent) ،

هذه التدرجات لإجراء تعديلات طفيفة على الأوزان، مما يقلل الخطأ بشكل متكرر .في مهام التصنيف متعدد الفئات، تستخدم طبقة الإخراج عادةً دالة تنشيطSoftmax، التي تحول النتائج الأولية للشبكة إلى توزيع احتمالي عبر جميع الفئات الممكنة(classes)، وتكون الفئة ذات الاحتمالية الأعلى هي التنبؤ النهائي.

[24]

Neural Network Hyperparameters

يتم التحكم في أداء وسلوك الشبكة العصبونية من خلال مجموعة متنوعة من المعلمات الفائقة (Hyperparameters) التي يجب تكوينها قبل بدء عملية التدريب. يمكن تصنيف هذه المعلمات بشكل عام إلى تلك التي تحدد بنية الشبكة (Architecture) وتلك التي تتحكم في عملية التدريب والأمثلية (Optimization). وهم:

المعلمات الفائقة المعمارية: Architectural Hyperparameters

- عدد الطبقات المخفية (Number of Hidden Layers): يحدد هذا عمق الشبكة. الشبكة التي لا تحتوي على طبقات مخفية هي نموذج خطي بسيط، بينما تسمح إضافة طبقة مخفية واحدة أو أكثر للشبكة بتعلم وظائف غير خطية أكثر تعقيداً وتجريداً بشكل تدريجي. تتمتع الشبكات الأعمق بقدرة تمثيلية أكبر ولكنها أيضاً أكثر تكلفة من الناحية الحسابية وقد يكون تدريبها أكثر صعوبة.
- عدد العصبونات في كل طبقة مخفية (Number of Neurons per Hidden Layer): يحدد هذا عرض كل طبقة وقدرتها الاستيعابية. يسمح عدد أكبر من العصبونات للطبقة بتعلم تمثيلات أكثر تعقيدا من مدخلاتها. ومع ذلك، يمكن أن يؤدي وجود عدد كبير جداً من العصبونات إلى فرط التخصيص (Overfitting)، حيث يحفظ النموذج بيانات التدريب، بينما يؤدي وجود عدد قليل جداً إلى نقص التخصيص (Underfitting)، حيث يفتقر النموذج إلى القدرة على التقاط الأنماط الأساسية في البيانات.
- دوال التنشيط (Activation Functions): هي دوال غير خطية يتم تطبيقها على مخرجات كل عصبون، وهي ضرورية للسماح للشبكة بتعلم العلاقات غير الخطية. تشمل الخيارات الشائعة للطبقات المخفية دالة الوحدة الخطية المصححة (ReLU) ومتغيراتها (مثل Leaky ReLU)، والتي تتسم بالكفاءة الحسابية وتساعد في التخفيف من مشكلة تلاشي التدرج (Vanishing Gradient problem). بالنسبة لطبقة الإخراج في المصنف، تُستخدم دالة Sigmoid للتصنيف الثنائي لإنتاج احتمال، بينما تُستخدم دالة Softmax للتصنيف متعدد الفئات لإنتاج توزيع احتمالي عبر جميع الفئات الممكنة.

معلمات التدريب والأمثلية

- المحسِّن (Optimizer): هي الخوارزمية المحددة المستخدمة لتحديث أوزان الشبكة أثناء الانتشار الخلفي (Backpropagation). في حين أن الانحدار التدريجي العشوائي (SGD) هو الخوارزمية التأسيسية، فإن المحسنات التكيفية الأكثر تقدماً مثل Adam (تقدير العزم التكيفي) و RMSprop و Adagrad أصبحت الآن ممارسة قياسية. تقوم هذه المحسنات بتكييف معدل التعلم لكل وزن على حدة، مما يؤدي غالباً إلى تقارب أسرع وأداء أفضل.
- معدل التعلم (Learning Rate): هذا هو أهم وسيط فهو يحدد حجم الخطوة التي يتخذها المحسن عند تحديث الأوزان في اتجاه التدرج السلبي. يمكن أن يؤدي معدل التعلم الصغير جداً إلى أوقات تدريب طويلة للغاية أو التعثر في حد أدنى محلي غير أمثل. ويمكن أن يؤدي معدل التعلم الكبير جداً إلى تباعد التدريب أو التذبذب بعنف، مما يؤدي إلى الفشل في التقارب على الإطلاق.
- حجم الدفعة (Batch Size): يحدد هذا عدد عينات التدريب التي تتم معالجتها قبل تحديث المعلمات الداخلية للنموذج. يعد استخدام مجموعة البيانات بأكملها (Full-batch) مكلفاً من الناحية الحسابية. بدلاً من ذلك، يتم تقسيم البيانات عادةً إلى دفعات صغيرة (Mini-batches). يُدخل حجم الدفعة الأصغر مزيداً من الضوضاء في تحديثات الوزن، والتي يمكن أن يكون لها تأثير تنظيمي ولكنها قد تؤدي إلى تدريب غير مستقر. يوفر حجم الدفعة الأكبر تقديراً أكثر دقة للتدرج ولكنه يستهلك ذاكرة أكبر.
- عدد الحقب (Number of Epochs): تمثل الحقبة (Epoch) مرور كامل واحد عبر مجموعة بيانات التدريب بأكملها. يحدد عدد الحقب عدد المرات التي سيرى فيها النموذج البيانات. سيؤدي التدريب لعدد قليل جداً من الحقب إلى نقص التخصيص، بينما يمكن أن يؤدي التدريب لعدد كبير جداً إلى فرط التخصيص أي الoverfitting. غالباً ما يتم تحديد العدد الأمثل باستخدام تقنية التوقف المبكر (Early Stopping)، حيث يتم إيقاف التدريب عندما يتوقف الأداء عن التحسن.

معلمات التنظيم (Regularization)

- معدل الاسقاط (Dropout Rate): هي تقنية تنظيم قوية حيث يتم أثناء التدريب إسقاط أو تجاهل جزء عشوائي من العصبونات في طبقة ما مؤقتاً لكل دفعة تدريب. هذا يمنع العصبونات من التكيف المفرط مع بعضها البعض ويجبر الشبكة على تعلم تمثيلات أكثر قوة وتكرار. يحدد معدل التسرب (مثلا 0.2 إلى 0.5) احتمال إسقاط العصبون، وهي مفيدة لمنع حدوث overfitting.

- تنظيم L1/L2 (تضاؤل الوزن – Weight Decay): تضيف هذه التقنيات حد عقوبة إلى دالة الخسارة بناءً على حجم أوزان الشبكة. يعاقب تنظيم L2 (الأكثر شيوعاً) المقدار التربيعي للأوزان، مما يشجع النموذج على تعلم قيم أوزان أصغر وأكثر انتشار، مما يساعد على منع فرط التخصيص.

[19] [24]

معايير القياس والأدوات المستخدمة

معايير القياس (Evaluation Metrics)

لتقييم ومقارنة أداء نماذج التصنيف المختلفة بشكل دقيق، استخدمنا عدة مقاييس تقييم. تعد هذه المقاييس ضرورية لفهم فعالية النموذج، خاصة في المسائل الحساسة التي يكون فيها صف معين مهم مثل مسألتنا هذه (اكتشاف الMalicious).

ان أهم مقياس هو مصفوفة الإرباك (Confusion Matrix) ومنه يمكننا الحصول على باقي المقاييس.

مصفوفة الالتباس (Confusion Matrix)

Table 1 confusion matrix structure

	Predicted: Malicious	Predicted: Benign
Actual: Malicious	True Positive (TP)	False Negative (FN)
Actual: Benign	False Positive (FP)	True Negative (TN)

مصفوفة الالتباس هي عبارة عن جدول يوضح أداء خوارزمية التصنيف. فهي تقدم تفصيل دقيق لعدد التنبؤات الصحيحة وعدد التنبؤات الخاطئة وأنواع الأخطاء التي ارتكبها النموذج. بالنسبة لمشكلة تصنيف ثنائي مثل مسألتنا (حميد أو خبيث)، تحتوي المصفوفة على أربعة أنواع:

- **الإيجابيات الحقيقية (True Positives - TP)**: عدد الحالات التي كانت خبيثة بالفعل (malicious actual) وتنبأ بها النموذج بشكل صحيح على أنها خبيثة.

- **السلبيات الحقيقية (True Negatives - TN)**: عدد الحالات التي كانت حميدة بالفعل (actual benign)وتنبأ بها النموذج بشكل صحيح على أنها حميدة.
- **الإيجابيات الخاطئة (False Positives - FP)**: عدد الحالات التي كانت حميدة ولكن تنبأ بها النموذج بشكل غير صحيح على أنها خبيثة. يُعرف هذا أيضا بالخطأ من النوع الأول.
- **السلبيات الخاطئة (False Negatives - FN)**: عدد الحالات التي كانت خبيثة ولكن تنبأ بها النموذج بشكل غير صحيح على أنها حميدة. هذا هو أخطر أنواع الأخطاء بحسب مسألتنا لأنه يمثل هجوماً لم يتم اكتشافه، ويُعرف أيضا بالخطأ من النوع الثاني.

[25]

نلخصهم في الجدول التالي:

Table2 the significance of cm fields

Prediction	Actual value	Type	Symbol	Explanation
1	1	True Positive	TP	Predicted Positive and was Positive
0	0	True Negative	TN	Predicted Negative and was Negative
1	0	False Positive	FP	Predicted Positive but was Negative
0	1	False Negative	FN	Predicted Negative but was Positive

الدقة (Accuracy)

وهي المقياس الأكثر انتشارا وتمثل الصحة الإجمالية للنموذج. يتم حسابها كنسبة جميع التنبؤات الصحيحة إلى العدد الإجمالي للحالات ونعطى بالعلاقة التالية:

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

ولكنها بالطبع ممكن أن تكون مضللة للغاية في مجموعات البيانات غير المتوازنة. على سبيل المثال، إذا كانت 99% من حركة المرور حميدة، فإن النموذج الذي يتنبأ ببساطة بأن كل اتصال "حميد" سيحقق دقة 99% ولكنه سيكون عديم الفائدة تماماً لأنه لن يكتشف أي هجوم. لهذا السبب، يجب استخدام مقاييس أخرى أيضاً. [25]

الضبط (Precision)

يقيس مدى موثوقية التنبؤات الإيجابية (الخبيثة). إنها تجيب على السؤال الاتي: من بين جميع الاتصالات التي صنفها (توقعها) النموذج على أنها هجوم، ما هي نسبة الهجمات الفعلية؟ تشير درجة الدقة العالية إلى انخفاض (False Positive)، ويعطى بالعلاقة الاتية:

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

[25]

الاستدعاء (Recall) أو الحساسية (Sensitivity)

وهو يقيس قدرة النموذج على تحديد جميع الحالات الإيجابية الفعلية. إنه يجيب على السؤال التالي: من بين جميع الهجمات الفعلية التي حدثت، ما هي النسبة التي نجح النموذج في اكتشافها؟ يعد الاستدعاء العالي أمر بالغ الأهمية لنظام كشف التسلل، لأنه يشير إلى انخفاض معدل السلبيات الخاطئة (FN)، أي توقع عدد قليل جدا من الاتصالات الخبيثة كاتصالات حميدة، ويعطى بالعلاقة التالية:

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

[25]

النوعية (Specificity)

وهي تقيس قدرة النموذج على تحديد جميع الحالات السلبية (الحميدة) الفعلية بشكل صحيح. إنها تجيب على السؤال التالي: من بين جميع حركة المرور الحميدة الفعلية، ما هي النسبة التي حددها النموذج بشكل صحيح على أنها حميدة؟ تعد النوعية العالية مهمة لضمان عدم تصنيف نشاط المستخدم العادي باستمرار على أنه خبيث.، وهو غير مهم في مسألتنا هذه لاننا نهتم بكشف الاتصالات الخبيثة، ويعطى بالصيغة التالية:

$$Specificity = \frac{TN}{TN + FP}$$

ولكن لم يتم استخدامه لعدم أهميته في دراستنا هذه.

[25]

مقياس (F1-Score)

مقياس **F1** هو المتوسط التوافقي للضبط (Precision) والاستدعاء (Recall). حيث يوفرمقياس واحد يوازن بين كلا الهدفين: تقليل false positive (ضبط عالي) وتقليل false negative (استدعاء عالي). يعد مقياس F1 مفيداً بشكل خاص لمجموعات البيانات غير المتوازنة لأنه يوازن بين مقاسي الضبط والاستدعاء. تشير درجة F1 العالية إلى أن النموذج جيد جدا في الناحيتين معاً، ويعطى بالصيغة التالية:

$$F1\ Score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2\ Precision * \ Recall}{Recall + Precision}$$

حيث الضرب ب2 ليصبح مجال المقياس في [0,1] لأن من دون الضرب ب2 كان المجال هو [0,1/2].

[25]

المساحة تحت المنحني (Area Under Curve)

يقدم مقياساً إجمالياً واحداً لأداء نموذج التصنيف عبر جميع عتبات التصنيف الممكنة. المنحنى المعني هنا هو منحنى خصائص التشغيل للمستقبِل (ROC - Receiver Operating Characteristic)، وهو رسم بياني يوضح العلاقة بين معدل الإيجابيات الحقيقية (True Positive Rate) – أي نسبة الحالات الإيجابية الفعلية التي تم تحديدها بشكل صحيح – وبين معدل الإيجابيات الكاذبة (False Positive Rate) – أي نسبة الحالات السلبية الفعلية التي تم تحديدها بشكل خاطئ – عند عتبات مختلفة. وتمثل قيمة AUC، التي تتراوح من 0 إلى 1، المساحة ثنائية الأبعاد الكلية الموجودة أسفل منحنى ROC هذا. حيث أن قيمة AUC تساوي 1.0 تشير إلى نموذج مثالي قادر على التمييز بين الفئات بشكل ممتاز، بينما تشير قيمة 0.5 إلى نموذج لا يمتلك أي قدرة تمييزية، أي ما يعادل التخمين العشوائي. لذلك، فإن مقياس AUC هو مقياس قوي ومفيد للغاية لأنه يقيس القدرة الكلية للنموذج على تصنيف حالة إيجابية عشوائية بشكل صحيح على أنها أكثر احتمالاً من حالة سلبية عشوائية، وذلك بغض النظر عن عتبة التصنيف المحددة التي يتم اختيارها. يمكن تمثيله بالشكل التالي:

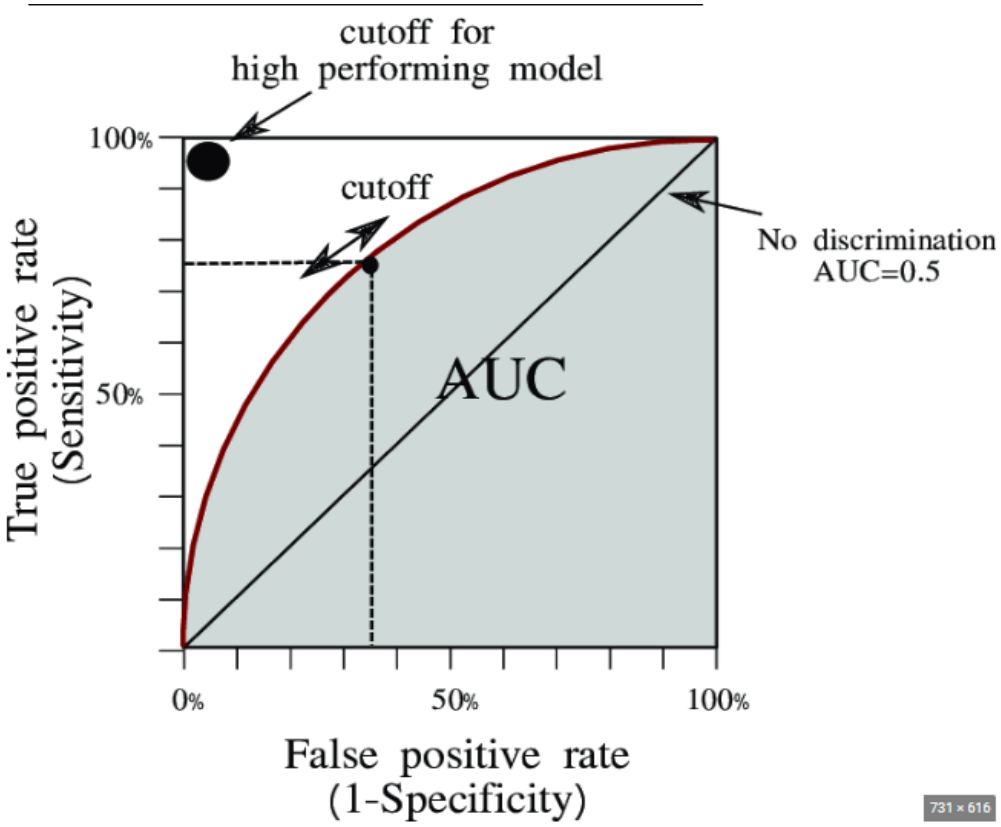


Figure10 **AUC METRIC**

بعد الحديث عن التعلم الالي والخوارزميات المتبعة ومعايير القياس المستحدمة، سنتحدث في الفصل التالي عن التطبيق العملي للمشروع.

الفصل الرابع

التطبيق العملي

سنتحدث في هذا الفصل عن انشاء نظام كشف اختراقات باسخدام 4 خوارزميات، واجراء محاكاة للهجوم والكشف.

مقدمة

يُعد هذا الفصل هو الجزء الأهم في المشروع، حيث يتم فيه الانتقال من الإطار النظري الذي تم استعراضه في الفصول السابقة إلى مرحلة التطبيق العملي وبناء النظام. يترجم هذا الفصل المفاهيم النظرية لأمن إنترنت الأشياء، وخوارزميات تعلم الآلة، وتقنيات كشف التسلل إلى بنية نظام متكاملة وقابلة للتشغيل. سنبدأ بتحديد المتطلبات الوظيفية وغير الوظيفية التي شكلت أساس تصميم النظام، لضمان قدرته على التعامل مع تحديات العالم الحقيقي من حيث الأداء والموثوقية والأمان. بعد ذلك، سيتم استعراض تفصيلي لمعمارية النظام المتبعة، موضحين كيفية دمج المكونات المختلفة، لإنشاء نظام كشف تسلل شبكي (NIDS) فعال . كما سيغطي هذا الفصل بالتفصيل دورة حياة البيانات بأكملها، من استخراجها ومعالجتها من مجموعة بيانات IoT-23، إلى تدريب وتقييم أربعة نماذج تصنيف مختلفة (Random Forest, XGBoost, LightGBM, Neural Network)، مع شرح مبررات اختيار المعلمات الفائقة لكل منها. وأخيراً، سنقدم البيئة الافتراضية التي تم بناؤها لمحاكاة هجوم حقيقي، واختبار قدرة النظام على كشف التهديدات في الزمن الفعلي.

المتطلبات الوظيفية

1- القدرة على تحميل بيانات التدريب والاختبار (IoT-23) بصيغة csv. وهي الصيغة التي استخدمناها من أجل :
   1-1. تدريب النموذج على بيانات التدريب (Training Data).
   1-2. اختبار النموذج على بيانات الاختبار (Testing Data).
2- معالجة البيانات بشكل مسبق (preprocessing data) ومنها:
   2-1. معالجة القيم المفقودة (NaN) مثل اسناد قيمة الصفر اليها او اسناد قيمة الوسيط (median) اليها.
   2-2. ترميز (encoding) قيم السمات من قيم فئوية (categorical values) الى قيم رقمية.
   2-3. حذف (drop) السمات غير المفيدة.
   2-4. تحويل أنماط (types) السمات الى الأنماط المناسبة لطبيعة قيمها.

2-5. تقسيم البيانات الى بيانات للتدريب وبيانات للاختبار باعتماد نسبة معينة للتقسيم مثل (30%-70%).

3- تدريب نموذج (classifier) أو أكثر وتقييم النماذج حيث أنه يجب أن يكون النظام قادرا على:

3-1. التدريب على نماذج مثل الغابة العشوائية والشبكات العصبونية (Random Forest, Neural Network, etc).

3-2. قادراً على تقييم كل نموذج باستخدام المقاييس المختلفة مثل: الدقة (Precision) والاسترجاع (Recall) و-F1 score والمساحة تحت المنحني (AUC: Area Under Curve) ومصفوفة الارتباك (Confusion Matrix).

3-3. القدرة على حفظ النموذج المدرَّب من أجل الاستخدام المستقبلي.

4- القدرة على تحميل النموذج واستخدامه للكشف في الوقت الحقيقي أي يجب أن يكون النظام قادراً على :

4-1. تحميل النوذج المدرّب من أجل استخدامه لكشف الهجومات.

4-2. استقبال البيانات ومعالجتها بشكل مباشر في الوقت الحقيقي أي يجب أن يكون النظام قادراً على:

4-2-1. مراقبة الشبكة وأخذ بيانات الاتصالات الواردة الى جهاز الكشف.

4-2-2. معالجة البيانات الملتقطة بحيث يتم حذف السمات واجراء معالجة للبيانات لتكون صيغتها مماثلة لصيغة البيانات التي تدرّب عليها النموذج المحمَّل.

4-3. التنبّؤ بماهية الاتصال ان كان سليم او هجوم اعتماداً على البيانات التي التقطها وعالجها.

4-4. توليد مخرجات تنبؤ النموذج المحمل بطريقة واضحة.

المتطلبات غير الوظيفية

1- أداء جيد للنظام حيث يجب أن:

1-1. معالجة اتصالات الشبكة واجراء التنبؤات بشكل سريع جدّا ليكون قريب من الوقت الفعلي مثلا اعطاء التنبّؤ المتوَقّع للاتصال في مدة أقل من 50 ميلي ثانية.

1-2. يكون قادراً على التعامل مع حركة مرور كبيرة ضمن الشبكة ومعالجتها من دون تأخير أي أنه يجب أن يتحمل الضغط الناجم عن الاتصالات.

1-3. يكون قادراً على استخدام العتاد بفعالية مثل استخدام محدود للذاكرة الرئيسية لتجنب فشل النظام.

2- يكون النظام موثوقا حيث أنه يجب أن:

2-1. يتوافر النظام بشكل كبير (availability) ويعمل باستمرار.

2-2. تكون دقّة النظام جيّدة بما يكفي من أجل اكتشاف الهجومات مثلاً أن تكون مقاييس الدقة تتجاوز ال90% عند تقييمها على بيانات الاختبار (Testing data).

2-3. يتعامل بسماحية مع الأخطاء مثلا حزم مشوّهة، أي بقاء النظام شغّال حتى لو حدث أخطاء خلال التحليل والكشف.

3- أمان النظام والملفات الخاصة به حيث يجب تأمين ملفات الأكواد والنموذج المحَمَّل من الوصول لغير المخوّلين او من التعديل والعبث بهم.

4- يكون النظام سهل الاستخدام فممكن عرض تعليمات لكيفية استخدامه خصوصاً لتشغيل خدمة الكشف.

5- تتوفر إمكانية التحديث والتغيير فمثلاً يجب أن يسمح النظام بتحميل نموذج اخر اذا كان أفضل من النموذج الحالي.

Dataset IOT-23

مقدمة وتوصيف لمجموعة البيانات

تعتمد فعالية أي نموذج من نماذج تعلم الآلة وقابليته للتطبيق في الواقع بشكل أساسي على جودة وملاءمة البيانات التي يتم تدريبه عليها. لهذا المشروع، تم اختيار مجموعة بيانات IoT-23 التي طورها مختبر الستراتوسفير (Stratosphere Laboratory) في الجامعة التقنية التشيكية. تُعد هذه المجموعة من البيانات حديثة وواسعة النطاق، وقد تم إنشاؤها خصيصاً لأغراض البحث والتطوير في مجال أنظمة كشف التسلل (Intrusion Detection Systems) لبيئة إنترنت الأشياء (Internet of Things).

نظرة عامة على مجموعة البيانات وتكوينها:

تتألف مجموعة بيانات IoT-23 من ٢٣ تسجيلاً لحركة مرور الشبكة، يُشار إلى كل منها بـ "سيناريو". تم إنشاؤها عبر تسجيل حركة المرور من أجهزة إنترنت الأشياء الحقيقية ضمن بيئة شبكة مراقبة بين عامي 2018 و 2019، مما يضمن أن البيانات تعكس سلوكيات الأجهزة والتهديدات المعاصرة.

تنقسم السيناريوهات الـ 23 إلى فئتين رئيسيتين:

- 20 سيناريو خبيث (Malicious Scenarios): تم توليد حركة المرور هذه عن طريق إصابة جهاز Raspberry Pi (يعمل كجهاز إنترنت أشياء متعدد الاستخدامات) بعينات متنوعة من البرمجيات الخبيثة. شملت عائلات البرمجيات الخبيثة المستخدمة أنواعاً متنوعة وذات صلة وثيقة بمشهد التهديدات في عالم إنترنت الأشياء، مثل Mirai، وToriii، وGagfyt، وHajime. قامت هذه البرمجيات بتنفيذ مجموعة من الإجراءات الخبيثة، بما في ذلك هجمات الحرمان من الخدمة (Denial-of-Service)، ومسح المنافذ (Port Scanning)، وإنشاء اتصالات القيادة والتحكم (C&C).
- 3 سيناريوهات حميدة (Benign Scenarios): لتوفير خط أساس للنشاط الطبيعي وغير الخبيث، تم تسجيل حركة المرور من ثلاثة أجهزة إنترنت أشياء تجارية مختلفة: مصباح Philips HUE الذكي، ومكبر صوت Amazon

Echo المنزلي الذكي، وقفل باب Somfy الذكي. والأهم من ذلك، أن هذه البيانات تم توليدها من أجهزة حقيقية وليست محاكاة، مما يسمح بتحليل سلوك شبكي حقيقي وموثوق.

تنسيق البيانات وعملية التصنيف (Labeling):

لكل سيناريو، تم تسجيل حركة مرور الشبكة الأولية كملف بصيغة .pcap بعد ذلك، تمت معالجة هذه التسجيلات باستخدام إطار عمل Zeek لتحليل الشبكات (المعروف سابقاً باسم Bro) لإنشاء سجلات الاتصال (conn.log) تكمن القيمة الأساسية لمجموعة بيانات IoT-23 في عملية التصنيف الدقيقة التي خضعت لها؛ حيث قام محللون بشريون بالتحقيق يدوياً في حركة المرور ووضع قواعد لتصنيف كل تدفق شبكي (network flow) داخل ملفات conn.log.

كانت النتيجة هي ملفات conn.log.labeled التي تم استخدامها في هذا المشروع، والتي لا تحتوي فقط على ميزات اتصال Zeek القياسية (مثل البروتوكول، مدة الاتصال، حجم البيانات المرسلة، حالة الاتصال)، بل تحتوي أيضاً على عمود تصنيف (label) حاسم يحدد صراحةً ما إذا كانت حركة المرور "حميدة" (Benign) أو نوع معين من النشاط "الخبيث "(Malicious)". هذا التصنيف المفصل لكل تدفق على حدة يُعد أمر ضروري لمهام تعلم الآلة الخاضع للإشراف (Supervised Machine Learning).

أسباب اختيار dataset iot23

كانت مجموعة بيانات IoT-23 الخيار المثالي لهذا المشروع لعدة أسباب رئيسية:

- الواقعية العالية والملاءمة: إن استخدام أجهزة إنترنت الأشياء الحقيقية لتسجيل حركة المرور الحميدة، واستخدام برمجيات خبيثة حقيقية تعمل على جهاز شبيه بأجهزة إنترنت الأشياء (Raspberry Pi)، يضمن أن البيانات تمثل تمثيلاً عالي الدقة لنشاط شبكات إنترنت الأشياء الحديثة. وهذا يتفوق بشكل كبير على البيانات المولدة عبر المحاكاة البحتة، والتي قد لا تلتقط الفروق الدقيقة في اتصالات الأجهزة في العالم الحقيقي.
- سيناريوهات غنية ومتنوعة: لا تقتصر مجموعة البيانات على نوع واحد من التهديدات، بل تحتوي على مجموعة واسعة من أنواع البرمجيات الخبيثة والسلوكيات الهجومية. هذا التنوع ضروري لتدريب نماذج تعلم آلة قوية وقابلة للتعميم يمكنها اكتشاف أكثر من نوع واحد من التهديدات.
- تصنيفات دقيقة وقابلة للتحقق: بما أنها مجموعة بيانات مصنفة (labeled)، فهي مناسبة تماماً لمهام تعلم الآلة الخاضع للإشراف (Supervised Learning). تم إنشاء التصنيفات من خلال تحليل بشري متخصص، مما يوفر حقيقة مرجعية (ground truth) موثوقة لتدريب النماذج، والأهم من ذلك، لتقييم أدائها بثقة عالية. كما أنها توفر تصنيفات خبيثة محددة (مثل PartOfAHorizontalPortScan و DDos).

- توفر حركة المرور الحميدة (Benign connections) : من التحديات الشائعة في أبحاث كشف التسلل هو الحصول على بيانات حميدة عالية الجودة. من خلال تضمين تسجيلات مخصصة لحركة المرور الحميدة من أجهزة إنترنت الأشياء الاستهلاكية الشائعة، تُمكّن مجموعة بيانات IoT-23 النماذج من تعلم الفرق بين الأنماط الطبيعية والخبيثة، وهو أمر بالغ الأهمية لتقليل النتائج الإيجابية الخاطئة (false positives) في بيئة التشغيل الفعلية.
- تنسيق البيانات وحجمها: يتم توفير البيانات بتنسيق conn.log الخاص بـ Zeek، والذي يعد بحد ذاته شكلاً من أشكال استخلاص الميزات (feature extraction)، حيث يقدم ملخصات عالية المستوى للاتصالات بدلاً من الحزم الأولية. إضافةً الى وجود أحجام كلبرة وكافية من البيانات للتدريب.

لم يتم اختيار جميع السيناريوهات في الداتاست بل اخترنا السيناريوهات او الcaptures التالية:

CTU-IoT-Malware-Capture-1-1, CTU-IoT-Malware-Capture-3-1, CTU-IoT-Malware-Capture-4-1, CTU-IoT-Malware-Capture-5-1, CTU-IoT-Malware-Capture-7-1, CTU-IoT-Malware-Capture-8-1, CTU-IoT-Malware-Capture-9-1, CTU-IoT-Malware-Capture-20-1, CTU-IoT-Malware-Capture-21-1, CTU-IoT-Malware-Capture-34-1, CTU-IoT-Malware-Capture-35-1, CTU-part of IoT-Malware-Capture-36-1, CTU-IoT-Malware-Capture-42-1, CTU-IoT-Malware-Capture-44-1, CTU-IoT-Malware-Capture-49-1, CTU-IoT-Malware-Capture-60-1

فيما يلي جدول يلخصم مع معلومات عن هذه السيناريوهات:

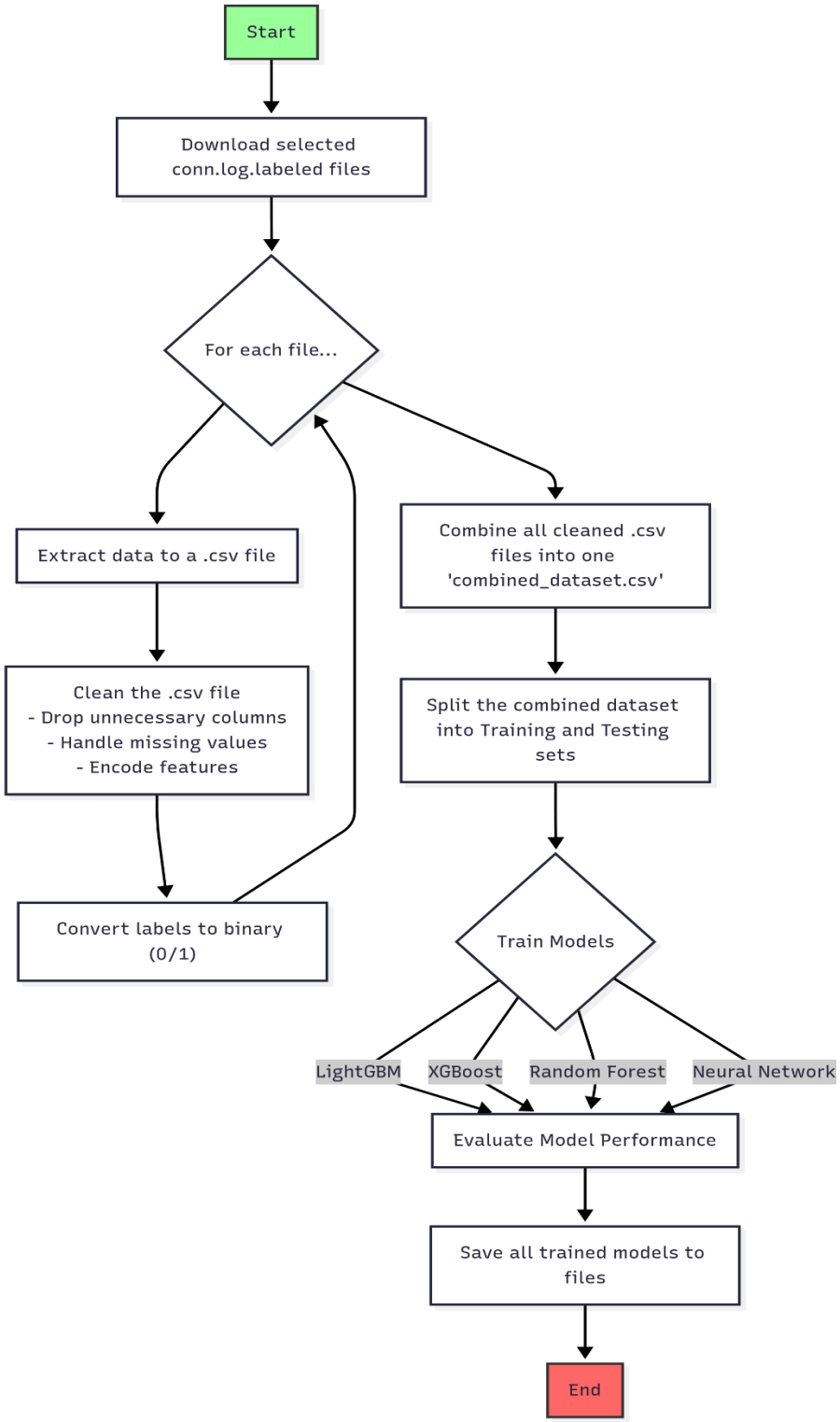
Table 3 Choosen Captures with descriptions

Scenario Name	Malware Type	Brief Description
CTU-IoT-Malware-Capture-1-1	Hide and Seek	شبكة بوت نت (Botnet) من نوع الند للند(P2P)
CTU-IoT-Malware-Capture-3-1	Muhstik	دودة (Worm) تنتشر عبر استغلال ثغرات تطبيقات الويب وbrute-force

CTU-IoT-Malware-Capture-4-1	Benign	حركة مرور طبيعية لمصباح Philips HUE الذكي.
CTU-IoT-Malware-Capture-5-1	Benign	حركة مرور طبيعية لمكبر صوت Amazon Echo المنزلي.
CTU-IoT-Malware-Capture-7-1	Mirai	انتشار عبر هجمات القوة الغاشمة على بروتوكولTelnet
CTU-IoT-Malware-Capture-8-1	Hakai	انتشار عبر هجمات القوة الغاشمة على بروتوكولTelnet
CTU-IoT-Malware-Capture-9-1	Hajime	شبكة بوت نت (Botnet) من نوع الند للند(P2P)
CTU-IoT-Malware-Capture-20-1	Torii	شبكة بوت نت متطورة تركز على البقاء الدائم وتسريب البيانات.
CTU-IoT-Malware-Capture-21-1	Torii	شبكة بوت نت متطورة.
CTU-IoT-Malware-Capture-34-1	Mirai	انتشار عبر هجمات القوة الغاشمة على بروتوكولTelnet
CTU-IoT-Malware-Capture-35-1	Mirai	انتشار عبر هجمات القوة الغاشمة على بروتوكولTelnet
CTU-part of IoT-Malware-Capture-36-1	Okiru	متغير من Mirai ينتشر عبر هجمات brute-force على Telnet
CTU-IoT-Malware-Capture-42-1	Trojan	برمجية حصان طروادة (Trojan) خبيثة.
CTU-IoT-Malware-Capture-44-1	Mirai	انتشار عبر هجمات القوة الغاشمة على بروتوكولTelnet
CTU-IoT-Malware-Capture-49-1	Mirai	انتشار عبر هجمات القوة الغاشمة على بروتوكولTelnet
CTU-IoT-Malware-Capture-60-1	Gagfyt	متغير من Mirai ينتشر عبر هجمات القوة الغاشمة على Telnet

حيث أنهم كلهم يحتوون اتصالات حميدة benign connections واتصالات خبيثة Malicious connections من عدة أنواع ولكننا لن نهتم لأنواع الهجومات لأن مسألتنا ستكون فقط التصنيف بين حميد benign وخبيث malicious.

المخطط التالي يعبر بشكل مختصر عن عملية استخراج مجموعة البيانات وتجهيزها للتدريب (preprocessing) والتدريب عليها، ويلي المخطط شرح الخطوات بالتفصيل:



Training flow chart Figure 11

استخراج البيانات

بعد الحصول على ملفات conn.log.labeled الستة عشر ذهبنا الى مرحلة استخراج البيانات من صيغتها السابقة الى ملفات من صيغة .csv حيث أن الملفات ال16 التي لدينا هي عبارة عن ملفات conn.log.labeled وهو ملف نصي تكون فيه البيانات مفصولة بمسافات جدولية (tab-separated)، ويتم إنشاؤه بواسطة أداة مراقبة الشبكات Zeek ولاستخراج البيانات منه تم تصميم كود بلغة بايثون (python script) وباستخدام المكاتب pandas,numpy إن وظيفة الكود البرمجي هي قراءة ملف بصيغة conn.log.labeled ويقوم بتفسير بنية الملف وترويساته الىheaders)) بشكل صحيح أي يقوم بمعرفة السمات (features) والقيم التابعة لكل سمة ويقوم بالعمليات المناسبة ليحفظ قيم كل سمة ضمن نفس العمود الخاص بالسمة ويحفظهم في اطار بيانات (dataframe) ومن ثم تصدير الاطار الى ملف .csv

إن هذه العملية تُعاد من أجل كل ملف conn.log.labeled وبالتالي ينتج لدينا 16 ملف بيانات من صيغة .csv

وها هم السمات التي تم استخراجها كاملة وسنغمق bold السمات التي سنستخدمها في مشروعنا:

Features of Iot-23 Dataset Table 4

Feature	Description
ts	The timestamp of the connection in UNIX epoch format.
uid	A unique identifier (ID) assigned to each specific connection.
id.orig_h	The IP address of the endpoint that initiated the connection (the originator).
id.orig_p	The TCP/UDP port of the originating endpoint.
id.resp_h	The IP address of the endpoint that the connection is destinated to.
id.resp_p	The TCP/UDP port of the responding endpoint.
proto	The transport layer protocol used for the connection (e.g., TCP, UDP, ICMP).
service	The application layer protocol that was dynamically detected (e.g., HTTP, DNS).
duration	The total duration of the connection, from the first packet to the last packet.
orig_bytes	The number of payload bytes sent from the originator to the responder.

resp_bytes	The number of payload bytes sent from the responder to the originator.
conn_state	The state of the TCP connection.
local_orig	A boolean indicating if the connection originated from the local network.
Local_resp	whether the responding endpoint is on the local network.
missed_bytes	The number of bytes missing from the content of the connection.
history	A string representing the state history of the connection (e.g., 'S' for SYN, 'A' for ACK).
orig_pkts	The total number of packets sent by the originator.
orig_ip_bytes	The total number of IP-layer bytes sent by the originator.
resp_pkts	The total number of packets sent by the responder.
resp_ip_bytes	The total number of IP-layer bytes sent by the responder.
tunnel_parents	The UID of the parent connection if the traffic is tunneled.
label	The classification label assigned to the flow (e.g., 'Benign' or 'Malicious').
detailed-label	A more specific label describing the type of activity (e.g., 'DDoS', 'PortScan').

ومن ثم يجب تطبيق المعالجات عليهم ليكونو جاهزين لاحقاً لعملية التدريب.

المعالجة المسبقة للبيانات

الان بعدما استخرجنا البيانات من ملفات conn.log.labeled وحصلنا عليهم بصيغة ملفات .csv يجب الان تطبيق معالجة للبيانات قبل أن يتم التدريب عليهم حيث أنه يوجد كود برمجي بلغة بايثون (python script) يقوم من أجل كل ملف بيانات .csv بتنفيذ التالي:

1- يقوم بتحويل المسألة الة مسألة من نوع تصنيف ثنائي(Binary classification) حيث يقوم بتحويلها الة تصنيف الاتصالا الى نوعين فقط هم حميدة (benign) وخبيثة (malicious) وقمنا بترميز الصفوف ب "0" للصف الحميد و "1" لجميع تصنيفات الصفوف الخبيثة.

2- تم حذف جميع السطور –التي تمثل الاتصالات– التي لا تحتوي تصنيف لأنها لن تفيدنا وهي عبارة عن ضجيج noisy (data)

3- تم حذف العديد من السمات (features) وهم:

4- { 'Unnamed: 0', 'ts', 'uid', 'local_orig', 'local_resp', 'id.orig_h', 'id.resp_h', 'id.orig_p', 'history' }

5- تم تحويل السمات {'resp_bytes', 'orig_bytes', 'duration'} من نمط (object) أي سلسلة نصية الى قيم رقمية من نمط float

6- تم معالجة القيم المفقودة داخل الأعمده {'resp_bytes', 'orig_bytes', 'duration'} عبر حالتين:

6-1. بالنسبة للاتصالات ذات الحالة S0 (والتي تشير إلى محاولة اتصال لم تتم الإجابة عليها)، تم ملء القيم المفقودة بالقيمة 0 لأن الحالة لدينا هنا هي محاولة اتصال لم يتم الرد عليها وبالتالي لم يتم تبادل معطيات ومدة الاتصال ستكون صفر.

6-2. بينما في جميع حالات الاتصال الأخرى، استُخدمنا الوسيط (Median) للعمود المعني لتعويض القيم المفقودة منه، وذلك لتجنب تحييز توزيع البيانات.

حيث أنه عند تطبيق الكودالبرمجي على كل ملف بيانات غير مُعالج ينتج لدينا مجموعة بيانات معالجة يتم حفظها في ملف csv. أيضا وبالتالي سينتج لدينا 16 ملف لمجموعة بيانات معالجة.

تبرير حذف السمات

يتم حذف السمتين **ts** (الطابع الزمني) و **uid** (المعرف الفريد) بسبب انعدام صلتهما بالمسألة وافتقارهما إلى أي قوة تنبؤية. فالطابع الزمني يسجل وقتاً محدداً لا يمثل مؤشراً عاماً على طبيعة الاتصال، والنموذج الذي يتدرب عليه سيفشل في التعميم على بيانات مستقبلية. وبالمثل، يُعد uid معرفاً فريداً لكل اتصال، مما يجعله مجرد ضجيج (noise) لا يحمل أي نمط متكرر يمكن للنموذج تعلمه.

يتم استبعاد سمتي عناوين IP، وهما **id.orig_h** (IP المصدر) و **id.resp_h** (IP المستجيب)، بسبب مشكلة التعددية العالية (High Cardinality) وضعف قدرتهما على التعميم. فوجود الملايين من عناوين IP المحتملة يجعل من المستحيل على النموذج تعلم أنماط سلوكية؛ وبدلاً من ذلك، سيقوم بحفظ قائمة بالعناوين المرتبطة بالهجمات في بيانات التدريب، مما يؤدي إلى فرط التخصيص (overfitting) وفشله في كشف نفس الهجوم عندما يأتي من مصدر جديد.

يُحذف **id.orig_p** (منفذ port المصدر) أيضاً بسبب التعددية العالية وانعدام صلته. عادةً ما يكون منفذ المصدر منفذاً مؤقتاً وعشوائياً (ephemeral port) يخصصه نظام التشغيل لكل اتصال جديد، وبالتالي لا يحمل أي نمط ذي معنى. هذا على عكس منفذ الوجهة (id.resp_p) الذي يُعد سمة بالغة الأهمية لأنه يحدد الخدمة المستهدفة (مثل منفذ 80 لـ HTTP)، ولذلك يتم الإبقاء عليه.

ويتم حذف **local_resp**/**local_orig** لأنهما عبارة عن أعمدة فاضية (خالية تماما من القيم).

أما السمة **history** و فيتم حذفها بسبب التعقيد الإضافي. تقدم السمة history تفاصيل معقدة حول انتقالات حالة الاتصال، والتي يتم تلخيصها بشكل كافٍ وموجز في السمة conn_state، مما يجعل history مكررة.

بإزالة هذه السمات، يتم إجبار النموذج على التعلم من الخصائص السلوكية ذات المغزى الأكبر لحركة مرور الشبكة، مثل مدة الاتصال، وحجم البيانات المنقولة، وحالة الاتصال، ونوع الخدمة. ينتج عن هذا نموذج أكثر قوة ودقة وقدرة على التعميم لمواجهة التهديدات الجديدة وغير المرئية.

دمج البيانات

بعدما حصلنا على 16 ملف لمجموعات بيانات نضيفة الان يجب دمجهم جميعاً الى مجموعة بيانات واحدة، حيث أنه تم دمج ملفات .csv المعالجة الخاصة بكل سيناريو في مجموعة بيانات رئيسية واحدة تحت اسم 'combined_dataset.csv' حيث أنه طورنا كود برمجي بلغة بايثون يقوم بقراءة جميع ملفات ال.csv ويدمجها الى ملف واحد بشكل عمودي وهكذا أصبح لدينا مجموعة بيانات dataset من 16 سيناريو مختلف وبالتالي ستكون مناسبة جداً لتدريب نماذج والاختبار عليها بسبب احتواءها على سيناريوهات مختلفة وبالتالي انواع مختلفة وقيم سطور connections مختلفة.

تدريب المصنفات (classifiers) واختبارهم

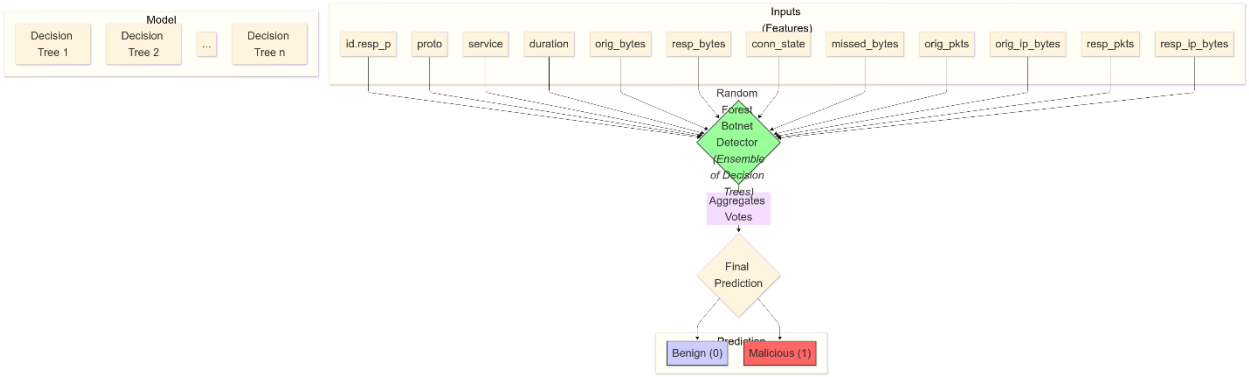
لقد دربنا 4 أنواع مختلفة من المصنفات (الclassifiers) على مجموعة البيانات التي حصلنا عليها وهم:

الغابة العشوائية Random Forestو LightGBMو XGBoostو الشبكة العصبونية Neural Network

حيث أن المصنفات الثلاثة الاولى هم تعلم الي (Machine Learning) والمصنف الرابع هو تعلم الي عميق (Deep Learning)

وذلك بهدف تدريب العديد من النموذج والمقارنة بينهم.

مصنف الغابة العشوائية (Random Forest)



Random forest diagram Figure 12

Random Forest Hyperparameters

في الجدول التالي يوجد الhyperparameters المختارة للمصنف ويليها شرح مفصل لهم.

Table 5 Random Forest Hyperparameters

Hyperparameter	Value	Description
n_estimators	200	The number of trees in the forest.
max_depth	20	The maximum depth of each tree.
min_samples_split	5	The minimum number of samples required to split an internal node.
min_samples_leaf	2	The minimum number of samples required to be at a leaf node.
max_features	sqrt'	The number of features to consider when looking for the best split (sqrt of total features).
class_weight	balanced'	Adjusts weights inversely proportional to class frequencies to handle imbalance.
n_jobs	1-	Use all available CPU cores for training.
random_state	42	Seed used for reproducibility.

Random Forest Implementation

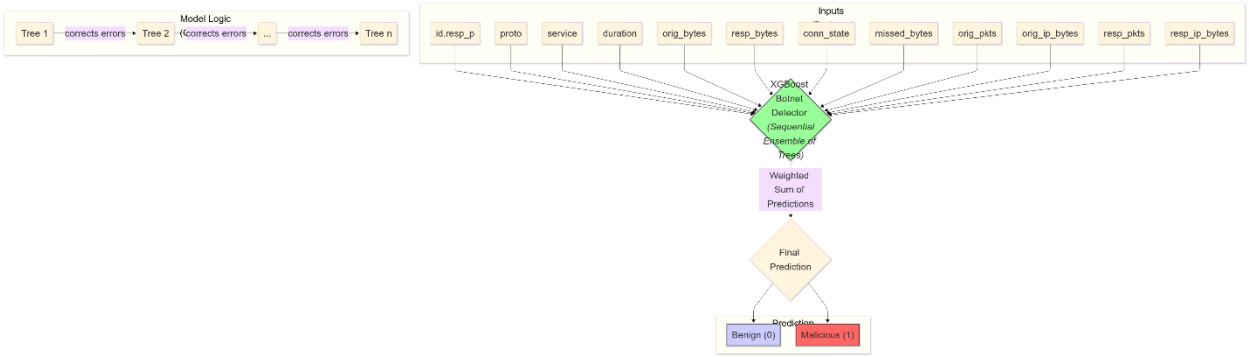
لقد تم تدريب مصنف الغابة العشوائية باستخدام كود بايثون (python code) حيث تم جلب المصنف من مكتبة sklearn عبر التالي: from sklearn.ensemble import RandomForestClassifier

وبالطبع تم قراءة ملف البيانات combined_dataset.csv وقسمناه الى بيانات للتدريب بنسبة 70% وبيانات للاختبار بنسبة 30%.

ولقد اخترنا الhyperparameters التالية:

- n_estimators=200 تم اختيار 200 شجرة لأن 200 شجرة قرار (Decision tree) هو عدد جيد من الأشجار حيث أن كل شجرة تعطي تنبؤ معين وبالتالي 200 شجره عدد كافي لجدوث الاستقرار بأخذ القرار ومنع الoverfitting.
- max_depth=20تم تحديد أقصى عمق للشجرة عند 20 لمنع النموذج من أن يصبح معقداً بشكل مفرط ومن أجل الا يعاني من ال(Overfitting) على بيانات التدريب.
- min_samples_split=5و min_samples_leaf=2 تم ضبط هذه الوسطاء على قيم صغيرة لتنظيم النموذج بشكل طفيف، مما يمنع حدوث انقسامات في العقد بناءً على عدد قليل جداً من العينات، ويضمن أن كل تنبؤ نهائي يستند إلى عينتي تدريب على الأقل.
- class_weight='balanced'أظهرت مجموعة بيانات IoT-23 عدم توازن بين الفئات، لذلك كان استخدام الوسيط 'balanced' أمر جيد لمواجهة عدم التوازن، حيث يقوم تلقائياً بتعيين أوزان أعلى للفئة ذات الأقلية، مما يجعل النموذج على إيلاء اهتمام أكبر لها أثناء عملية التدريب وهذا يؤدي الى التوازن بين الصفين.
- random_state=42تم استخدام حالة عشوائية ثابتة لضمان أن تكون عملية تدريب النموذج قابلة للتكرار بالكامل (fully reproducible)، مما يسمح بالحصول على نتائج متسقة وإجراء مقارنات عادلة مع النماذج الأخرى.

XGBoost classifier



Xgboost diagram Figure 13

XGBoost Hyperparameters

في الجدول التالي يوجد الhyperparameters المختارة للمصنف ويليها شرح مفصل لهم.

Table 6 XGBoost Hyperparameters

Hyperparameter	Value	Description
objective	binary:logistic'	Specifies the learning task is binary classification with logistic regression.
eval_metric	logloss'	The evaluation metric used for the validation data.
use_label_encoder	FALSE	Disables the automatic label encoding by XGBoost.
random_state	42	Seed used for reproducibility.
n_jobs	1-	Use all available CPU cores for training.
tree_method	hist'	Uses a faster, histogram-based algorithm for tree construction.

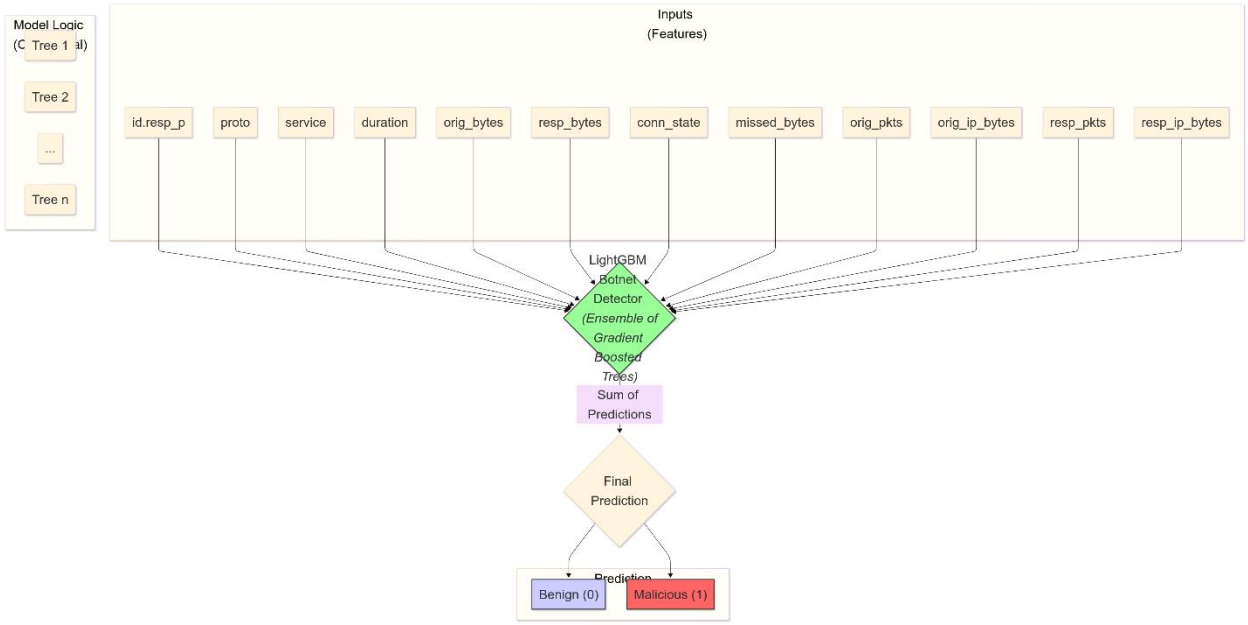
XGBoost Implementation

لقد تم تدريب مصنف xgbosot باستخدام كود بايثون (python code) حيث تم جلب المصنف من مكتبة xgboost عبر التالي: from xgboost import XGBClassifier

وبالطبع تم قراءة ملف البيانات combined_dataset.csv وقسمناه الى بيانات للتدريب بنسبة70% وبيانات للاختبار بنسبة 30%.

- ولقد اخترنا الhyperparameters التالية:
- 'objective='binary:logistic هذا الوسيط مهم لأنه يخبر خوارزمية XGBoost بأن المسألة لدينا هي مسألة تصنيف ثنائي (أي أن هناك نتيجتين محتملتين فقط وهم كما نعلم 0 أي benign و 1 أي malicious). و logistic يحدد أن النموذج يجب أن يُخرج النتائج على شكل احتمالات (قيمة بين 0 و 1) بدلاً من مجرد تصنيف حاسم للتوقع (0 أو 1).
- 'eval_metric='logloss تحدد هذه المعلمة المقياس الذي سيتم استخدامه لتقييم أداء النموذج أثناء التدريب (من أجل التوقف المبكر). مقياس الخسارة اللوغاريتمية (Logloss) يقيس مدى جودة التنبؤات الاحتمالية للنموذج، حيث يفرض عقوبة كبيرة على التنبؤات التي تكون واثقة وخاطئة في نفس الوقت. كلما انخفضت قيمة logloss، كان الأداء أفضل. وهو مناسب للنموذج الذي ينتج احتمالات.
- use_label_encoder=False وهي معلمة تقوم بعمل ترميز للتصنيفات النهائية ووضعناخها false في مسألتنا لأننا قد قمنا بترميز الصفوف سابقا الى 0 و1.
- random_state=42 تم استخدام حالة عشوائية ثابتة لضمان أن تكون عملية تدريب النموذج قابلة للتكرار بالكامل (fully reproducible)، مما يسمح بالحصول على نتائج متسقة وإجراء مقارنات عادلة مع النماذج الأخرى.
- n_jobs=-1 وهي معلمة لضبط الأداء. تخبر XGBoost باستخدام جميع أنوية المعالج (CPU) المتاحة على جهازك لتشغيل أجزاء من عملية التدريب بشكل متوازٍ، وهي ضروري لأن مجموعة بيانات IoT-23 كبيرة جداً، وقد يكون التدريب بطيئ. فاستخدام جميع الأنوية المتاحة سيقلل بشكل كبير من وقت التدريب، مما يجعل التدريب أسرع وأكثر كفاءة.
- 'tree_method='hist تحدد الخوارزمية المستخدمة لبناء الأشجار. الطريقة الافتراضية (exact) تأخذ في الاعتبار كل نقطة انقسام محتملة للميزات. أما طريقة hist فهي خوارزمية تقريبية أسرع بكثير تقوم بتجميع الميزات في (bins) منفصلة (مثل المدرج التكراري histogram) وتجد أفضل الانقسامات بينهم.

Lightgbm classifier



Lightgbm diagram Figure 14

LightGBM Hyperparameters

في الجدول التالي يوجد الhyperparameters المختارة للمصنف ويليها شرح مفصل لهم.

Table 7 LightGBM Hyperparameters

Hyperparameter	Value	Description
objective	binary'	Specifies that the model is performing binary classification.
metric	auc'	The evaluation metric used is the Area Under the ROC Curve.
n_estimators	1000	The maximum number of boosting trees to be built.
learning_rate	0.05	The step size at each iteration while moving toward the minimum of a loss function.
num_leaves	31	The maximum number of leaves in one tree (the default value).
max_depth	1-	No limit on the tree depth.

random_state	42	Seed used for reproducibility.
n_jobs	1-	Use all available CPU cores for training.
colsample_bytree	0.8	Subsample ratio of columns when constructing each tree.
subsample	0.8	Subsample ratio of the training instance.
reg_alpha	0.1	L1 regularization term on weights.
reg_lambda	0.1	L2 regularization term on weights.
scale_pos_weight	2.28	Weighting for the positive class (Malicious) to handle class imbalance.

lightgbm Implementation

لقد تم تدريب مصنف lightgbm باستخدام كود بايثون (python code) حيث تم جلب المصنف من مكتبة lightgbm عبر التالي: import lightgbm as lgb

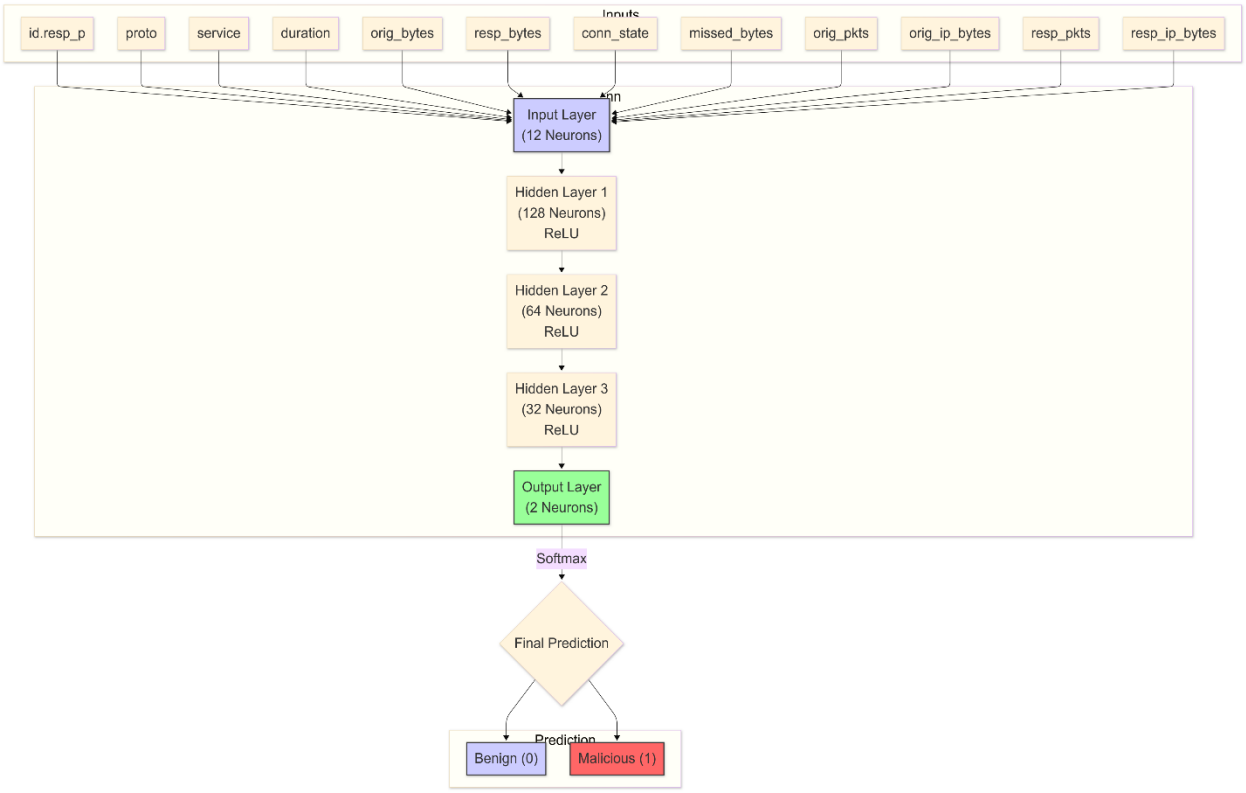
وبالطبع تم قراءة ملف البيانات combined_dataset.csv وقسمناه الى بيانات للتدريب بنسبة70% وبيانات للاختبار بنسبة 30%.

ولقد اخترنا الhyperparameters التالية:

- 'objective='binary: تم تحديد هدف النموذج على أنه تصنيف ثنائي، وهو الإعداد الصحيح للمهمة الحالية التي تتطلب التمييز بين فئتين فقط: حركة المرور "الحميدة" (Benign) وحركة المرور "الخبيثة" (Malicious).
- 'metric='auc: تم اختيار مقياس المساحة تحت منحنى ROC (Area Under the ROC Curve) لتقييم أداء النموذج. يُعد هذا المقياس خيار ممتاز لمجموعات البيانات غير المتوازنة(imbalanced data)، لأنه يقيس قدرة النموذج على الفصل والتمييز بين الفئات المختلفة، بدلاً من الاعتماد على الدقة الكلية accuracy التي قد تكون مضللة.
- n_estimators=1000 و learning_rate=0.05: تم ضبط هاتين المعلمتين معاً. حيث أن استخدام معدل تعلم منخفض (0.05) يجعل عملية التعلم أكثر تحفظ وتدرج، مما يقلل من خطر فرط التخصيص. وللتعويض عن بطء التعلم هذا، تم تحديد عدد كبير من الأشجار (1000) لضمان وصول النموذج إلى التقارب الأمثل وبناء نموذج قوي.

- num_leaves=31 و max_depth=-1: تم استخدام القيمة الافتراضية لعدد الأوراق (31)، وهي نقطة انطلاق جيدة للتحكم في تعقيد الشجرة. أما max_depth=-1 فتعني عدم فرض أي قيود على عمق الشجرة، مما يجعل num_leaves هي المتحكم الرئيسي في حجم الأشجار.
- random_state=42 و n_jobs=-1: تم تحديد حالة عشوائية ثابتة لضمان قابلية تكرار النتائج بشكل كامل، وهو أمر ضروري للمقارنات العلمية الدقيقة. كما تم استخدام جميع أنوية المعالج (n_jobs=-1) لتسريع عملية التدريب بشكل كبير، وهو أمر حيوي نظراً للحجم الكبير لمجموعة البيانات.
- colsample_bytree=0.8 و subsample=0.8: تم تطبيق هاتين التقنيتين كشكل من أشكال التنظيم (Regularization) العشوائي. حيث يتم استخدام 80% من الميزات (الأعمدة) و 80% من البيانات عشوائياً لبناء كل شجرة. هذا يُدخل العشوائية في عملية التدريب، مما يقلل من الارتباط بين الأشجار ويحسن من قدرة النموذج على التعميم.
- reg_alpha=0.1 و reg_lambda=0.1: تم تطبيق تنظيمي L1 و L2 بقيم صغيرة (0.1). تضيف هذه المعلمات عقوبة طفيفة على تعقيد النموذج، مما يساعد على منع فرط التخصيص دون تقييد قدرة النموذج على التعلم بشكل مفرط.
- scale_pos_weight=scale_pos_weight_value: هذه المعلمة حاسمة لمعالجة مشكلة عدم توازن الفئات في مجموعة بيانات IoT-23. من خلال تعيين وزن أعلى للفئة الموجبة (الخبيثةmalicious)، يتم إجبار النموذج على اعطاء اهتمام أكبر للعينات الmalicious، مما يحسن بشكل كبير من قدرته على اكتشاف التهديدات وتقليل عدد السلبيات الخاطئة (False Negatives).

الشبكة العصبونية Neural Network



Neural Network model diagram Figure 15

Neural Network hyperparameters

في الجدول التالي يوجد الhyperparameters المختارة للمصنف ويليها شرح مفصل لهم.

Table 8 Neural Network Hyperparameters

Parameter	Value	Description
Batch Size	1024	The number of samples processed before the model is updated.
Epochs	50	The maximum number of times the training dataset is passed through the model.
Loss Function	CrossEntropyLoss	Computes the cross-entropy loss between input logits and target.

Optimizer	Adam	An adaptive learning rate optimization algorithm.
Learning Rate	0.001	The initial learning rate for the Adam optimizer.
Weight Decay	1.00E-05	L2 penalty (regularization term) added to the loss function.
LR Scheduler	ReduceLROnPlateau	Reduces the learning rate when the validation loss has stopped improving.
Scheduler Patience	3	Number of epochs with no improvement after which the learning rate will be reduced.
Scheduler Factor	0.5	Factor by which the learning rate will be reduced (new_lr = lr * factor).
Early Stopping Patience	5	Number of epochs with no improvement in validation loss before training is stopped.

Neural Network Implementation

لقد تم تدريب مصنف الشبكة العصبونية باستخدام كود بايثون (python code) حيث تم جلب المصنف من مكتبة Pytorchعبر التالي: import torch.nn as nn

وبالطبع تم قراءة ملف البيانات combined_dataset.csv وقسمناه الى بيانات للتدريب بنسبة60% وبيانات للاختبار بنسبة 40%.

تم بناء النموذج، المسمى BotNetDetector، كشبكة عصبونية تسلسلية متصلة بالكامل(Fully connected neural network). تتكون من طبقة إدخال(input layer)، وثلاث طبقات مخفية(3 hidden layers)، وطبقة إخراج(output layer). ولتحسين استقرار التدريب ومنع فرط التخصيص الoverfitting، تم دمج كل من (Batch Normalization) و (Dropout) في البنية.

البنية التفصيلية لكل طبقة هي التالي:

- طبقة الإدخال: طبقة خطية تستقبل الميزات (الfeatures) الـ 12 المدخلة وتحولها إلى فضاء ذي 128 بُعداً.
- الطبقة المخفية الأولى: طبقة خطية تحول الميزات من 128 إلى 64. يتم تمرير المخرجات عبر طبقة تطبيع الدفعات، تليها دالة تنشيط ReLU وطبقة dropout.

- الطبقة المخفية الثانية: طبقة خطية تحول الميزات من 64 إلى 32. يتبعها تطبيع الدفعات، ودالة تنشيط ReLU، وطبقة تسرب dropout.
- الطبقة المخفية الثالثة: طبقة تحول من 64 إلى 32 عصبون، متبوعة بتطبيع الدفعات، ودالة ReLU، والتسرب. تعمل هذه كآخر طبقة مخفية قبل الإخراج.
- طبقة الإخراج: طبقة خطية أخيرة تحول الميزات الـ 32 إلى 2 من المخرجات، بما يتوافق مع الفئتين (حميد وخبيث). يتم تطبيق دالة Softmax ضمنياً بواسطة دالة الخسارة أثناء التدريب لتوليد الاحتمالات النهائية للفئات.

اعداد الوسطاء hyperparameters configuration

تم التحكم في عملية التدريب وسلوك النموذج من خلال مجموعة من المعلمات الفائقة:

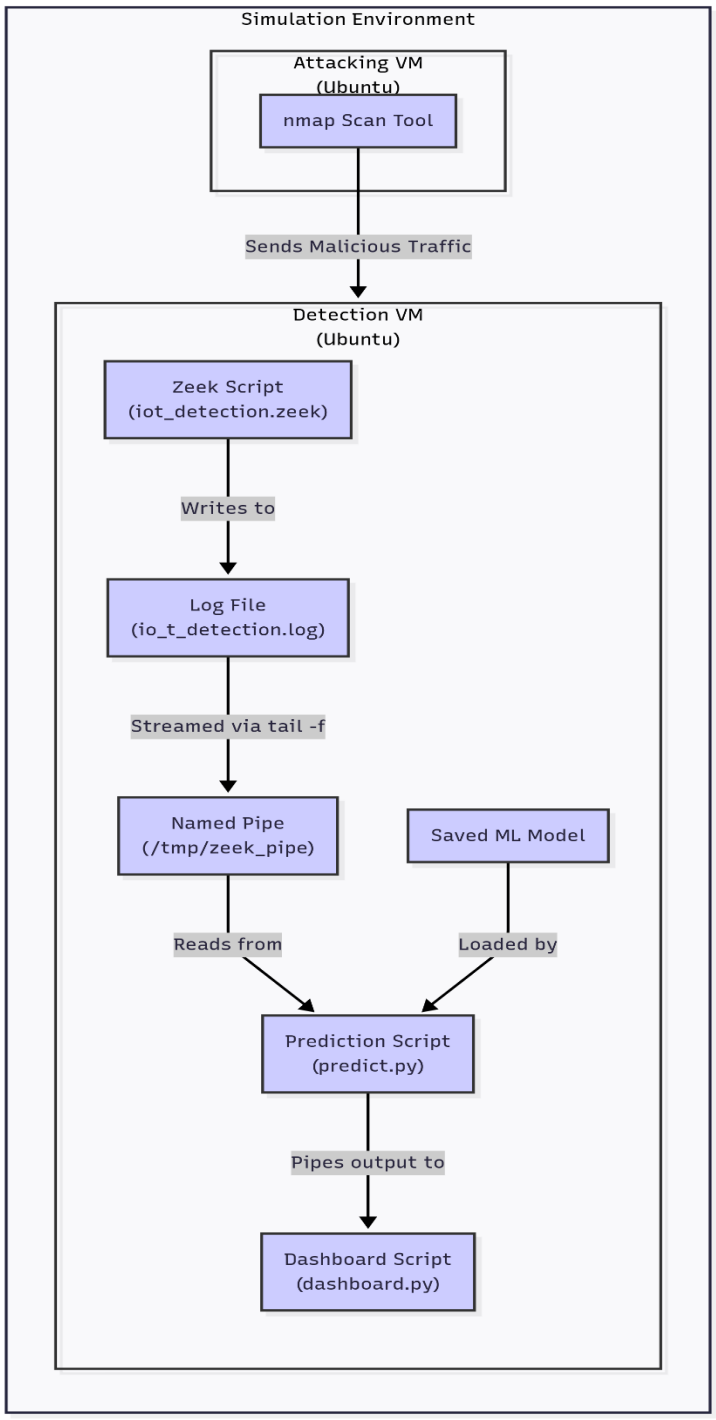
- المحسِّن : (Optimizer) تم استخدام مُحسِّن Adam، وهو خوارزمية أمثلية ذات معدل تعلم تكيفي مناسبة لمجموعة واسعة من المشاكل.
- معدل التعلم (Learning Rate): تم تحديد معدل التعلم الأولي عند 0.001.
- دالة الخسارة (Loss Function): تم اختيار خسارة الإنتروبيا المتقاطعة (nn.CrossEntropyLoss) كمعيار للخطأ. وهي دالة الخسارة القياسية لمهام التصنيف، حيث إنها فعالة في قياس أداء نموذج يُخرج احتمالات.
- حجم الدفعة (Batch Size): تم تغذية الشبكة بالبيانات على شكل دفعات صغيرة بحجم 1024 عينة. يوفر هذا الحجم توازن جيد بين الكفاءة الحسابية وتقدير التدرج المستقر.
- عدد الحقب (Number of Epochs): تم ضبط النموذج للتدريب لمدة أقصاها 50 حقبة.
- مجدول معدل التعلم (Learning Rate Scheduler): تم تطبيق مجدول من نوع ReduceLROnPlateau. يقوم هذا المجدول بمراقبة خسارة التحقق ويقلل معدل التعلم بمقدار 0.5 إذا لم تتحسن الخسارة لمدة 3 حقب(epochs) متتالية.
- التوقف المبكر (Early Stopping): لمنع فرط التخصيص وتوفير الوقت الحسابي، تم استخدام آلية التوقف المبكر مع patienceتساوي 5تم إيقاف التدريب عندما فشلت خسارة التحقق في التحسن لمدة خمس حقب متتالية.

معلمات التنظيم (Regularization)

- معدل التسرب (Dropout Rate): تم تطبيق احتمال تسرب قدره 0.3 بعد كل طبقة مخفية. هذا يعني أنه خلال كل تكرار تدريبي، تم إلغاء تنشيط 30% من العصبونات عشوائياً، مما يجبر الشبكة على تعلم ميزات أكثر قوة.
- تضاؤل الوزن (L2 Regularization): تمت إضافة قيمة تضاؤل وزن صغيرة تبلغ e-51 إلى مُحسّن Adam لتطبيق تنظيم L2 ، مما يساعد على منع أوزان النموذج من النمو بشكل كبير.

المحاكاة والنشر (Simulation for Deployment)

يلخص المخطط التالي عملية المحاكاة كاملةً ويليه التفاصيل كاملةً.



Deployment diagram Figure 16

لغرض التحقق من الجدوى العملية والأداء في الزمن الحقيقي لنماذج تعلم الآلة المدربة، تم بناء بيئة محاكاة شاملة ومتكاملة. تم اختيار نموذج LightGBM لمرحلة النشر والتوظيف، صُممت هذه البيئة لتعمل كنظام مصغر لكشف التسلل الشبكي (Network Intrusion Detection System - NIDS)، بحيث تكون قادرة على التقاط حركة مرور الشبكة وتحليلها وتصنيفها بشكل فوري.

إعداد البيئة الافتراضية والشبكة (Virtual Environment and Network Setup)

تم بناء بيئة المحاكاة باستخدام برنامج المحاكاة الافتراضية VMware Workstation، حيث قمنا باستضافة جهازين افتراضيين (Virtual Machines - VMs) منفصلين، كلاهما يعمل بنظام التشغيل Ubuntu.

- آلة الكشف (NIDS, Detection Machine): عملت هذه الالة الافتراضية كنظام الكشف، وكان عنوانها ال IP هو 192.168.32.129.
- آلة الهجوم (Attacking Machine): استُخدمت هذه الالة لتوليد حركة مرور شبكية خبيثة، وعنوانها الip هو 192.168.32.132.

تم اعداد كلا الجهازين الافتراضيين باستخدام محولات شبكة(Network Adapter) في وضع NAT (Network Address Translation). هذا الإعداد أنشأ شبكة افتراضية معزولة حيث يمكن للجهازين التواصل مباشرة مع بعضهما البعض، مع الحفاظ على إمكانية الوصول إلى الإنترنت عبر الجهاز المضيف (الhost). يحاكي هذا التكوين طوبولوجيا شبكة نموذجية في العالم الحقيقي، حيث تكون الأجهزة محمية خلف جهاز توجيه (Router).

تكوين جهاز الكشف: البنية النظامية (Detection Machine: System Architecture)

تم تصميم جهاز الكشف بخط أنابيب متعدد المكونات لمعالجة حركة مرور الشبكة في الزمن الحقيقي. تضمن سير العمل أربع مراحل، تم تنسيقها من خلال سلسلة من النصوص البرمجية وأدوات نظام لينكس Linux.

المكون الأول: التقاط حركة المرور واستخلاص الميزات (Zeek)

أساس نظام كشف التسلل هو أداة مراقبة أمان الشبكات Zeek. تم تكوين Zeek ليتنصت على واجهة الشبكة الرئيسية للجهاز الافتراضي (ens33) باستخدام نص برمجي مخصص باسم iot_detection.zeek. تم تصميم هذا النص خصيصاً ليتكامل مع خط أنابيب تعلم الآلة من خلال عدة تكوينات رئيسية:

- تسجيل JSON: ان النص البرمجي يقوم بتحميل سياسة تسجيل JSON في Zeek، مما يضمن أن تكون جميع المخرجات بتنسيق منظّم وقابل للقراءة آلياً، لتسهيل تحليلها بواسطة نصوص بايثون (python scripts) البرمجية اللاحقة.

- معالجة الاتصالات غير المكتملة: تم ضبط مهلة عدم نشاط بروتوكول TCP (tcp_inactivity_timeout) على 0sec . يعد هذا تعديل ضروري يجعل Zeek يقوم بتسجيل سجلات اتصالات TCP فوراً، حتى لو لم تكتمل المصافحة الثلاثية (Three-way Handshake). تم تصميم هذا خصيصاً لالتقاط تقنيات الاستطلاع الخفية مثل هجمات مسح TCP SYN.

- هندسة الميزات في الزمن الحقيقي: قام النص البرمجي بتعريف سجل مخصص (IoTDetection::LOG) يقوم باستخلاص الميزات الـ 12 المستخدمة بالضبط أثناء تدريب النموذج. والأهم من ذلك، أنه يقوم بنفس عملية الترميز من فئوي (categorical) إلى رقمي التي تم تطبيقها على بيانات التدريب أي يقوم بعمل encoding. على سبيل المثال، يحتوي على قوائم لتحويل أسماء البروتوكولات مثل tcp إلى 1 وحالات الاتصال مثل S0 إلى 0 أي أن القوائم تربط كل قيمة لسمه معينة برقم معين. هذا يضمن أن البيانات الملتقطة يتم تنسيقها بشكل مثالي لنموذج تعلم الآلة دون الحاجة إلى خطوة معالجة مسبقة منفصلة في بايثون، مما يزيد من كفاءة النظام.

المكون الثاني: خط أنابيب البيانات في الزمن الحقيقي (Pipe)

لتدفق البيانات من Zeek إلى نص التنبؤ البرمجي (predict.py script) في الزمن الحقيقي، تم استخدام أنبوب pipe، وهو أسلوب معتمد في نظام لينكس.

تم تحقيق ذلك باستخدام الأمر التالي الذي يعمل في terminal مخصصه له:

```
tail -f io_t_detection.log > /tmp/zeek_pipe
```

يقوم هذا الأمر بمراقبة (-f) ملف السجل (io_t_detection.log) الذي يولده Zeek باستمرار، ويعيد توجيه أي أسطر جديدة من البيانات إلى أنبوب موجود في /tmp/zeek_pipe. يعمل هذا الأنبوب كمخزّن بيانات مؤقت فوري في الذاكرة بين مستشعر Zeek وكود التنبؤ.

المكون الثالث: محرك التنبؤ (predict.py)

يقوم كود بايثون البرمجي بالتنبؤ بماهية الاتصالات فهو يقرأ باستمرار من الأنبوب المسمى /tmp/zeek_pipe، ويعالج سجل اتصال واحد في كل مرة. حيث يعمل كالتلي:

- تحميل النموذج: عند بدء التشغيل، يقوم النص البرمجي بتحميل نموذج lightgbm المدرب مسبقا والمحفوظ في (models/lightgbm.joblib) باستخدام مكتبة joblib.

- القراءة من الأنبوب: يقرأ النص البرمجي إدخالات السجل الجديدة بصيغة JSON التي تظهر في الأنبوب
- التنبؤ: لكل سجل اتصال جديد، يستخرج قيم الميزات الـ 12، ويضمن أنها بالترتيب الصحيح، ثم يدخلها إلى نموذج lightgbm المحمّل. يُخرج النموذج تنبؤاً prediction (0 للحميد، 1 للخبيث) ودرجة ثقة (احتمالية أن يكون الاتصال خبيث) (confidence score)
- الإخراج: يقوم النص البرمجي بعد ذلك بتجميع الميزات الأصلية، والتنبؤ، ودرجة الثقة(confidence score) في كائن JSON جديد ويطبعه على شاشة الterminal.

المكون الرابع: واجهة المستخدم (dashboard.py)

المكون النهائي هو لوحة معلومات سهلة القراءة تعمل في الزمن الحقيقي لعرض مخرجات محرك التنبؤ بطريقة جيدة. تم ربط النصين البرمجيين باستخدام أنبوب Shell :

```
python3 scripts/predict.py | python3 scripts/dashboard.py
```

يقرأ نص dashboard.py مخرجات JSON من predict.py ويقدم واجهة terminal يتم تحديثها باستمرار. تعرض لوحة المعلومات ما يلي:

- إحصائيات عامة: مجموع كلي للاتصالات التي تم تحليلها، مقسمة إلى حركة مرور عادية (Normal Traffic) و حركة مرور هجومية(Attack Traffic) مع أعداددهم ونسبهم المئوية.
- مخطط شريطي مرئي (Bar Chart): مخطط شريطي بسيط يمثل بصريا نسبة حركة المرور الحميدة مقابل الخبيثة.
- تفاصيل آخر هجوم: قسم مخصص يعرض الطابع الزمني، وعنوان IP المصدر والوجهة/المنفذ، وحالة الاتصال لآخر اتصال خبيث تم اكتشافه.
- ترميز لوني: لجذب انتباه المستخدم على الفور، حيث استخدمنا في الكود البرمجي ألوان لعرض جميع المعلومات المتعلقة بالهجمات باللون الأحمر وحركة المرور العادية باللون الأخضر.

تكوين جهاز الهجوم: محاكاة التهديدات (Attacking Machine: Threat Simulation)

تم استخدام الة الهجوم لتوليد حركة مرور شبكية نتحكم بها لاختبار نظام كشف التسلل.

فيما يلي المخطط التدفقي للهجوم (Attacking Flow Chart) ويليه التفاصيل كاملةً

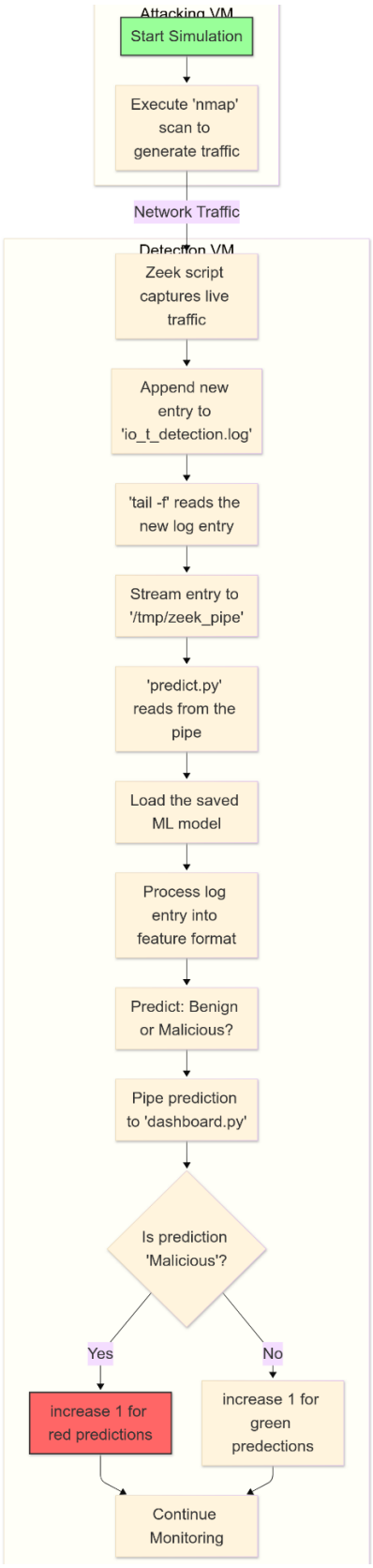


Figure 17 Attacking flow chart

توليد حركة المرور الخبيثة

تمت محاكاة النشاط الخبيث باستخدام ماسح الشبكات Nmap، وهو أداة قياسية لاستطلاع الشبكات.

حيث قمنا باستخدام الأمر التالي على terminal في الة الهجوم (attacking machine):

```
sudo nmap -sS -p 80 192.168.32.129 -n
```

- sS (مسح TCP SYN): تقوم باجراء مسح خفي (stealth) أو نصف مفتوح (half-open). يقوم بإرسال حزم SYN ولكنه لا يكمل الاتصال أبدا، وهي تقنية شائعة يستخدمها المهاجمون لاكتشاف المنافذ المفتوحة(open ports) دون أن يتم تسجيلهم بسهولة بواسطة الأنظمة التقليدية.
- p 80: يستهدف هذا المنفذ 80 فقط، وهو المنفذ القياسي لحركة مرور HTTP (الويب)، مما يجعل المسح يبدو وكأن المهاجم يبحث عن خوادم ويب.
- 192.168.32.129: هذا هو عنوان IP لجهاز الكشف (Detection machine)، مما يجعله الهدف المباشر للمسح.
- n: تعمل هذه العلامة على تعطيل تحليل أسماء النطاقات (DNS)، مما يسرع عملية المسح.

يولد هذا الأمر حجم كبير من الاتصالات بحالة S0 في سجلات Zeek، والتي تم تصميم نموذج lightgbm، المدرب على أنماط مماثلة من مجموعة بيانات IoT-23، لتحديدها على أنها استطلاع خبيث.

الخاتمة

تحدثنا في هذا الفصل عن الجهود العملية في هذا المشروع، حيث قمنا بنجاح بتصميم وتنفيذ واختبار نظام متكامل لكشف التسلل في شبكات إنترنت الأشياء. لقد أثبتت مراحل العمل، بدءاً من المعالجة الدقيقة لمجموعة بيانات IoT-23، مروراً بالتدريب المنهجي لأربعة نماذج تعلم آلة وتعلم عميق، وانتهاءً بنشر النموذج الأفضل أداءً في بيئة محاكاة واقعية، جدوى وقوة المنهجية المتبعة. أظهرت بيئة المحاكاة، التي ضمت آلة كشف وآلة هجوم، القدرة العملية للنظام على التقاط حركة مرور الشبكة، ومعالجتها، وتصنيفها بدقة وفي زمن يقترب من الزمن الفعلي، ونجح في تحديد الهجوم الذي تم شنه باستخدام أداة nmap.

بعدما تحدثنا في هذا الفصل عن النموذج المقترح والخوارزميات المستخدة والمحاكاة المطبقة، سنتطرق في الفصل التالي على عرض النتائج ومقارنتها.

الفصل الخامس

عرض ومقارنة النتائج

سنعرض في هذا الفصل نتائج الدقة النهائية لكل مصنف، وسنجري مقارنة مع الأعمال السابقة.

مقدمة

يقدم هذا الفصل تحليلاً شاملاً لنتائج أداء نماذج تعلم الآلة والتعلم العميق الأربعة التي تم تدريبها بهدف كشف حركة المرور الخبيثة في شبكات إنترنت الأشياء. يستند هذا التقييم على مجموعة بيانات الاختبار (Test Set) المستمدة من مجموعة بيانات IoT-23، والتي لم يتم استخدامها إطلاقاً أثناء مرحلة التدريب لضمان تقييم موضوعي. سنقوم باستعراض الأداء التفصيلي لكل مصنف: الغابة العشوائية(Random Forest) ، وXGBoost، وLightGBM، والشبكة العصبونية (Neural Network) وذلك بالاعتماد على طيف واسع من مقاييس التقييم القياسية. ويتمثل الهدف الأساسي في إجراء مقارنة كمية لفعالية هذه النماذج لتحديد النموذج الأكثر ملاءمة وقوة لمهمة كشف التسلل المحددة هذه.

النتائج النهائية

فيما يلي جدول يلخص كافة نتائج القياسات النهائيه التي حصلنا عليها من المصنفات الأربعة وسيليهم النتائج التقصيلية لكل مصنف.

Table 9 Evaluation metrics obtained on the trained models

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	0.9938	1	0.9911	0.9955
XGBoost	0.9931	0.9993	0.9908	0.9950
LightGBM	0.9938	0.9999	0.9912	0.9955
Neural Network	0.8487	0.8253	0.9923	0.9011

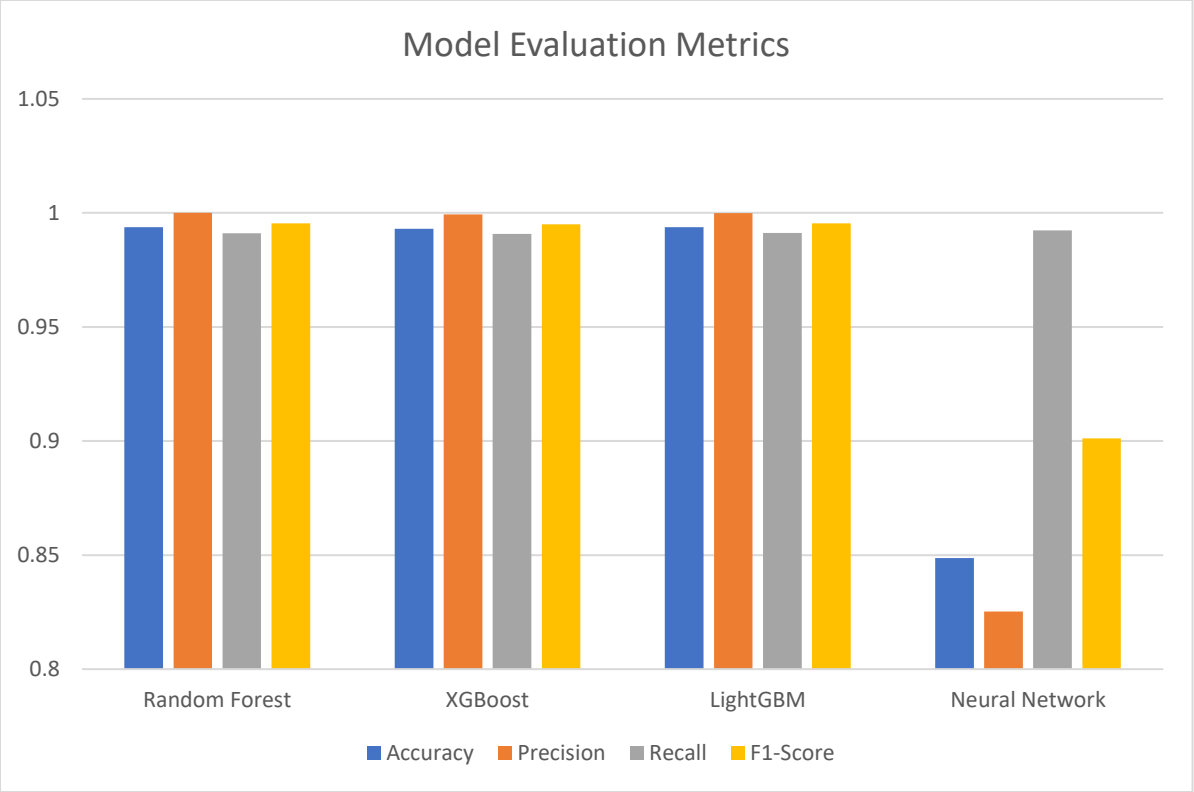


Figure 18 Clustered Bar Chart to observe models' evaluation metrics

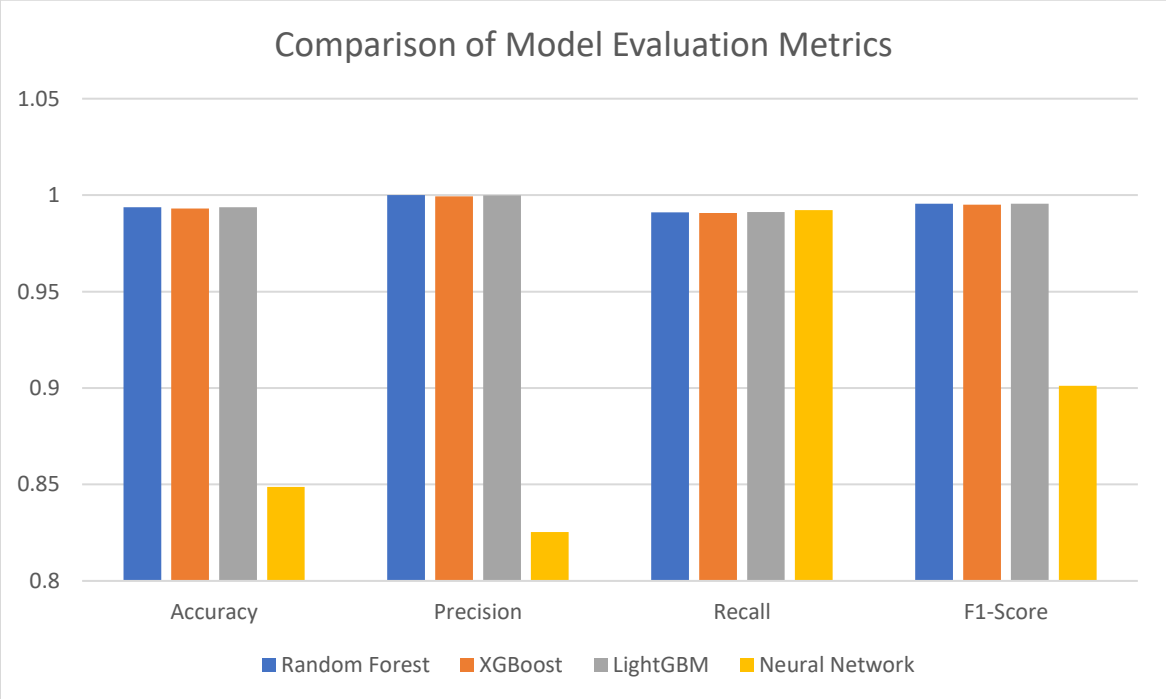


Figure 19 compare Evaluation metrics

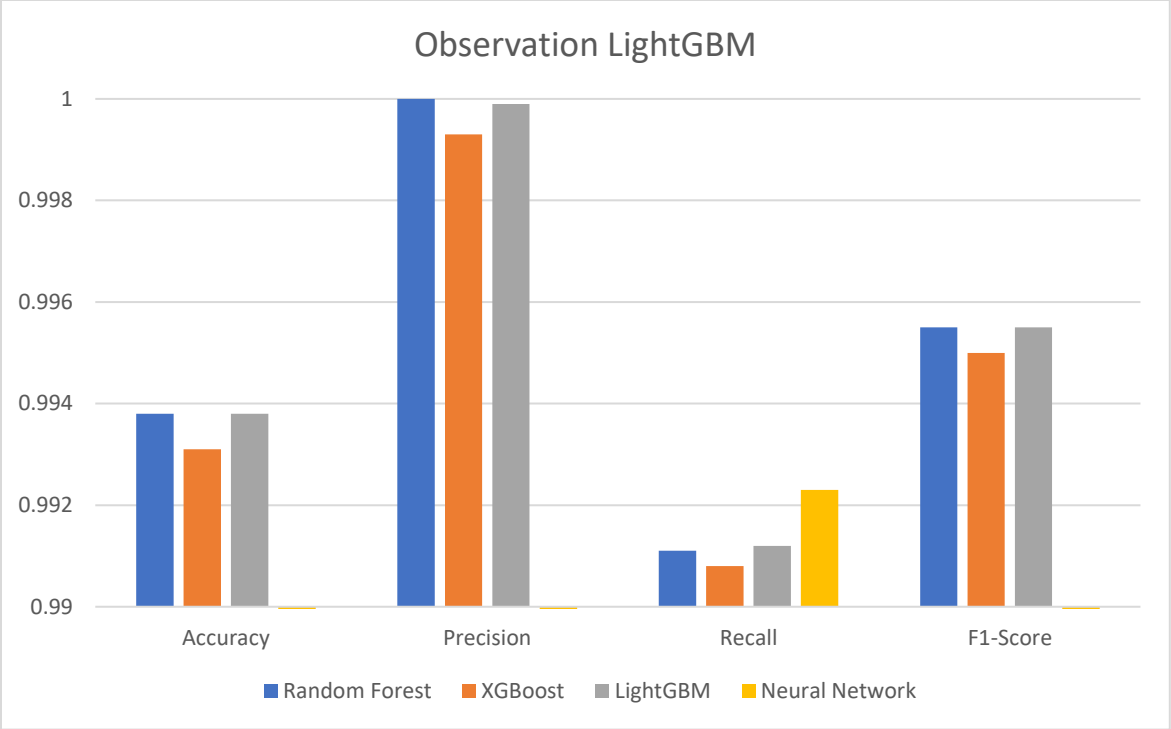


Figure 20 LightGBM priority

عرض نتائج كل نموذج بالتفصيل وتحليلها

سنقوم بعرض وتحليل الأداء التفصيلي لكل نموذج من النماذج الأربعة التي تم تدريبها، مع التركيز على تفسير النتائج المستخلصة من مقاييس التقييم ومصفوفة الارتباك الخاصة بكل مصنف.

Random forest

بعدما انتهى المصنف من عملية التدريب قمنا باختباره على بيانات الاختبار وحصلنا على النتائج التالية:

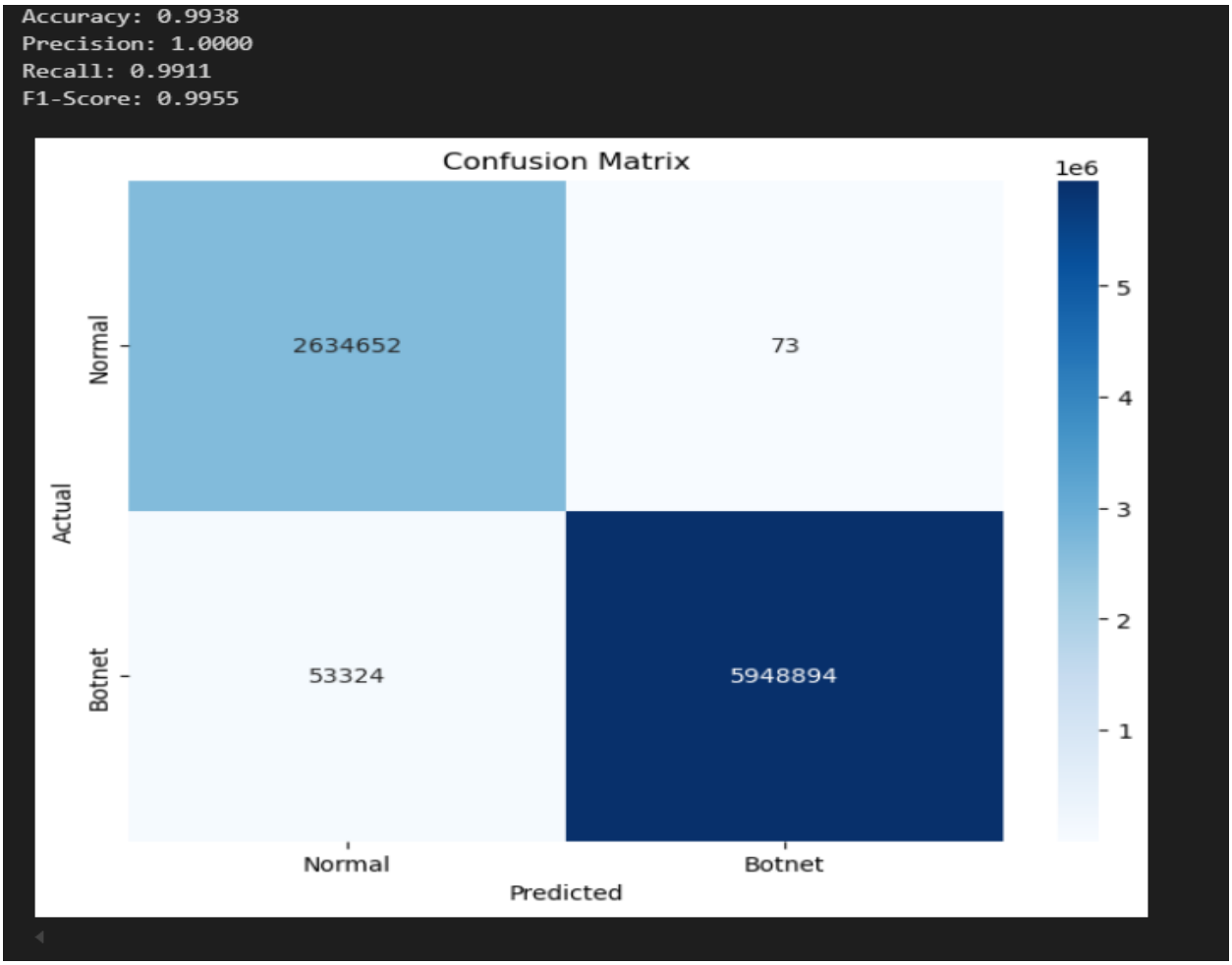


Figure21 Random Forest results

تحليل النتائج وتفسيرها:

تُظهر النتائج أن نموذج الغابة العشوائية قد حقق أداءً استثنائياً ودقة عالية جداً في مهمة تصنيف حركة مرور شبكة إنترنت الأشياء. حيث حصلنا على قيم المقاييس التالية:

- Accuracy: 0.9938 وتعني هذه القيمة أن النموذج قام بتصنيف 99.38% من إجمالي الاتصالات (سواء كانت طبيعية أو خبيثة) بشكل صحيح، وهو مؤشر ممتاز على فعالية النموذج الكلية.
- Precision: 1.0000 تعني أن النموذج حقق دقة 100% في تنبؤاته الإيجابية. بمعنى آخر، عندما يُصنف النموذج اتصالاً ما على أنه هجوم، فإنه يكون صحيحاً في كل مرة. هذا يشير إلى أن النظام لن يصدر أي إنذارات كاذبة تقريباً.

- Recall: 0.9911 تشير هذه القيمة إلى أن النموذج نجح في اكتشاف 99.11% من جميع هجمات البوت نت الفعلية الموجودة في مجموعة بيانات الاختبار. هذه نسبة اكتشاف ممتازة، وتدل على قدرة النموذج العالية على تحديد التهديدات.
- F1-Score: 0.9955 كمتوسط توافقي بين الضبط والاستدعاء، تؤكد هذه النتيجة المرتفعة جداً أن النموذج يحقق توازناً ممتازاً بين كونه دقيقاً في تنبؤاته (Precision) وشاملاً في تغطيته للتهديدات (Recall).

تحليل مصفوفة الارتباك (Confusion Matrix):

- الإيجابيات الحقيقية 5,948,894 :(True Positives - TP) أي نجح النموذج في تحديد 5,948,894 اتصال خبيث بشكل صحيح.
- السلبيات الحقيقية 2,634,652 :(True Negatives - TN)أي نجح النموذج في تحديد أكثر من 2,634,652 مليون اتصال طبيعي بشكل صحيح.
- الإيجابيات الخاطئة 73 :(False Positives - FP) هذا الرقم المنخفض للغاية هو السبب وراء درجة الضبط المثالية (Precision = 1.0). لقد أخطأ النموذج وصنف 73 اتصالاً طبيعياً فقط على أنها هجمات، وهو عدد ضئيل جداً مقارنة بملايين التصنيفات الصحيحة.
- السلبيات الخاطئة 53,324 :(False Negatives - FN) هذه هي نقطة الضعف الأهم في النموذج. على الرغم من أن نسبة الاستدعاء (Recall) عالية، إلا أن النموذج فشل في اكتشاف 53,324 هجوماً فعلياً، وصنفها بالخطأ على أنها حركة مرور طبيعية. في بيئة تشغيل حقيقية، يمثل هذا العدد من الهجمات الفائتة خطراً أمنياً يجب أخذه بعين الاعتبار.

XGBoost

بعدما انتهى المصنف من عملية التدريب قمنا باختباره على بيانات الاختبار حصلنا على النتائج التالية:

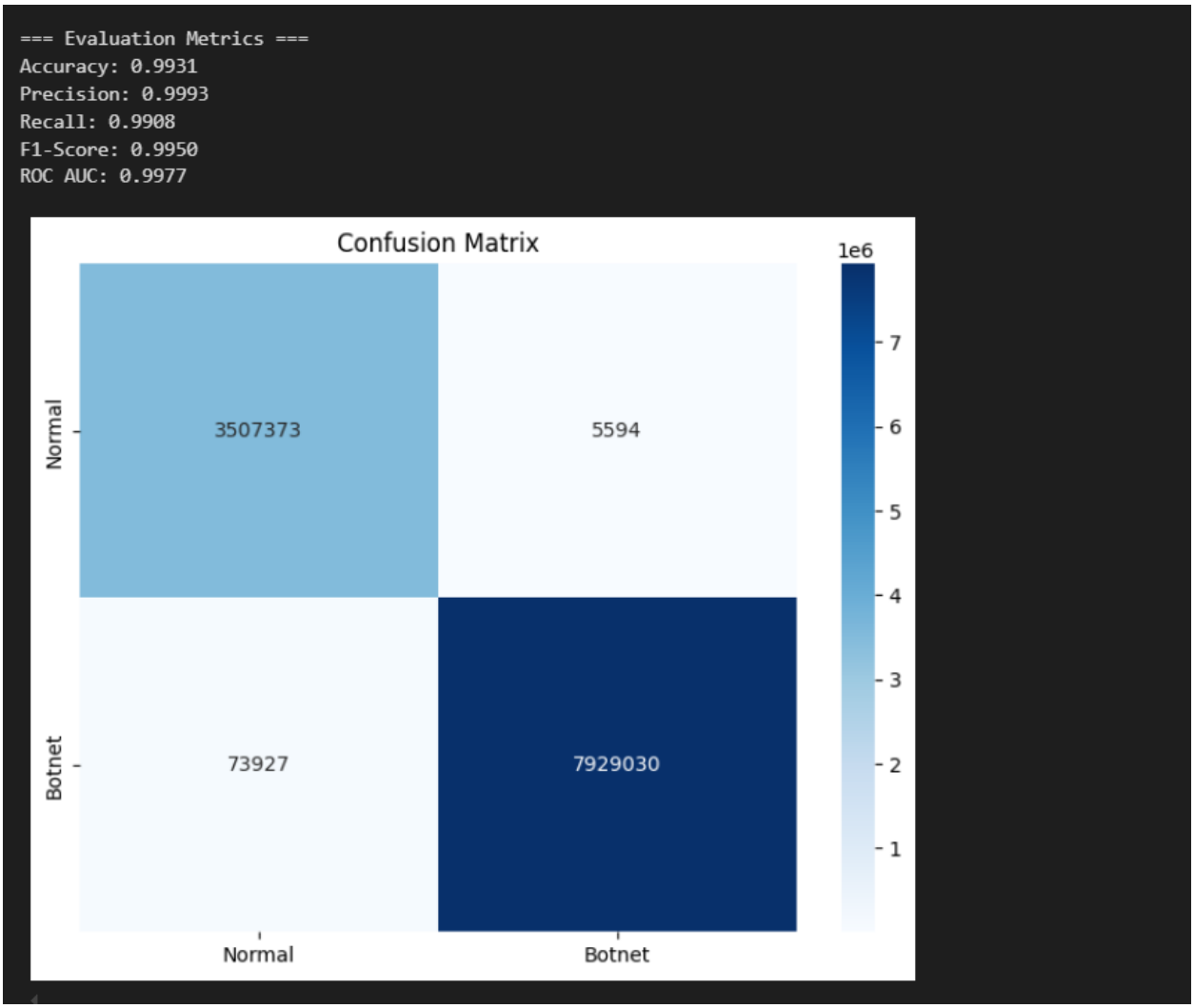


Figure22 XGBoost results

تحليل النتائج وتفسيرها

يُظهر نموذج XGBoost أداء قوي. حيث حصلنا على قيم المقاييس التالية:

- Accuracy: 0.9931 أي حقق النموذج دقة إجمالية تبلغ 99.31%، مما يعني أنه نجح في تصنيف الغالبية العظمى من الاتصالات بشكل صحيح.

- Precision: 0.9993 أي أن النموذج يتمتع بموثوقية استثنائية عند إطلاقه لتنبيه بوجود هجوم. عملياً، هذا يعني أن 99.93% من الاتصالات التي يصنفها النموذج على أنها هجوم هي بالفعل كذلك، مما يؤدي إلى عدد قليل جداً من الإنذارات الكاذبة.
- Recall: 0.9908 أي تمكّن النموذج من اكتشاف 99.08% من جميع هجمات البوت نت الفعلية في مجموعة الاختبار. هذه النسبة المرتفعة تدل على فعالية النموذج في التعرف على التهديدات وعدم مرورها بدون كشف.
- F1-Score: 0.9950 تؤكد هذه النتيجة الممتازة وجود توازن قوي بين الضبط والاستدعاء، مما يعني أن النموذج فعال في تقليل كلا النوعين من الأخطاء (الإيجابيات الخاطئة والسلبيات الخاطئة).
- ROC AUC: 0.9977 وهي تشير إلى أن النموذج يمتلك قدرة تمييزية شبه مثالية بين حركة المرور الحميدة والخبيثة عبر جميع عتبات التصنيف الممكنة.

تحليل مصفوفة الارتباك (Confusion Matrix):

- الإيجابيات الحقيقية 7,929,030 و السلبيات الحقيقية 3,507,373 أي أن النموذج قادر على تصنيف كل من الهجمات وحركة المرور الطبيعية بشكل ممتاز.
- الإيجابيات الخاطئة 5,594 أي أنه أخطأ النموذج في تصنيف 5,594 اتصالاً طبيعياً على أنها هجمات.
- السلبيات الخاطئة 73,927 هذه هي نقطة الضعف الرئيسية للنموذج. لقد فشل النموذج في اكتشاف 73,927 هجوماً فعلياً، وصنفها على أنها طبيعية. يمثل هذا العدد من الهجمات الفائتة الخطر الأمني الأكبر الذي يقدمه هذا النموذج، وهو أعلى بشكل ملحوظ من عدد السلبيات الخاطئة في نموذج الغابة العشوائية.

LightGBM

بعدما انتهى المصنف من عملية التدريب قمنا باختباره على بيانات الاختبار حصلنا على النتائج التالية:

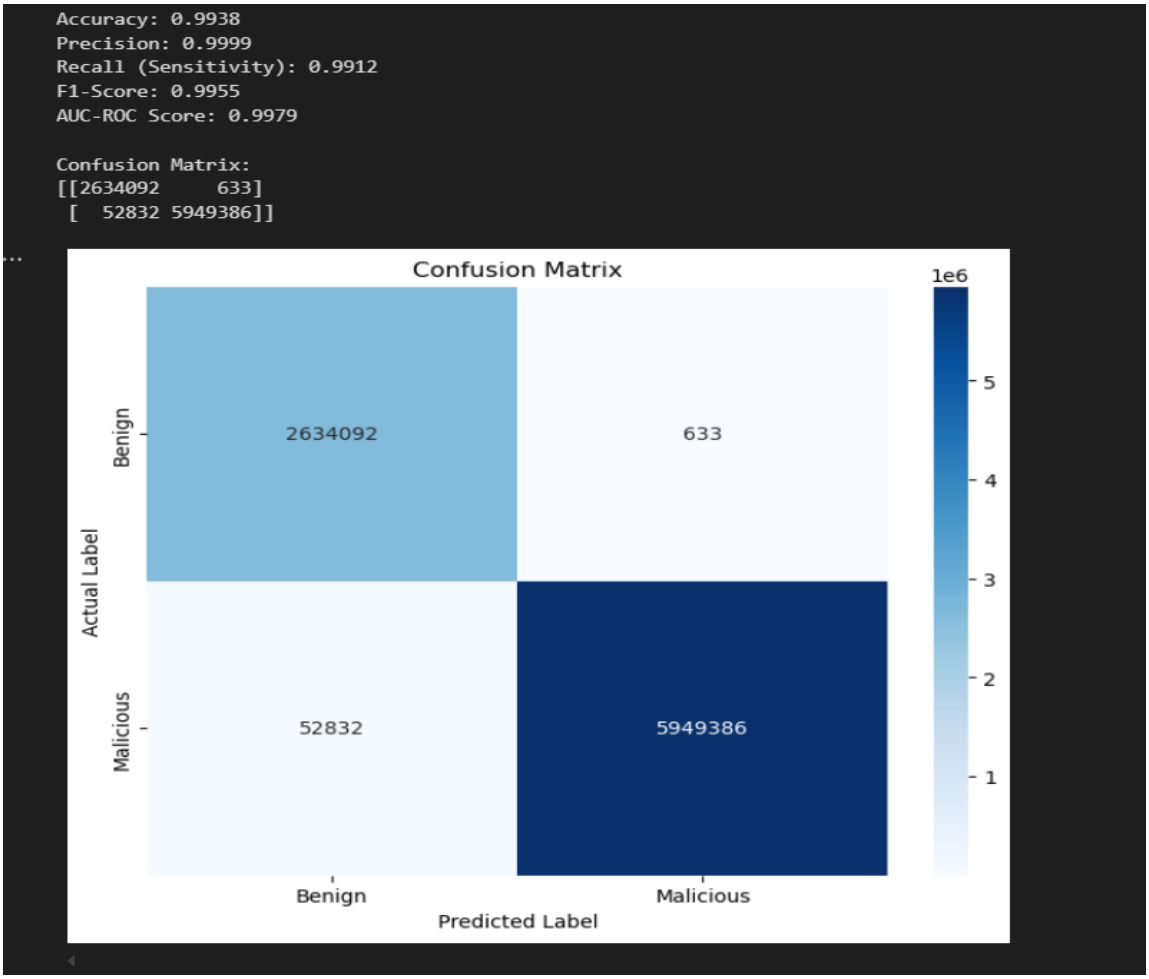


Figure23 LightGBM results

تحليل النتائج وتفسيرها

يُظهر نموذج LightGBM أداءً فائقاً يضعه في مقدمة المصنفات التي تم اختبارها، مما يجعله مرشحاً قوياً للاستخدام في نظام الكشف. حيث حصلنا على قيم المقاييس التالية:

- Accuracy: 0.9938 أي يثبت النموذج قدرته الممتازة على تصنيف معظم الاتصالات بشكل صحيح، وهي نفس دقة نموذج الغابة العشوائية.
- Precision: 0.9999 أي أنّ هذا النموذج موثوقاً للغاية. عملياً، هذا يعني أن احتمال أن يكون التنبيه بوجود هجوم إنذاراً كاذباً هو احتمال ضئيل جداً (حوالي 1 في 10,000).

- Recall: 0.9912 أي ينجح النموذج في اكتشاف الغالبية العظمى من الهجمات الفعلية. هذه النتيجة تتطابق تقريباً مع نتيجة الغابة العشوائية وتتفوق على XGBoost، مما يشير إلى حساسية عالية للتهديدات.
- F1-Score: 0.9955 هذه النتيجة المرتفعة، والمطابقة لنموذج الغابة العشوائية، تؤكد أن النموذج يحقق توازناً مثالياً بين تقليل الإنذارات الكاذبة (Precision) وتعظيم اكتشاف الهجمات الحقيقية (Recall).

تحليل مصفوفة الارتباك (Confusion Matrix):

- الإيجابيات الحقيقية 5,949,386 و السلبيات الحقيقية 2,634,092 تُظهر الأرقام المرتفعة جداً في هذه الخانات أن النموذج ممتاز في تحديد كلتا الفئتين بشكل صحيح.
- الإيجابيات الخاطئة 633 هذا العدد المنخفض للغاية من الإنذارات الكاذبة هو ما يمنح النموذج درجة ضبط شبه مثالية.
- السلبيات الخاطئة 52,832 أي فشل النموذج في اكتشاف ما يقرب من 53 ألف هجوم فعلي. هذا العدد أقل من العدد الخاص بنموذج الغابة العشوائية ولكنه أقل بكثير من نموذج XGBoost.

Neural Network

بعدما انتهى المصنف من عملية التدريب قمنا باختباره على بيانات الاختبار حصلنا على النتائج التالية:

```
=== Test Set Performance ===
Accuracy: 0.8487
Precision: 0.8253
Recall: 0.9923
F1-Score: 0.9011
```

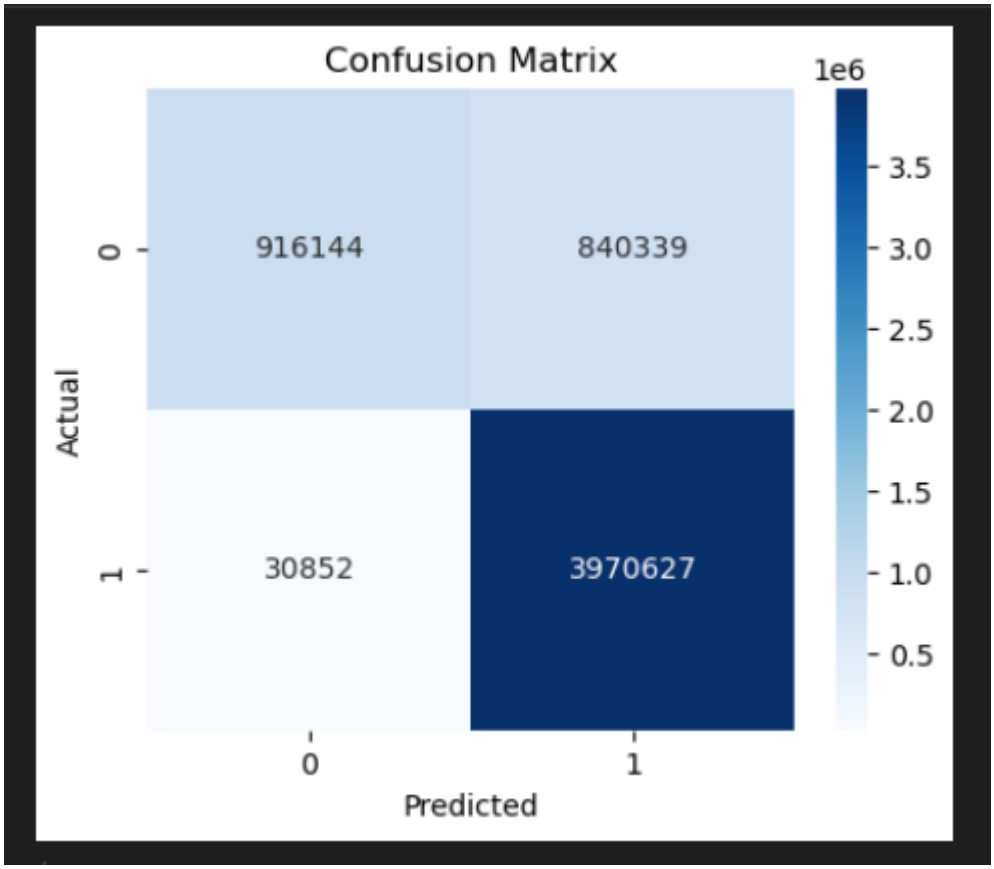


Figure24 Neural Network results

تحليل النتائج وتفسيرها

يُظهر نموذج الشبكة العصبونية نمط أداء مختلف جذرياً وأقل توازناً بكثير مقارنة بالنماذج القائمة على الأشجار (Tree-based models). على الرغم من أنه يتفوق في جانب واحد، إلا أنه يظهر ضعفاً كبيراً في جانب آخر، مما يجعله أقل ملاءمة لهذه المهمة. حيث حصلنا على قيم المقاييس التالية:

- Accuracy: 0.8487 تعتبر هذه الدقة جيدة بشكل عام، ولكنها أقل بكثير من دقة الـ 99% التي حققتها النماذج الثلاثة الأخرى، مما يشير إلى عدد كبير من التصنيفات الخاطئة.
- Precision: 0.8253 هذه هي نقطة الضعف الكبرى للنموذج. درجة ضبط تبلغ 82.53% فقط تعني أن هناك نسبة كبيرة من الإنذارات الكاذبة. عملياً، هذا يعني أنه من بين كل 100 تنبيه يصدره النظام بوجود هجوم، هناك حوالي 17 تنبيهاً خاطئاً. هذا المعدل المرتفع من الإنذارات الكاذبة يجعل النظام غير عملي.
- Recall: 0.9923 هذه هي نقطة القوة الوحيدة للنموذج. بدرجة استدعاء تبلغ 99.23%، فإن هذا النموذج هو جيد في اكتشاف الهجمات الفعلية،
- F1-Score: 0.9011 هذه النتيجة، على الرغم من أنها جيدة، إلا أنها الأدنى بين جميع النماذج، مما يؤكد الاختلال الكبير في التوازن بين الضبط (Precision) والاستدعاء (Recall).

تحليل مصفوفة الارتباك (Confusion Matrix)

- الإيجابيات الحقيقية 3,970,627 و السلبيات الحقيقية 916,144 أي قام النموذج بتصنيف أعداد كبيرة من الحالات بشكل صحيح، ولكنه أقل فعالية في تحديد الحالات الطبيعية (TN) مقارنة بالنماذج الأخرى.
- الإيجابيات الخاطئة 840,339 هذا الرقم هو السبب الرئيسي في ضعف النموذج. لقد أخطأ النموذج في تصنيف أكثر من 840 ألف اتصال طبيعي على أنها هجمات.
- السلبيات الخاطئة 30,852 هذا هو أقل عدد من الهجمات الفائتة بين جميع النماذج التي تم اختبارها. يشير هذا إلى أن النموذج يميل إلى الإفراط في الشك، حيث يفضل تصنيف حالة طبيعية على أنها هجوم على أن يفوت هجوماً حقيقياً.

المقارنة

بعد إجراء تحليل مقارن وشامل لنتائج الأداء للمصنفات الأربعة، تم اختيار نموذج LightGBM ليكون النموذج المعتمد في بيئة المحاكاة والنشر في الزمن الفعلي. على الرغم من أن نموذج الغابة العشوائية قدم أداءً مطابقاً تقريباً من حيث مقاييس الدقة (F1-Score)، إلا أن LightGBM أظهرت تفوق طفيف في مقياس الاستدعاء (Recall)، مما يعني أنه الأقدر على اكتشاف أكبر عدد ممكن من الهجمات الفعلية وتقليل عدد السلبيات الخاطئة (False Negatives)، وهو هدف أي نظام كشف تسلل. والأهم من ذلك، وبالنظر إلى متطلبات الأداء في الزمن الفعلي، فإن نموذج LightGBM يتميز بكفاءته الحسابية العالية وسرعته الفائقة في التدريب والتنبؤ مقارنة بالنماذج الأخرى، وهو ما يجعله الخيار الأمثل لنظام يجب أن يتعامل مع حركة مرور الشبكة بشكل فوري.

مقارنة مع اعمال سابقة

(Michael Austin, 2021 – IoT Malicious Traffic Classification Using Machine Learning)

هذه الدراسة أيضا اختارت iot23 للتدريب والاختبار على 4 نماذج مختلفة

لقد قام بعمل subsampling لمجموعة البيانات لكي يوازن بين الصفوف الحميدة والخبيثة ووصل الى فقط 1,027,714 مدخل. وأصبحت الصفوف متوازنه. وقام بscaling للبيانات عبر standardScaler() من مكتبة Scikit-Learn

وقام بتدريب 4 نماذج تعلم الة مختلفة وهي:

- Random forest: باستخدام 50 شجرة قرار كل واحده بعمق 6 عقد.
- Naïve Baes: واستخدم الGaussian naïve bayes
- Support Vector Machine: واستخدم linear svm أي استخدم تابع خطي للفصل بين الصفوف
- Stochastic Gradient Descent: مصنف خطي تم تحسينه باستخدام SGD

وقد قام باعتبار الخبيث 1 والحميد 0، وقام باستخدام السمات التي هي مغمقة bold التالية:

Table 3.1: Zeek conn.log features (bolded features used in this problem report)

Feature	Data Type	Description
ts	time	Timestamp in UNIX epoch format
uid	string	Unique ID of Connection
id.orig_h	string	Originating endpoint's IP address (AKA ORIG)
id.orig_p	integer	Originating endpoint's TCP/UDP port (or ICMP code)
id.resp_h	addr	Responding endpoint's IP address (AKA RESP)
id.resp_p	integer	Responding endpoint's TCP/UDP port (or ICMP code)
proto	string	Transport layer protocol of connection
service	string	Dynamically detected application protocol, if any
duration	integer	Time of last packet seen – time of first packet seen
orig_bytes	integer	Originator payload bytes; from sequence numbers if TCP
resp_bytes	integer	Responder payload bytes; from sequence numbers if TCP
conn_state	string	Connection state (see conn.log:conn_state table)
local_orig	bool	If conn originated locally T; if remotely F. If Site::local_nets empty, always unset.
missed_bytes	integer	Number of missing bytes in content gaps
history	string	Connection state history (see conn.log:history table)
orig_pkts	integer	Number of ORIG packets
orig_ip_bytes	integer	Number of ORIG IP bytes (via IP total_length header field)
resp_pkts	integer	Number of RESP packets
resp_ip_bytes	integer	Number of RESP IP bytes (via IP total_length header field)
tunnel_parents	set	If tunneled, connection UID of encapsulating parent (s)

Figure25 features used by Austin

وقام بتحويل سمة الtimestamp الى time of day وهذا غير جيد بالطبع وسيؤدي الى overfitting

وحصل على النتائج التالية:

Table 10 results with Michael Austin

Model	Accuracy	F1-Score	Precision	Recall
Random Forest	97.45	97.39%	99.92%	94.98%
Naïve Bayes	94.19%	94.50%	89.75%	99.75%
Linear SVM	91.72%	92.33%	85.85%	99.90%
Stochastic Gradient Descent	94.45%	94.41%	95.33%	93.51%

[26]

ونلاحظ تفوق النماذج التي دربناها على النماذج الخاصه به حيث يكفي مقارنة F1-Score وRecall لمعرفة الأفضل:

Table 11 comparing results with Austin

Model	Recall	F1-Score	Model	Recall	F1-Score
Random Forest	0.9911	0.9955	Random Forest	94.98%	97.39%
XGBoost	0.9908	0.9950	Naïve Bayes	99.75%	94.50%
LightGBM	0.9912	0.9955	Linear SVM	99.90%	92.33%
Neural Network	0.9923	0.9011	Stochastic Gradient Descent	93.51%	94.41%

الجدول السابق يعرض مقارنة بين نتائج الدقه التي حصلنا عليها ونتائج الدقت التي حصل عليها Austin حيث أن أول عمودين عاليسار يمثلون نتائجنا.

A Deep Learning Ensemble for Network Anomaly and Cyber-Attack Detection

[27]

قام Vibekananda Dutta، Michał Choras، Marek Pawlicki وRafał Kozik بنشر ورقة بحثية في عام 2020.

تقدم هذه الورقة البحثية طريقة تجميعية متطورة للتعلم العميق مكونة من خطوتين، مصممة لكشف الشذوذ في الشبكات والهجمات السيبرانية بكفاءة أعلى من أساليب تعلم الآلة التقليدية عند التعامل مع حركة مرور الشبكة واسعة النطاق.

- هندسة السمات (Feature Engineering): في الخطوة الأولى، تستخدم الطريقة مشفراً تلقائياً عميقاً (Deep Sparse AutoEncoder - DSAE) لهندسة واختيار أهم السمات من بيانات الشبكة الأولية وتقليل أبعادها. تساعد هذه العملية في خفض التعقيد الحسابي وتحسين كفاءة المصنف.
- التصنيف (Classification): في الخطوة الثانية، يتم استخدام نهج التعميم المكدس (stacked generalization). تتكون النماذج الأساسية (المستوى-0) من شبكة عصبونية عميقة (DNN) وشبكة الذاكرة طويلة قصيرة المدى (LSTM). بعد ذلك، يتم تغذية تنبؤات هذين النموذجين إلى مصنف وصفي (meta-classifier) أبسط (وهو الانحدار اللوجستي)، الذي يقوم بإصدار التنبؤ النهائي حول ما إذا كانت حركة المرور تمثل شذوذاً أم لا.

وقد تم التحقق من فعالية النموذج المقترح على ثلاث مجموعات بيانات مختلفة، بما في ذلك مجموعة بيانات IoT-23. أظهرت النتائج أن النهج التجميعي المكدس قد تفوق بشكل كبير على المصنفات الفردية الحديثة الأخرى مثل الغابة العشوائية (Random Forest).

Table 12 comparing results

Model	Recall	F1-Score	Method	Recall	F1-Score
Random Forest	0.9911	0.9955	Random Forest	0.89	0.896
XGBoost	0.9908	0.9950	DNN	0.89	0.87
LightGBM	0.9912	0.9955	LSTM	0.92	0.95
Neural Network	0.9923	0.9011	Stacked (proposed)	0.95	0.98

ونلاحظ تفوق نموذجنا lightgbm.

الخاتمة

قمنا بعرض وتحليل نتائج الأداء التفصيلية للمصنفات الأربعة بشكل منهجي، مما قدم رؤية واضحة حول فعالية كل نموذج. أظهرت النتائج تفوقاً كبيراً للنماذج القائمة على الأشجار(Random Forest, XGBoost, LightGBM) ، التي حققت جميعها مقاييس أداء تتجاوز 99%، بينما أظهر نموذج الشبكة العصبونية أداءً غير متوازن لا يتناسب مع متطلبات النظام العملية.

وبناءً على المقارنة الكمية، برز نموذج **LightGBM** كأفضل مصنف لهذه المهمة، ليس فقط لتحقيقه أعلى مقاييس-F1 Scoreوتوازناً مثالياً بين الضبط (Precision) والاستدعاء(Recall) ، بل أيضاً بفضل كفاءته الحسابية العالية التي تجعله الخيار الأمثل للتطبيق في الزمن الفعلي. والجدير بالذكر أن النماذج التي تم تطويرها في هذا المشروع أثبتت تفوقها على النتائج المنشورة في الدراسات السابقة التي تمت مقارنتها، مما يؤكد قوة المنهجية المتبعة في معالجة البيانات واختيار المعلمات الفائقة. تؤسس هذه النتائج القوية والمثبتة أساساً متيناً للخلاصة النهائية للمشروع وتوصياته المستقبلية.

الخاتمة والافاق المستقبلية

يختتم هذا المشروع رحلة متكاملة بدأت من تحديد التحديات الأمنية الحرجة في شبكات إنترنت الأشياء، وانتهت ببناء وتشغيل نظام فعال لكشف التسلل يعتمد على تعلم الآلة، حيث تم بنجاح تصميم وتنفيذ وتقييم خط أنابيب كامل، بدءاً من المعالجة الدقيقة لمجموعة بيانات IoT-23 الواقعية، مروراً بالتدريب المنهجي لأربعة نماذج تصنيف مختلفة، وصولاً إلى اختيار نموذج LightGBM كالحل الأمثل الذي يوازن بين الدقة الفائقة التي تجاوزت 99% والكفاءة الحسابية. إن الإنجاز الأبرز لهذا العمل لا يكمن فقط في تحقيق نتائج تتفوق على دراسات سابقة، بل في إثبات الجدوى العملية للمنهجية عبر بناء بيئة محاكاة متكاملة، نجحت في كشف هجوم استطلاعي في الزمن الفعلي، مما يؤكد بقوة على أن توظيف تعلم الآلة ليس مجرد إمكانية نظرية، بل هو حل عملي وقوي يمكن الاعتماد عليه لبناء جيل جديد من أنظمة الدفاع الذكية والاستباقية لتأمين منظومات إنترنت الأشياء. وبناءً على هذا الأساس المتين الذي تم إرساءه، يمكن اقتراح عدة مسارات للعمل المستقبلي بهدف زيادة تطوير النظام وتعزيز قدراته، وفي مقدمتها الانتقال إلى التصنيف متعدد الفئات (Multi-class Classification) لاستغلال التصنيفات التفصيلية في مجموعة بيانات IoT-23 وتدريب نموذج قادر على تحديد نوع الهجوم المحدد، مما يوفر معلومات أكثر قيمة للمسؤولين. كما يمكن تعزيز موثوقية النظام عبر اختباره ضد مجموعة أوسع وأكثر تعقيداً من الهجمات، مثل هجمات DDoS الفعلية وتسريب البيانات. ومن الناحية العملية، يمكن التركيز على تحسين أداء النموذج للأجهزة محدودة الموارد باستخدام تقنيات مثل التكميم (Quantization) لتقليل متطلباته الحاسوبية وجعله مناسباً للنشر مباشرة على بوابات إنترنت الأشياء. وأخيراً، يمكن تطوير واجهة مستخدم رسومية (GUI) قائمة على الويب لاستبدال لوحة المعلومات النصية الحالية، مما يوفر عرضاً أكثر تفاعلية وسهولة للبيانات والتنبيهات في بيئة تشغيل فعلية.

الملاحق

الملحق آ

Pseudocode ترميز

Start

Read conn.log.labeled files

For each file extract data from it and save it in a .csv format

Then for each .csv dataset file do:

Drop the features { 'Unnamed: 0', 'ts', 'uid', 'local_orig', 'local_resp', 'id.orig_h', 'id.resp_h', 'id.orig_p', 'history' }

Do Label encoding: 0 for benign, 1 for malicious

convert 'duration', 'orig_bytes', 'resp_bytes' to numeric types

handle nan values for 'duration', 'orig_bytes', 'resp_bytes' features:

calculate medians for 'duration', 'orig_bytes', 'resp_bytes' from non-S0 connections

FOR each row in the DataFrame:

IF 'conn_state' is 'S0' AND 'duration' is missing:

duration' <= 0

ELSE IF 'conn_state' is NOT 'S0' AND 'duration' is missing:

duration' <= calculated median

FOR each row in the DataFrame:

IF 'conn_state' is 'S0' AND 'orig_bytes' is missing:

orig_bytes' <= 0

ELSE IF 'conn_state' is NOT 'S0' AND 'orig_bytes ' is missing:

orig_bytes ' <= calculated median

FOR each row in the DataFrame:

IF 'conn_state' is 'S0' AND ‘resp_bytes ' is missing:

resp_bytes ' <= 0

ELSE IF 'conn_state' is NOT 'S0' AND ' resp_bytes ' is missing:

resp_bytes ' <= calculated median

Combine all cleaned_datasets.csv files to one fille named combined_dataset.csv

Select X features without the label and Y is for the Label

Split into training and testing

For each model from the models=[random_forest, xgboost, lightgbm, neural_network]do:

Put the corresponding hyperparameters

Train on training data

Evaluate on testing data with many metrics like [precesion, recall, f1-score, AUC]

Save the model to [.pkl or .pth pr .joblib format

End

References

[1] "IEEE Internet of Things Journal," *IEEE Internet of Things Journal.*

[2] Cisco, The Internet of Things Reference Model, Cisco, 2014.

[3] N. –. N. I. o. S. a. Technology, "IoT Device Cybersecurity Capability Core Baseline (NISTIR 8259A)," NIST – National Institute of Standards and Technology, 2020.

[4] J. P. Walters, Z. Liang, W. Shi and V. Chaudhary, "A survey of security attacks in wireless sensor networks," *Computer Networks,* vol. 51, no. 11, pp. 3000-3023, 2007.

[5] M. Antonakakis, T. April, M. Bailey and e. al., "Dissecting the Mirai Botnet," in *26th USENIX Security Symposium (USENIX Security '17)*, 2017.

[6] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, "A Survey of Botnet Technology, Detection and Defenses," *IEEE Communications Surveys & Tutorials,* vol. 17, no. 2, pp. 824-844, 2015.

[7] M. Antonakakis, T. April, M. Bailey and e. al., "Dissecting the Mirai Botnet," in *26th USENIX Security Symposium (USENIX Security '17)*, 2017.

[8] C. a. I. S. A. (CISA), "nderstanding Denial-of-Service Attacks," 27 October 2022. [Online]. Available: https://www.cisa.gov/news-events/news/understanding-denial-service-attacks. [Accessed 6 August 2025].

[9] M. Antonakakis and e. al., "Dissecting the Mirai Botnet," in *26th USENIX Security Symposium (USENIX Security '17)*, Vancouver, 2017.

[10] T. Liu and e. al., "A Multi-Faceted Analysis of Satori, a Mirai-like IoT Botnet," in *Proceedings of the Internet Measurement Conference 2018*, 2018.

[11] A. T. Labs, "New Torii botnet is a master of stealth," 27 September 2018. [Online]. Available: https://blog.avast.com/new-torii-botnet-threat-research. [Accessed 6 August 2025].

[12] S. Sicari, A. Rizzardi, L. A. Grieco and A. Coen-Porisini, "Security, privacy, and trust in IoT: A survey," *Computer Networks,* vol. 76, pp. 190-211, 2015.

[13] K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," National Institute of Standards and Technology, Gaithersburg, MD, 2007.

[14] K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," National Institute of Standards and Technology, Gaithersburg, MD, 2007.

[15] C. M. Bishop, Pattern Recognition and Machine Learning, New York: Springer, 2006.

[16] T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, New York: Springer, 2009.

[17] T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed., New York: Springer, 2009.

[18] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow, 3rd Edition, Sebastopol, CA: O'Reilly Media, 2022.

[19] C. M. Bishop, Pattern Recognition and Machine Learning, New York, NY: Springer, 2006.

[20] L. Breiman, "Random forests," *Machine Learning,* pp. 5-32, 2001.

[21] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, 2016.

[22] G. M. Q. F. T. W. T. C. W. M. W. .. &. L. T. Y. Ke, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, Red Hook, NY, USA, 2017.

[23] Microsoft and LightGBM Contributors, "Parameters," 4 August 2025. [Online]. Available: https://lightgbm.readthedocs.io/en/latest/Parameters.html.

[24] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, Cambridge, MA: MIT Press, 2016.

[25] T. S.-l. developers, "Model evaluation: quantifying the quality of predictions," 2025. [Online]. Available: https://scikit-learn.org/stable/modules/model_evaluation.html. [Accessed 8 August 2025].

[26] M. Austin, "IoT Malicious Traffic Classification Using Machine Learning," 2021.

[27] V. Dutta, M. Choraś, M. Pawlicki and R. Kozik, "A Deep Learning Ensemble for Network Anomaly and Cyber-Attack Detection," *Sensors,* vol. 20, no. 16, p. 4583, 2020.

[28] E. –. E. U. A. f. Cybersecurity, "Threat Landscape [2024]," ENISA – European Union Agency for Cybersecurity, 2024.

[29] T. S.-l. developers, "Model evaluation: quantifying the quality of predictions," 2025. [Online]. [Accessed 7 August 2025].

الخلاصة

مع الانتشار المتسارع لشبكات إنترنت الأشياء (IoT) وما يصاحبها من تحديات أمنية، أصبحت هذه الأجهزة هدفاً رئيسياً لتشكيل الروبوتات الشبكية (Botnets) التي تُستخدم لشن هجمات واسعة النطاق. يعالج هذا المشروع هذه المشكلة من خلال تصميم وتنفيذ وتقييم نظام ذكي لكشف التسلل (IDS) يعتمد على تعلم الآلة والتعلم العميق، بهدف تحديد حركة المرور الخبيثة في شبكات IoT في الزمن الفعلي. تم الاعتماد على مجموعة بيانات dataset IoT-23، حيث تم استخلاص البيانات من 16 سيناريو مختلف وخضعت لعملية معالجة مسبقة شاملة تضمنت تنظيف البيانات، وهندسة السمات، وتحويل المسألة إلى تصنيف ثنائي بين حركة المرور الحميدة والخبيثة. تم بعد ذلك تدريب وتقييم أربعة نماذج تصنيف مختلفة: ثلاثة نماذج تعلم آلة (Random Forest, XGBoost, LightGBM) ونموذج تعلم عميق (Neural Network)، وذلك باستخدام مقاييس أداء دقيقة مثل F1-Score.

لإثبات الجدوى العملية، تم بناء بيئة محاكاة متكاملة باستخدام جهازين افتراضيين، حيث يقوم نظام الكشف باستخدام أداة Zeek لمراقبة الشبكة وتحليلها فورياً، وتمرير البيانات عبر قناة (Pipe) إلى محرك تنبؤ يستخدم نموذج LightGBM المدرّب مسبقاً، مع عرض النتائج على لوحة معلومات تفاعلية. أظهرت النتائج تفوقاً كبيراً للنماذج القائمة على الأشجار، التي حققت دقة تجاوزت 99%، وتم اختيار نموذج LightGBM للنشر بفضل توازنه المثالي بين الدقة العالية والكفاءة الحسابية. يخلص هذا المشروع إلى إثبات فعالية وجدوى تطبيق تعلم الآلة في بناء أنظمة كشف تسلل قوية واستباقية قادرة على تأمين بيئة إنترنت الأشياء ضد التهديدات الحديثة.

الكلمات المفتاحية: *إنترنت الأشياء، أمن الشبكات، نظام كشف التسلل، تعلم الآلة، تعلم عميق، الروبوتات الشبكية (Botnets)، مجموعة بيانات IoT-23.*

Abstract

With the rapid spread of Internet of Things (IoT) networks and their accompanying security challenges, these devices have become a primary target for the formation of botnets used to launch large-scale attacks. This project addresses this problem by designing, implementing, and evaluating an intelligent Intrusion Detection System (IDS) based on machine learning and deep learning, with the goal of identifying malicious traffic in IoT networks in real-time.

The methodology was based on the IoT-23 dataset, where data was extracted from 16 scenarios and underwent comprehensive preprocessing, including data cleaning, feature engineering, and transformation into a binary classification problem (benign vs. malicious). Four different classification models were subsequently trained and evaluated: three machine learning models (Random Forest, XGBoost, LightGBM) and one deep learning model (Neural Network), using precise performance metrics such as the F1-Score.

To demonstrate practical feasibility, an integrated simulation environment was built using two virtual machines. The detection system utilizes the Zeek tool for real-time network monitoring, streaming data via a pipe to a prediction engine that employs the pre-trained LightGBM model, with results displayed on an interactive dashboard. The results showed the significant superiority of the tree-based models, which achieved accuracy metrics exceeding 99%. The LightGBM model was selected for deployment due to its optimal balance between high accuracy and computational efficiency. This project concludes by demonstrating the effectiveness and feasibility of applying machine learning to build robust and proactive intrusion detection systems capable of securing the IoT environment against modern threats.

Keywords*: Internet of Things, Network Security, Intrusion Detection System, Machine Learning, Deep Learning, Botnets, IoT-23 Dataset.*