Syrian Arab Republic

Higher Institute for Applied Science and Tecnology

Forth Year

4th year project:

Packet inspection to detect attacks using machine and deep learning

By student: Hussein Salloum

Supervised by:

Dr- Sameeh Jamoul

Dr- Mohammed Bashar Dessouki

2024-2025

Contents

[ملخص 3](#_Toc205085722)

[هدف المشروع 4](#_Toc205085723)

[المتطلبات الوظيفية 4](#_Toc205085724)

[المتطلبات غير الوظيفية 5](#_Toc205085725)

[مخططات تصميم النظام 6](#_Toc205085726)

[فصل التنفيذ والاختبارات 12](#_Toc205085727)

[معمارية النظام المتبعة 12](#_Toc205085728)

[تفصيل أجزاء النظام 12](#_Toc205085729)

[Pseudocode 12](#_Toc205085730)

[خطة الاختبارات 13](#_Toc205085731)

[Figure 1 System block diagram 6](#_Toc205085752)

[Figure 2 Training flow chart 7](#_Toc205085753)

[Figure 3 Attacking flow chart 8](#_Toc205085754)

[Figure 4 Random forest model diagram 9](#_Toc205085755)

[Figure 5 Xgboost model diagram 9](#_Toc205085756)

[Figure 6 Lightgbm diagram 10](#_Toc205085757)

[Figure 7 Neural Network model diagram 10](#_Toc205085758)

[Figure 8 Deployment diagram 11](#_Toc205085759)

# ملخص

لقد انتشرت شبكات انترنت الاشياء بشكل ملحوظ في الأونة الأخيره مما أدّى بالطبع الى ازدياد الهجومات عليهم وخصوصا باستخدام الروبوتات الشبكية botnets, ومن هنا أتت أهمية كشف الهجومات على شبكات انترنت الاشياء في الوقت الحقيقي واخترنا لتنفيذ ذلك تدريب نماذج باستخدام التعلم الالي والتعلم العميق لكشف هجومات الروبوتات الشبكية على شبكات انترنت الاشياء باستخدام البيانات IoT-23 dataset

واحتجنا لتنفيذ ذلك الى تحميل العديد من ملفات conn.log.labeled واستخراج البيانات منها الى صيغ .csv وتبع ذلك العديد من تقنيات تنظيف البيانات وحذف السمات الغير مهمة وتحويل التصنيف الى تصنيف ثنائي (binary classification) وهو هجوم أو لا (benign or malicious) وقد دربنا على 4 أنواع من المصنفات هم: LightGBM, XGBoost, Random Forest Neural Network .

وتم عمل محاكاة لكشف الهجوم بالوقت الحقيقي باستخدام التان افتراضيتان واحده لانشاء الهجومات واخرى لكشفها

حسث أن الة الكشف استخدمت zeek لتحليل حركة المرور التي وضعت الاتصالات في ملف log ثم تم قراءة الاسطر من ملف الlog وارسالها الى pipe التي يقرأها ملف الpredection الذي يحوي نموذج مدرب ليقوم بالتنبؤات حول ماهية الاتصالات وعرض النتائج بطريقه واضحة باستخدام dashboard.

هدف المشروع

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

# المتطلبات الوظيفية

1. القدرة على تحميل بيانات التدريب والاختبار (IoT-23) بصيغة .csv وهي الصيغة التي استخدمناها من أجل :
   1. تدريب النموذج على بيانات التدريب (Training Data).
   2. اختبار النموذج على بيانات الاختبار (Testing Data).
2. معالجة البيانات بشكل مسبق (preprocessing data) ومنها:
   1. معالجة القيم المفقودة (NaN) مثل اسناد قيمة الصفر اليها او اسناد قيمة الوسيط (median) اليها.
   2. ترميز (encoding) قيم السمات من قيم فئوية (categorical values) الى قيم رقمية.
   3. حذف (drop) السمات غير المفيدة.
   4. تحويل أنماط (types) السمات الى الأنماط المناسبة لطبيعة قيمها.
   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:
   1. التدريب على نماذج مثل الغابة العشوائية والشبكات العصبونية (Random Forest, Neural Network, etc).
   2. قادراً على تقييم كل نموذج باستخدام المقاييس المختلفة مثل: الدقة (Precision) والاسترجاع (Recall) وF1-score والمساحة تحت المنحني (AUC: Area Under Curve) ومصفوفة الارتباك (Confusion Matrix).
   3. القدرة على حفظ النموذج المُدرَّب من أجل الاستخدام المستقبلي.
4. القدرة على تحميل النموذج واستخدامه للكشف في الوقت الحقيقي أي يجب أن يكون النظام قادراً على :
   1. تحميل النوذج المُدرّب من أجل استحدامه لكشف الهجومات.
   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 المُحمَّل.

* 1. التنبّؤ بماهية الاتصال ان كان سليم او هجوم اعتماداً على البيانات التي التقطها وعالجها.
  2. توليد مخرجات تنبؤ النموذج المحمل بطريقة واضحة.

# المتطلبات غير الوظيفية

1. أداء جيد للنظام حيث يجب أن:
   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.
   2. يكون قادراً على التعامل مع حركة مرور كبيرة ضمن الشبكة ومعالجتها من دون تأخير أي أنه يجب أن يتحمل الضغط الناجم عن الاتصالات.
   3. يكون قادراً على استخدام العتاد بفعالية مثل استخدام محدود للذاكرة الرئيسية لتجنب فشل النظام.
2. يكون النظام موثوقا حيث أنه يجب أن:
   1. يتوافر النظام بشكل كبير (availability) ويعمل باستمرار.
   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).
   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 المُحَمَّل من الوصول لغير المحوّلين او من التعديل والعبث بهم.
4. يكون النظام سهل الاستخدام فممكن عرض تعليمات لكيفية استخدامه خصوصاً لتشغيل خدمة الكشف.
5. تتوفر إمكانية التحديث والتغيير فمثلا يجب أن يسمح النظام بتحميل نموذج اخر اذا كان أفضل من النموذج الحالي.

# مخططات تصميم النظام

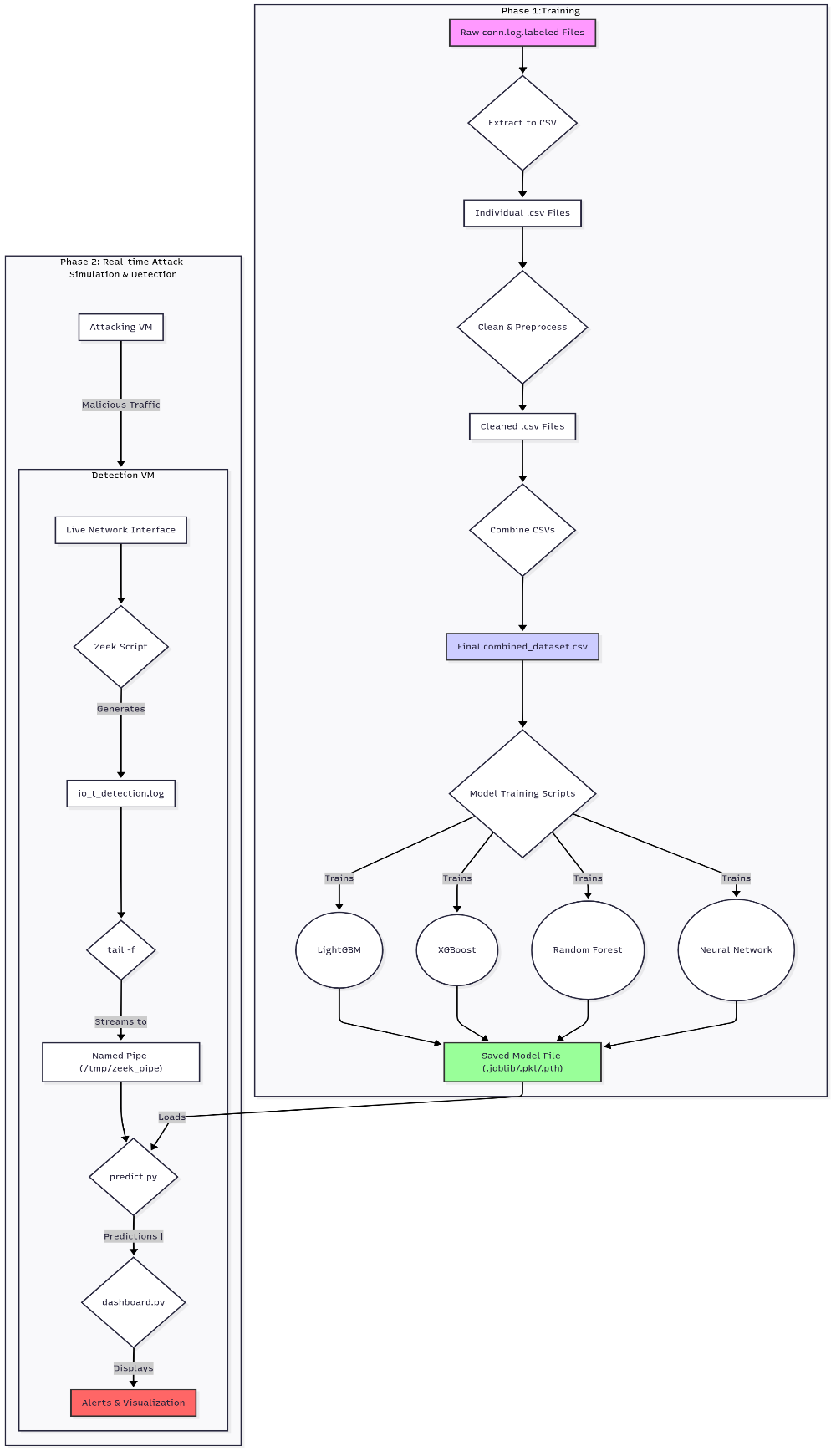


Figure 1 System block diagram

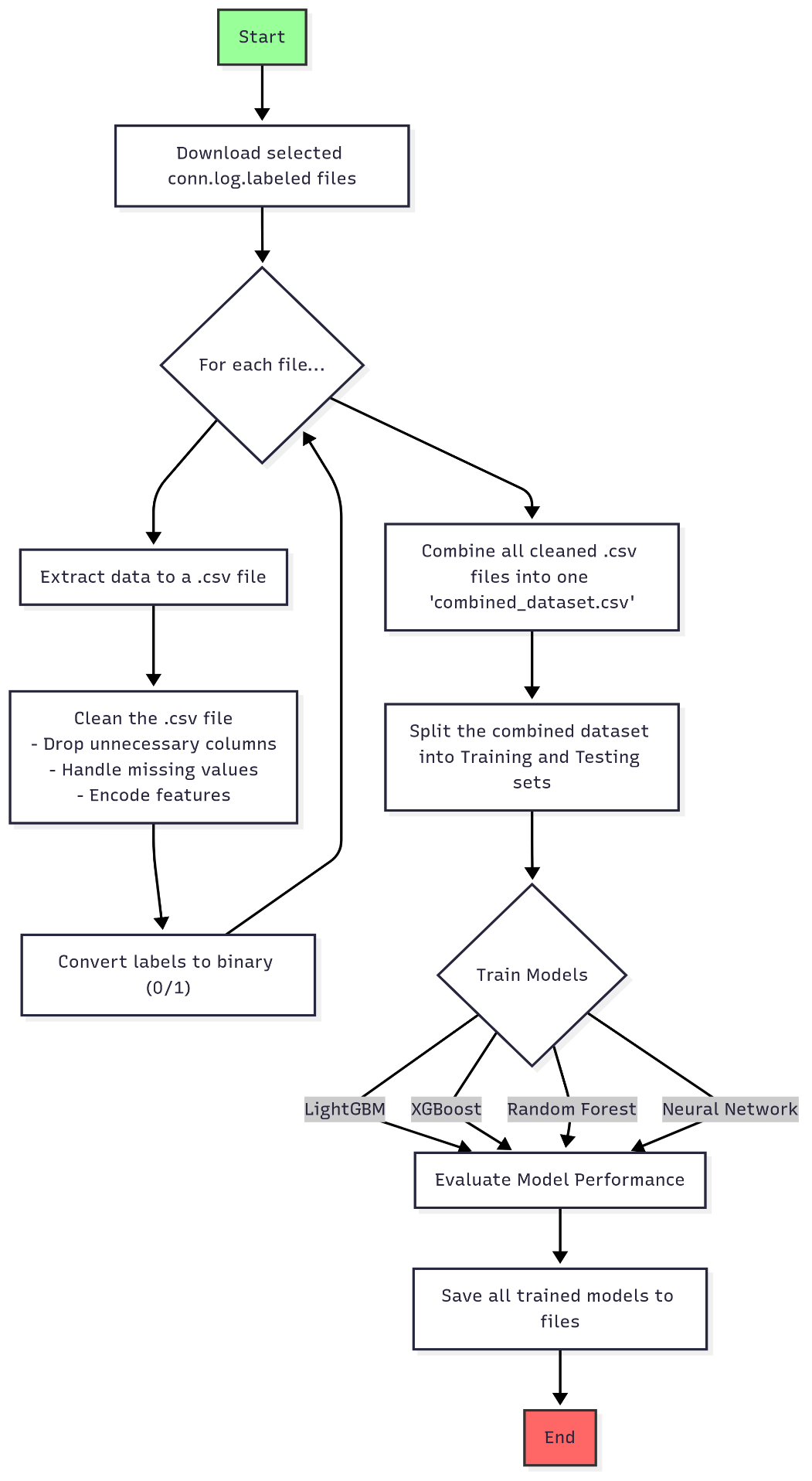


Figure 2 Training flow chart

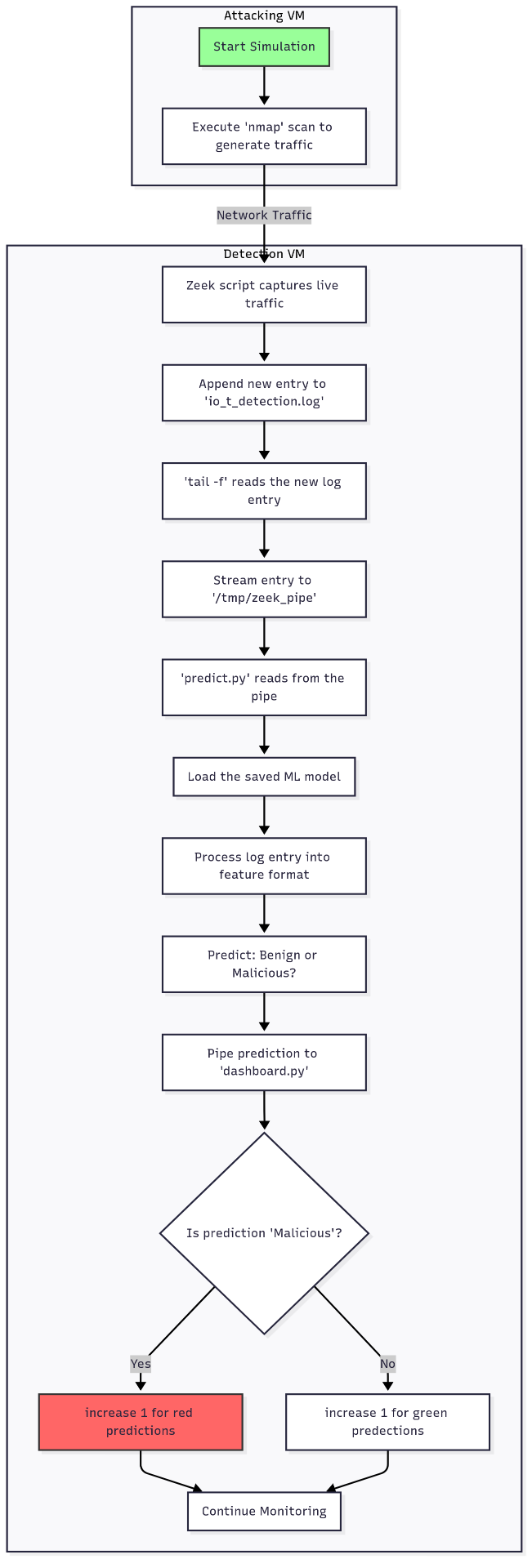


Figure 3 Attacking flow chart

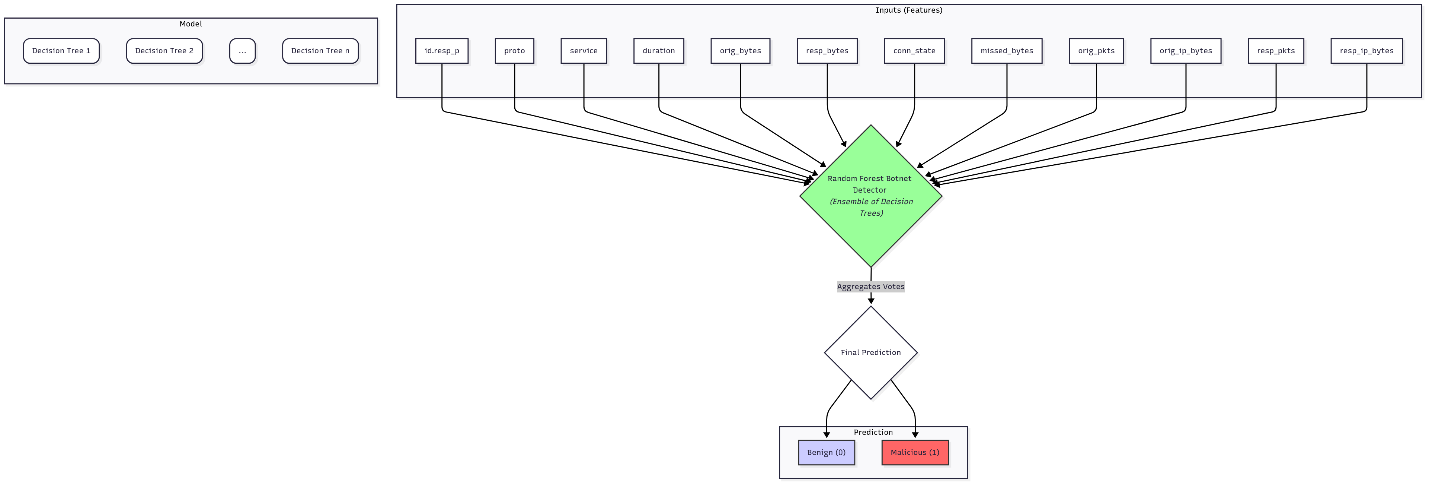


Figure 4 Random forest model diagram

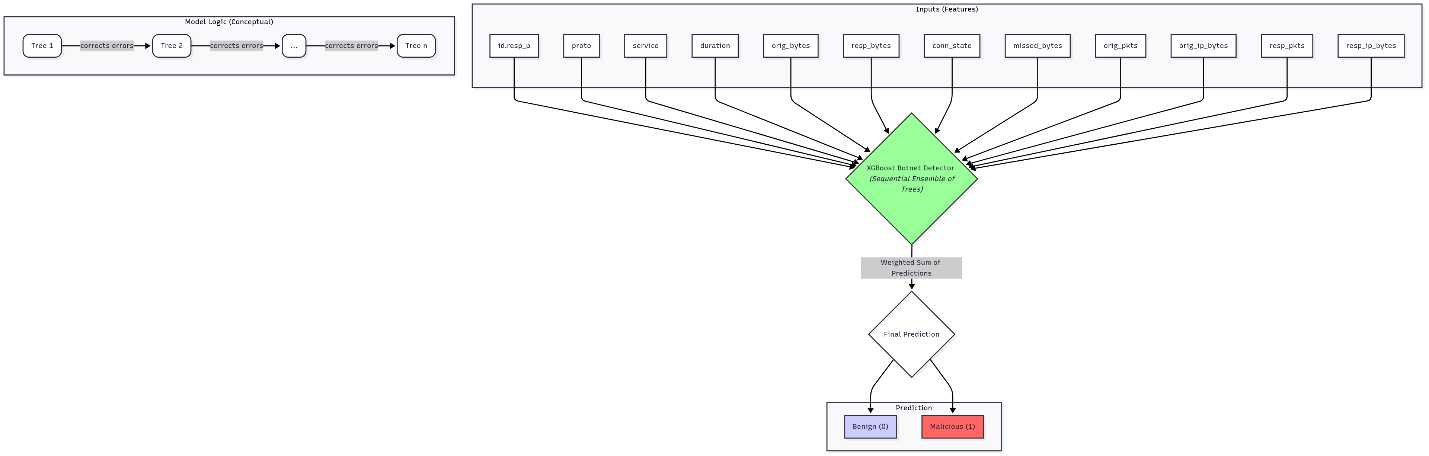


Figure 5 Xgboost model diagram

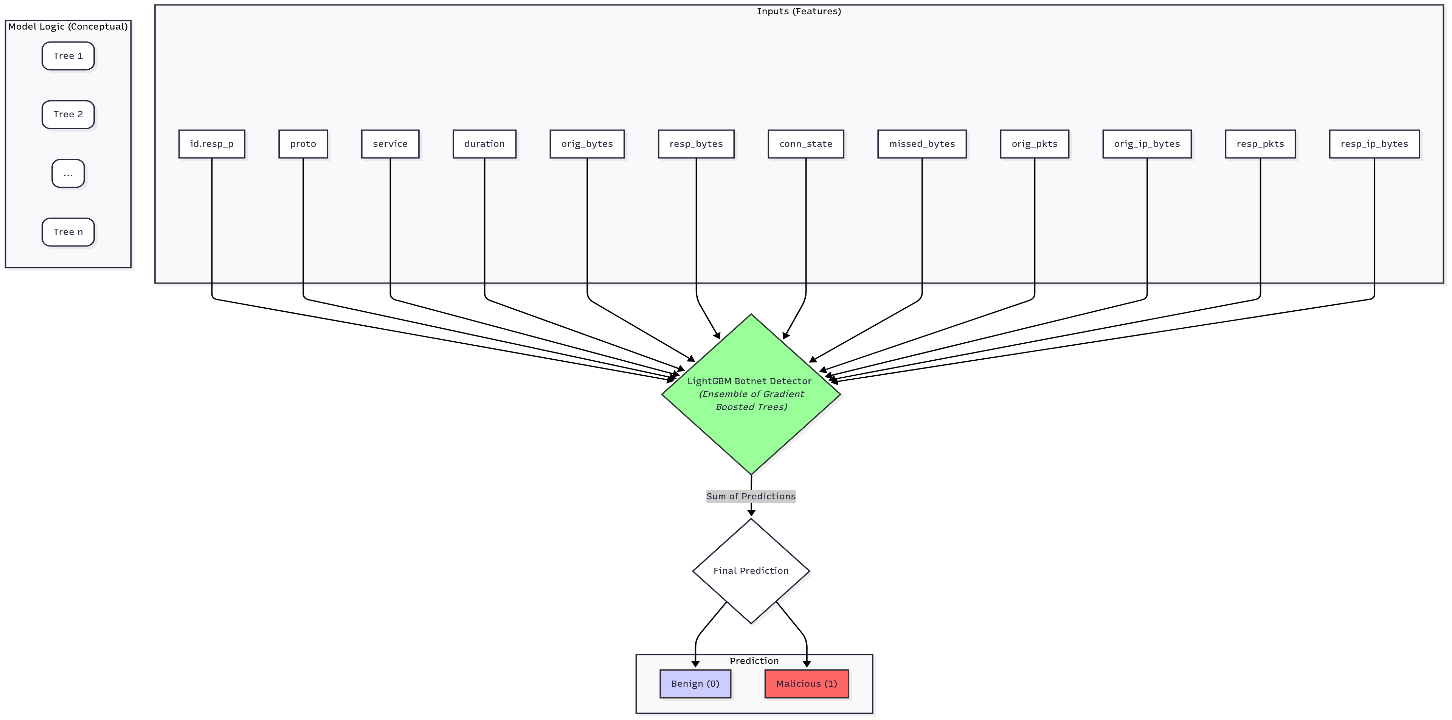


Figure 6 Lightgbm diagram

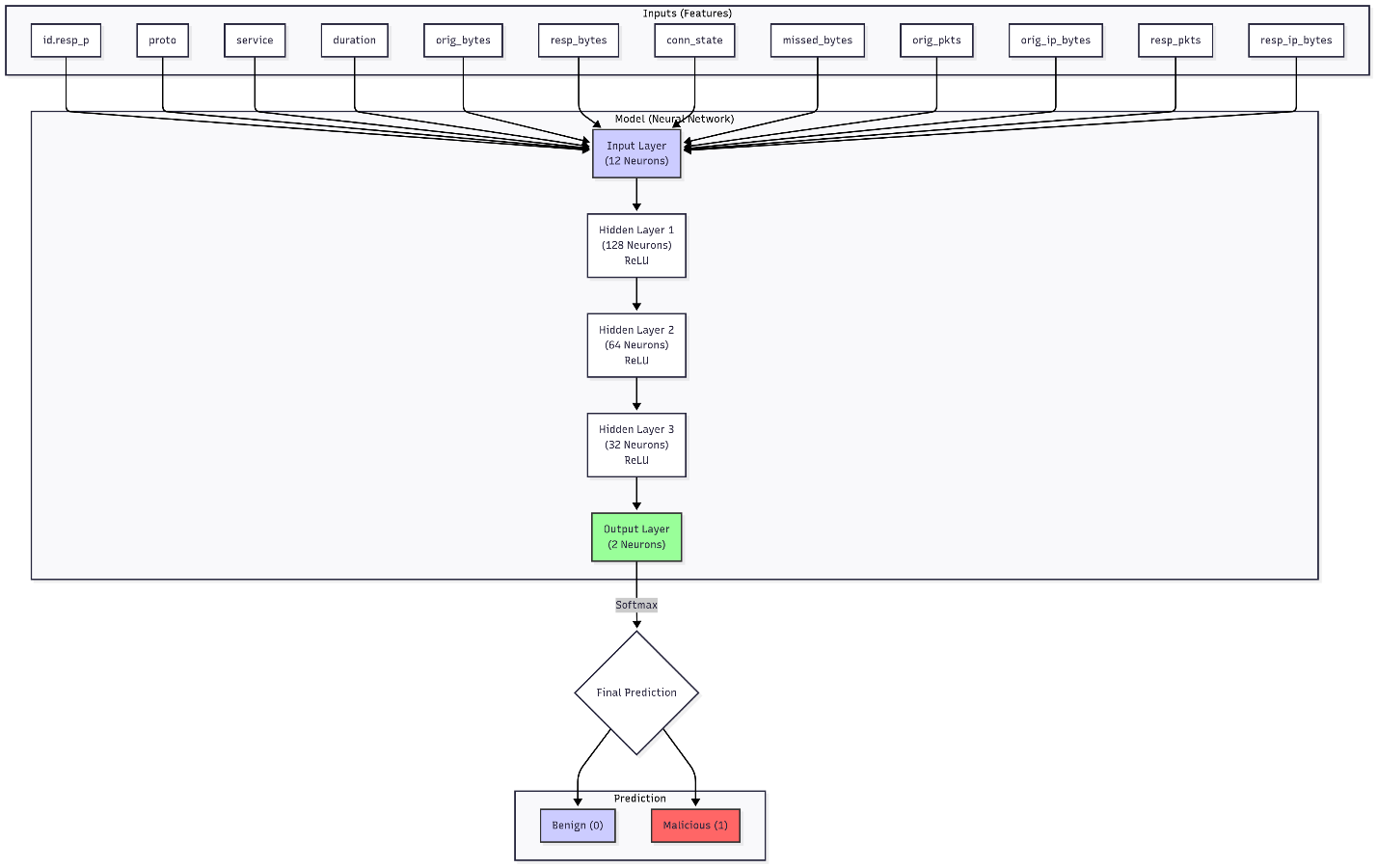


Figure 7 Neural Network model diagram

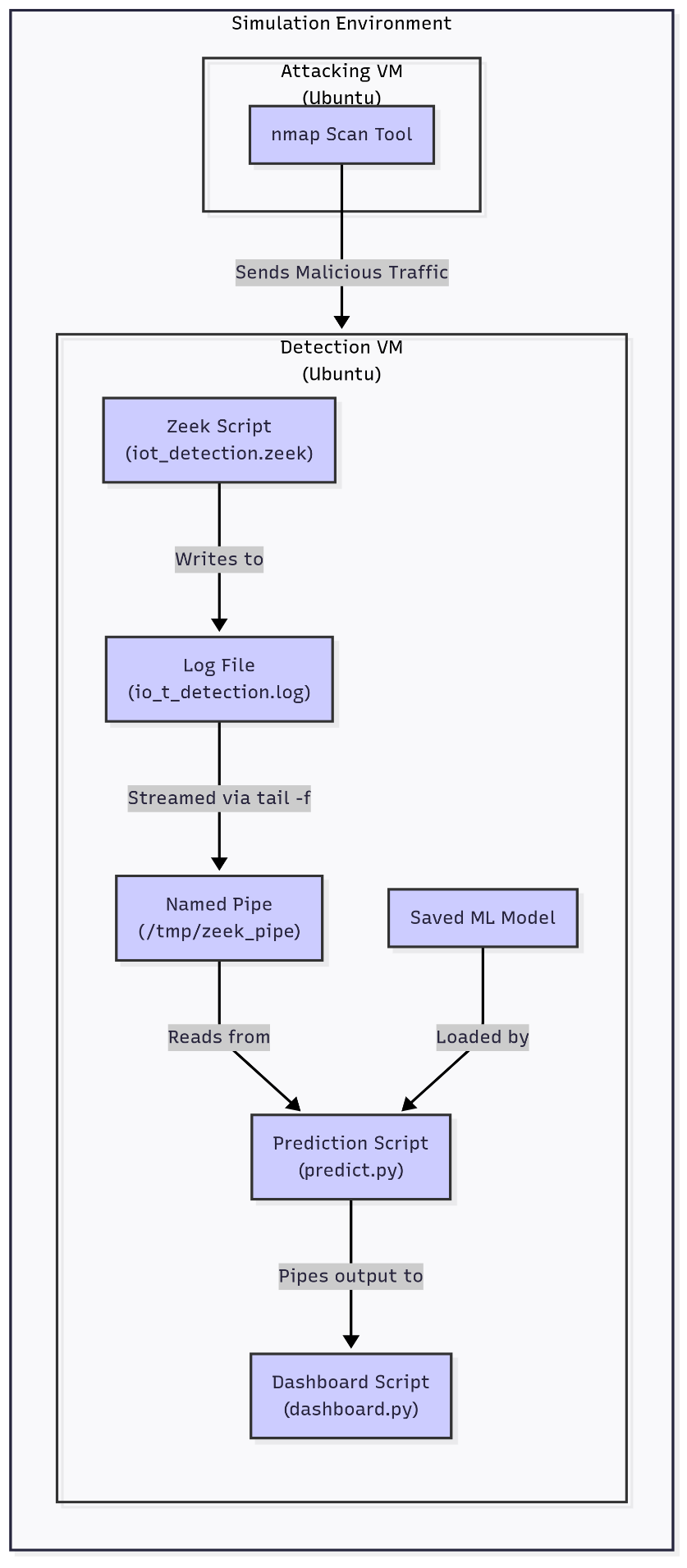


Figure 8 Deployment diagram

# فصل التنفيذ والاختبارات

معمارية النظام المتبعة: تم توضيحها في system block diagram الذي يوضح النظام كاملاً من استخراج البيانات الى محاكاة الهجومات.

تفصيل أجزاء النظام: لدينا العديد من الأجزاء مثل النماذج المدربه مثل XGBoost.pkl, Random\_forest.pkl

وملفات الاكواد مثل predict.py, iot\_detection.zeek, dashboard.py

حيث أن zeek هو لمراقبة الشبكة خيث يقوم بتجميع الحزم المتفرقه الى اتصالات ومن ثم يقوم بانشاء سطر لكل اتصال هذا السطر يمثل السمات وقيمها ويكتب هذا السطر في ملف io\_t\_detection.log

ولدينا الأمر tail -f io\_t\_detection.log > /tmp/zeek\_pipe وهو يقوم بقؤاءة الاسطر الجديده من ملف io\_t\_detection.log وارسالها الى قناة pipe

ولدينا كود التنبؤ predict.py الذي يقوم بتحميل نموذج تعلم الاله المدرب ويفوت بحلقة لقراءة الاسطر الخاصه بالقناة zeek\_pipe ومن أجل كل سطر يقوم بمعالجته المعالجة المطلوبه لبصيح الدخل للمودل مثل البيانات التي تدرب عليها، ومن ثم يتنبأ بماهية الاتصال ان كان benign أو malicious باستخدام المودل المحمّل المدرّب.

ويوجد كود dashboard.py ةهة يقةم بقراءة خرج الpredict.py ويعرض النتائج بشكل اوضح للمستخدم

تم الدمج (integration) بينهم كما التالي:

حزمة بيانات الشبكة ← أداة ←zeek ملف السجلio\_t\_detection.log ← قناة الاتصال ←(Pipe) كود التنبؤ ← طباعة النتيجة ← سكربت واجهة العرض dashboard

Pseudocode:

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

## خطة الاختبارات

Integration tests: إنّ المحاكاة التي تم توظيف المودل بها تشمل اختبارات الدمج لأن المحاكاة تدمج المودل مع اكواد مراقبة البيانات وتحليلها zeek وكود الكشف predit.py وعرض النتائج dashboard مع الهجوم

Working tests: أيضا المحاكاة التي قمنا بها تسمح بعمل اختبارات لمعرفة اذا كان كود الكشف قادر على التنبؤ بشكل صحيح ام لا اعتمادا على النماذج المدرّبة

# 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 خصيصًا 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 ٢٠١٨ و ٢٠١٩، مما يضمن أن البيانات تعكس سلوكيات الأجهزة والتهديدات المعاصرة.

تنقسم السيناريوهات الـ ٢٣ إلى فئتين رئيسيتين:

٢٠ 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 أنواعًا متنوعة وذات صلة وثيقة بمشهد التهديدات في عالم إنترنت الأشياء، مثل Mirai، وTorii، 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).

٣ 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 سابقًا 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 يدويًا 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 أيضًا على عمود تصنيف (label) حاسم يحدد صراحةً ما إذا كانت حركة المرور "حميدة "(Benign) أو نوعًا معينًا من النشاط "الخبيث "(Malicious)". هذا التصنيف المفصل لكل تدفق على حدة يُعد أمرًا ضروريًا لمهام تعلم الآلة الخاضع للإشراف (Supervised Machine Learning).

أسباب اختيار dataset iot23:

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

١. 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.

٢. 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.

٣. تصنيفات دقيقة وقابلة للتحقق: بما أنها مجموعة بيانات مصنفة (labeled)، فهي مناسبة تمامًا 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).

٤. 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.

٥. 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

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.

## 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 بتقسير بنية الملف وترويساته الheaders)) بشكل صحيح أي يقوم بمعرفة السمات (features) والقيم التابعة لكل سمه 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

ومن ثم يجب تطبيف 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 {‘duration’, ‘orig\_bytes’, ‘resp\_bytes’} من نمط (object) أي سلسلة نصية الى قيم رقمية من نمط float
6. تم معالجة القيم المفقودة داخل الأعمده {‘duration’, ‘orig\_bytes’, ‘resp\_bytes’} 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.

## دمج البيانات

بعدما حصلنا على 16 ملف لمجموعات بيانات نضيفة الان يجب دمجهم جمبعا 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.