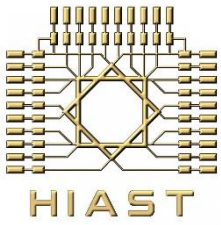
Syrian Arab Republic

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
هدف المشروع 4
المتطلبات الوظيفية 4
المتطلبات غير الوظيفية 5
مخططات تصميم النظام 6
فصل التنفيذ والاختبارات 12
معمارية النظام المتبعة 12
تفصيل أجزاء النظام 12
Pseudocode 12
خطة الاختبارات 13

Figure 1 System block diagram 6
Figure 2 Training flow chart 7
Figure 3 Attacking flow chart 8
Figure 4 Random forest model diagram 9
Figure 5 Xgboost model diagram 9
Figure 6 Lightgbm diagram 10
Figure 7 Neural Network model diagram 10
Figure 8 Deployment diagram 11

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 :
   1-1. تدريب النموذج على بيانات التدريب (Training Data).
   2-1. اختبار النموذج على بيانات الاختبار (Testing Data).
2- معالجة البيانات بشكل مسبق (preprocessing data) ومنها:
   1-2. 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.
   3-2. حذف (drop) السمات غير المفيدة.
   4-2. تحويل أنماط (types) السمات الى الأنماط المناسبة لطبيعة قيمها.
   5-2. 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:
   1-3. التدريب على نماذج مثل الغابة العشوائية والشبكات العصبونية (Random Forest, Neural Network, etc).
   2-3. 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 المُدرَّب من أجل الاستخدام المستقبلي.
4- القدرة على تحميل النموذج واستخدامه للكشف في الوقت الحقيقي أي يجب أن يكون النظام قادراً على :
   1-4. تحميل النوذج المُدرّب من أجل استحدامه لكشف الهجومات.
   2-4. استقبال البيانات ومعالجتها بشكل مباشر في الوقت الحقيقي أي يجب أن يكون النظام قادراً على:
      1-2-4. مراقبة الشبكة وأخذ بيانات الاتصالات الواردة الى جهاز الكشف.
      2-2-4. معالجة البيانات الملتقطة بحيث يتم حذف السمات واجراء معالجة للبيانات لتكون صيغتها مماثلة لصيغة البيانات التي تدرّب عليها النموذج المُحمَّل.
   3-4. 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.
   2-1. يكون قادراً على التعامل مع حركة مرور كبيرة ضمن الشبكة ومعالجتها من دون تأخير أي أنه يجب أن يتحمل الضغط الناجم عن الاتصالات.
   3-1. يكون قادراً على استخدام العتاد بفعالية مثل استخدام محدود للذاكرة الرئيسية لتجنب فشل النظام.
2- يكون النظام موثوقا حيث أنه يجب أن:
   1-2. 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).
   3-2. 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

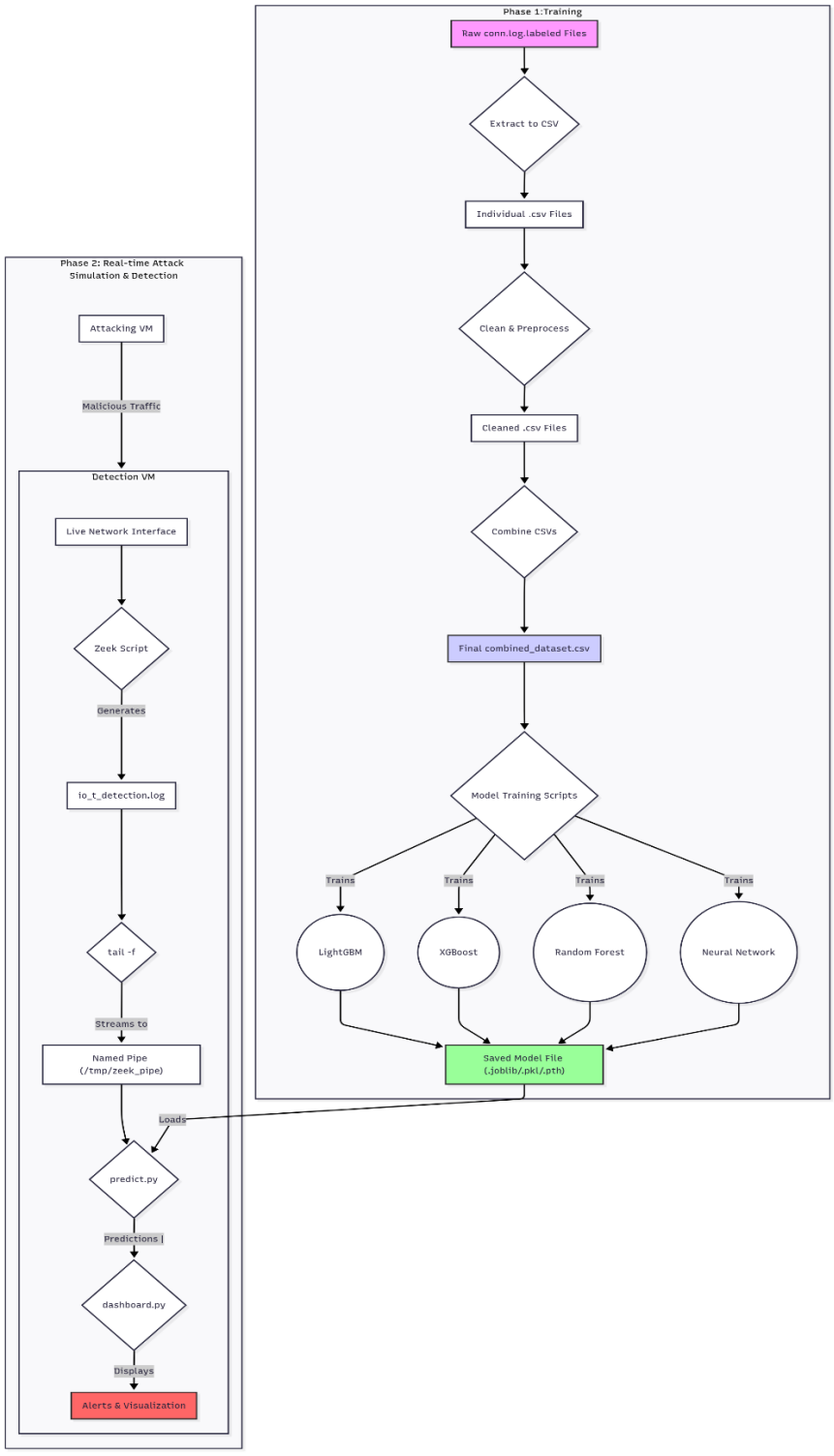


Figure 1 System block diagram

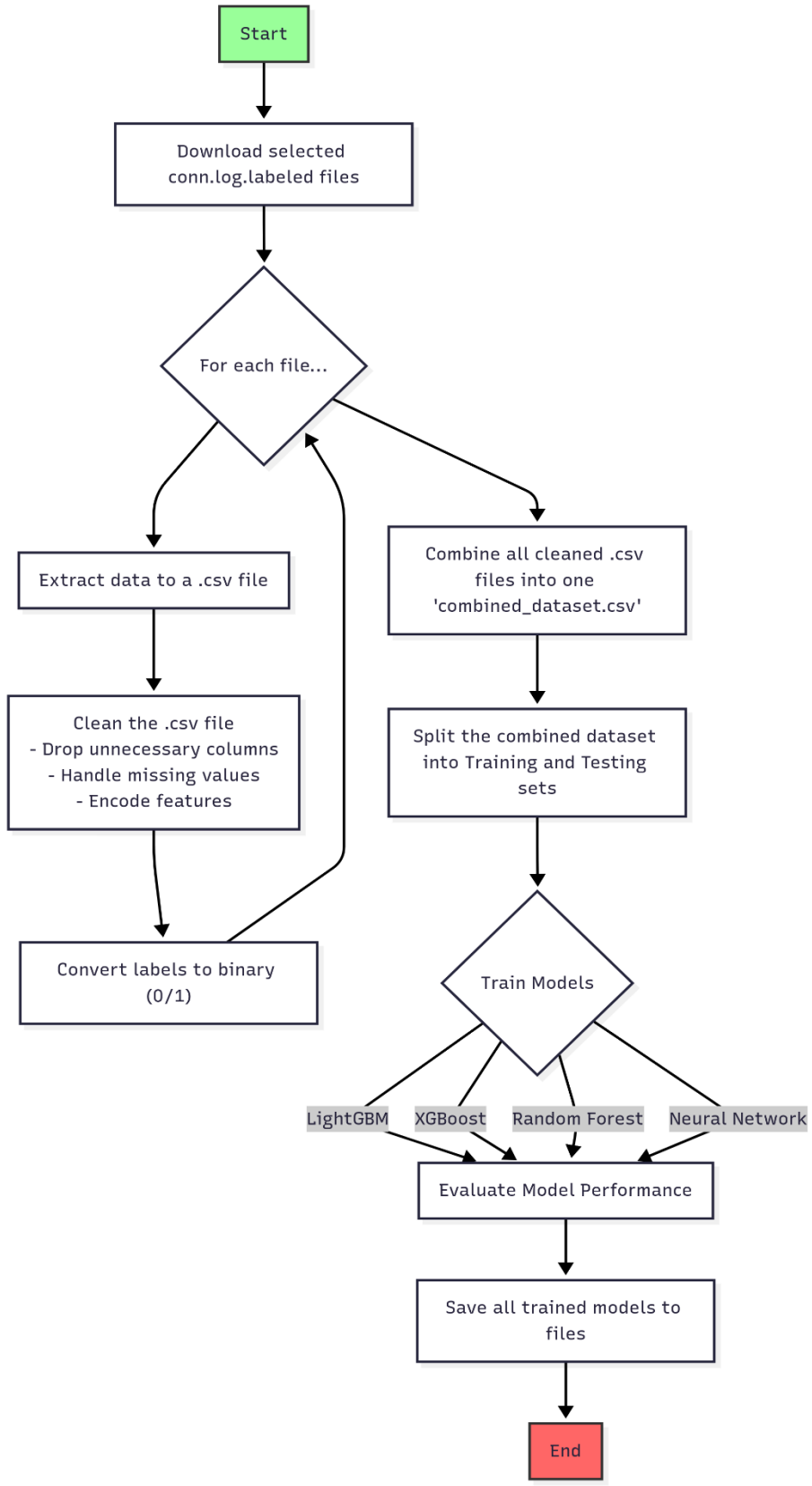


Figure 2 Training flow chart

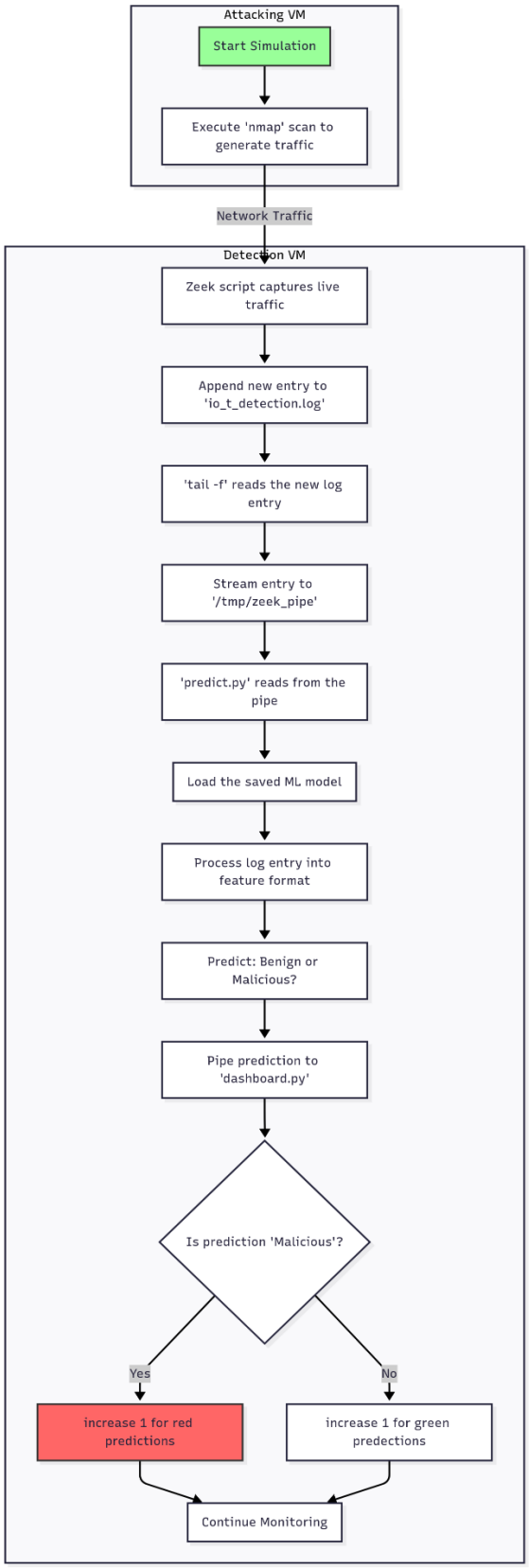


Figure 3 Attacking flow chart

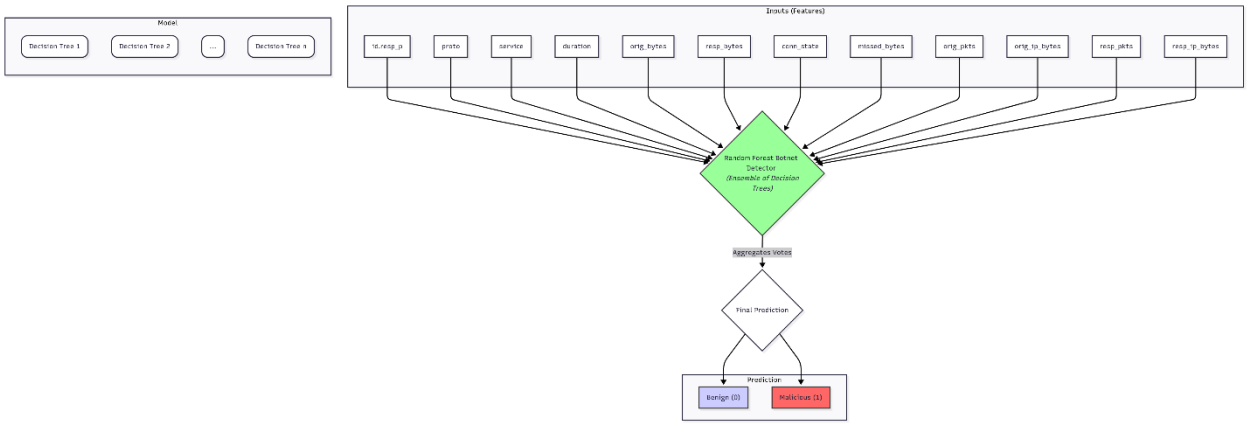


Figure 4 Random forest model diagram

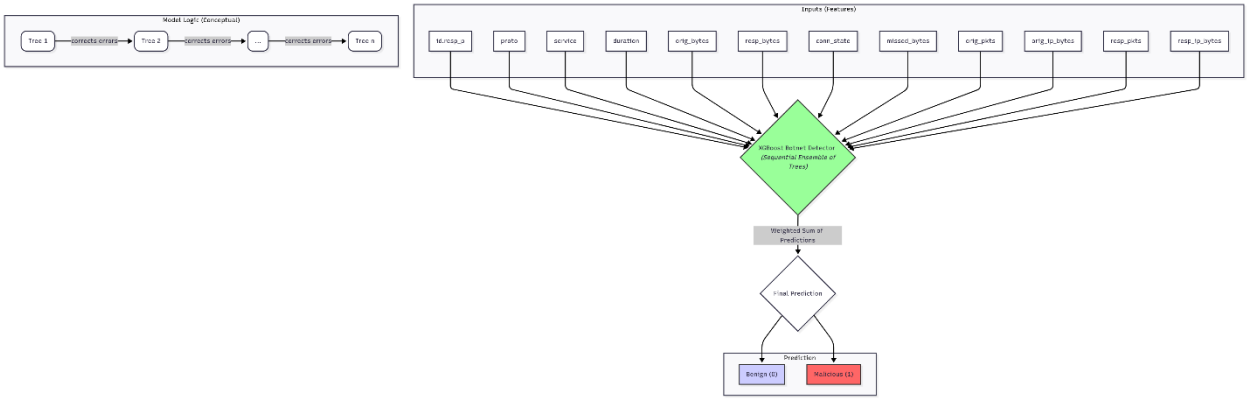


Figure 5 Xgboost model diagram

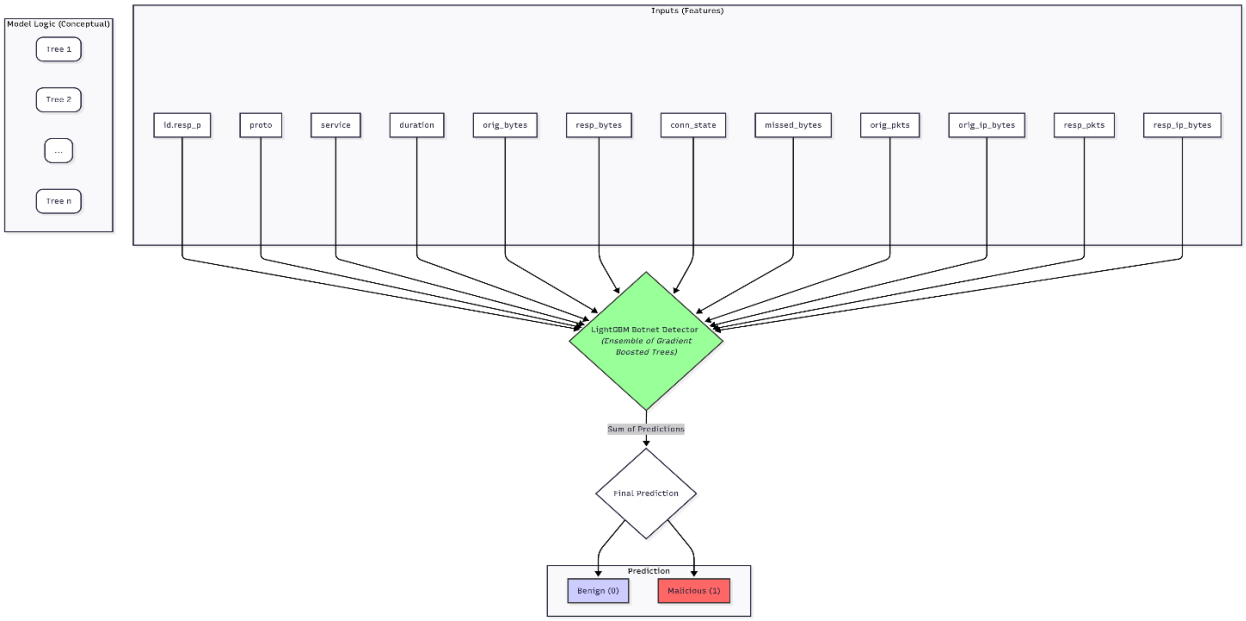


Figure 6 Lightgbm diagram

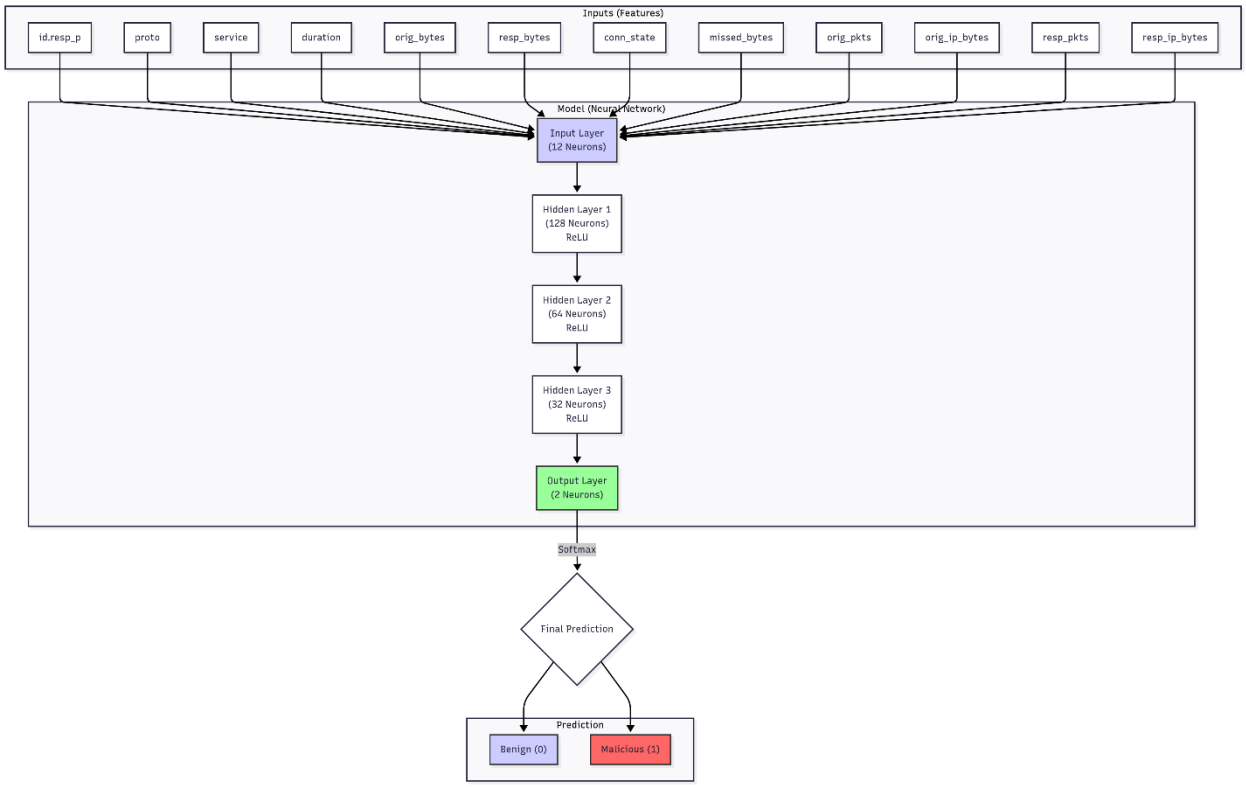


Figure 7 Neural Network model diagram

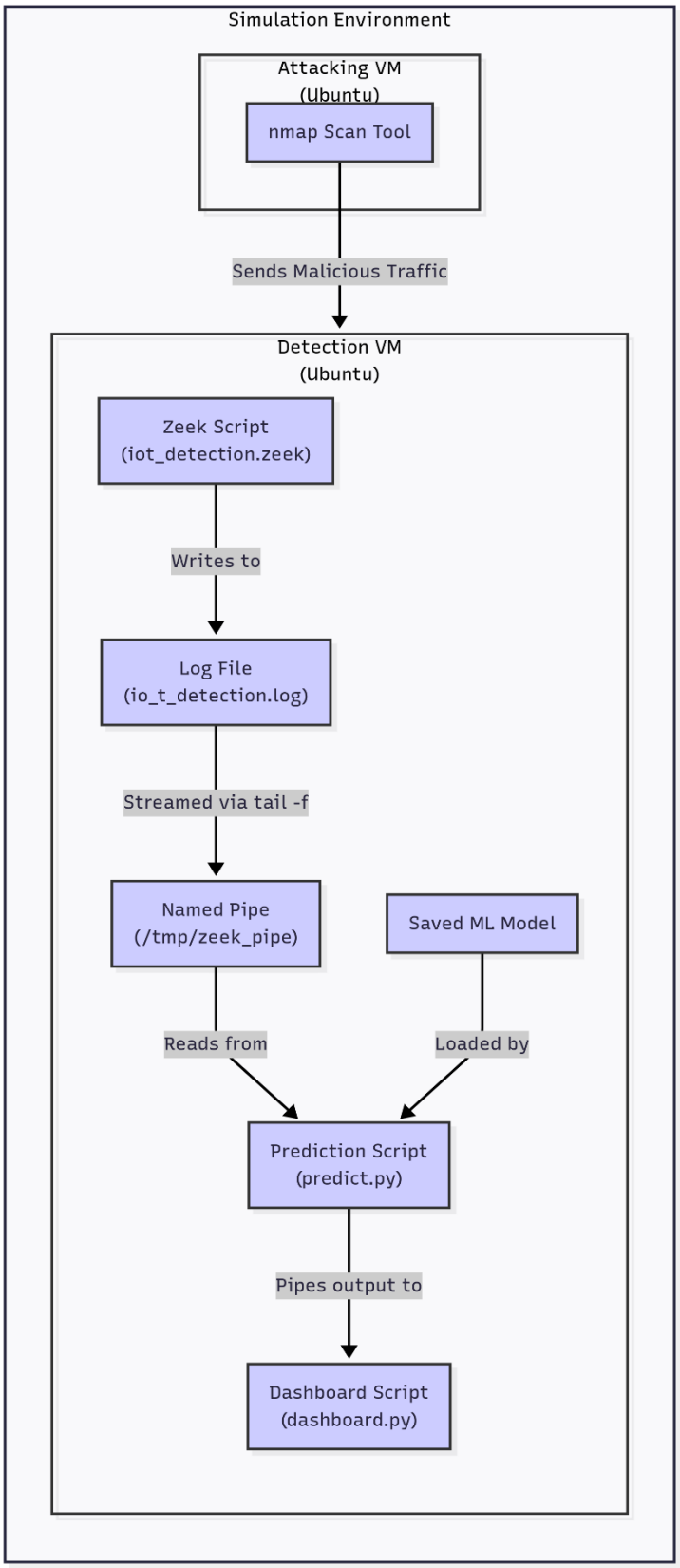


Figure 8 Deployment diagram

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 benignأو 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