Secure IoT Network Attestation Framework Using Ensemble Learning

***Abstract* — Attestation involves providing verifiable evidence to support claims about the characteristics of a target device. Devices that manage sensitive data—such as wearables, Industrial IoT devices, and servers—are vulnerable to various threats, including edge-based attacks, denial-of-service (DoS) attacks, counterfeiting, and firmware tampering. Attestation plays a crucial role in ensuring the integrity of firmware and configurations, authenticating genuine hardware, and verifying that composite device assemblies remain unaltered. However, a review of existing solutions highlights significant deficiencies, particularly in comprehensive lifecycle management and support for large-scale networks. This paper introduces a novel attestation method based on optimized ensemble learning, which achieves a predictive accuracy of 93%, addressing these gaps and enhancing security across the entire device lifecycle. .**

Keywords—Attestation, IoT Security, Device Verification, Machine Learning based Attestation

I. Introduction

Attestation is a mechanism used to verify the authenticity and integrity of the hardware and firmware of the device by a relying party. It ensures that the device is genuine, untampered, and performing its operations as expected. A typical attestation architecture as shown in Figure 1 includes a verifier, attester, relying party, endorser, reference value provider, verifier owner and a relying party owner.

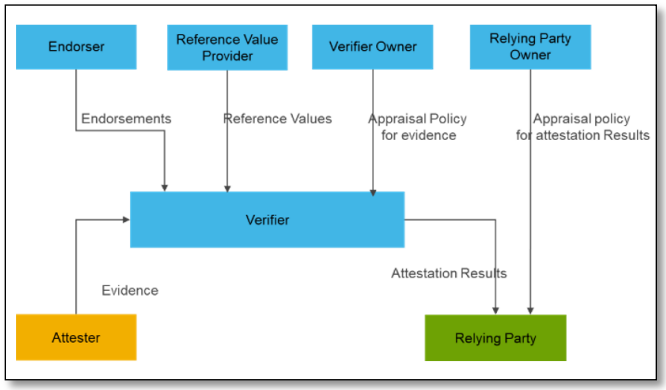

Figure 1: Attestation Architecture

The attester is the device whose trustworthiness has to be established. It produces evidence that is conveyed to a verifier. The verifier uses this evidence, along with any reference values conveyed from reference value providers, and any endorsements received from endorsers by applying an appraisal policy for evidence to the verifier owner for assessing the trustworthiness of the attester. The appraisal policy for evidence is obtained from the verifier owner either via some protocol mechanism, or via configurations made into the verifier by the verifier owner, or via programming into the verifier, or via some other mechanism. The verifier generates attestation results to be used by relying parties. The relying party uses attestation results by applying its own appraisal policy to make application-specific decisions, such as authorization decisions which is known as the appraisal of attestation results. While attestation based on the device manufacturer's information such as hardware identifier certificate, and firmware measurements are described in multiple prior studies, continuous device attestation based on the behaviour of the device within its operational environment creates security challenges. Some of the examples of security challenges in the runtime environment are injection based attacks, ransomware, and cross site scripting.

Runtime attestation based on in-field attack scenarios demands mechanisms that incorporate attack patterns, which are rarely implemented in traditional attestation architectures. This paper seeks to redesign the verifier in a typical attestation architecture to address potential runtime attacks, thereby achieving a secure IoT network. We propose a novel attestation framework for detecting malicious IoT devices and various malware attacks using a Bayesian-optimized maximum voting classification ensemble model and a stack ensemble model. The proposed solution leverages the strengths of diverse base models with an XGBoost meta-learner, eliminating the need for a legitimate copy of the original firmware for multi-class classification. The effectiveness of these ensemble models in managing complex patterns and enhancing predictive performance is demonstrated by the experimental results on a realistic cybersecurity dataset encompassing various IoT and Industrial IoT Perception Layer attacks.

In this paper, Section II describes the prior studies and related works in this field, Section III discusses in-depth about the characteristics of the dataset used, Section IV provides detailed explanation of the algorithms used, preprocessing of data, model training, optimization and performance comparison, Section V highlights the experimental results obtained, inferences and analysis of results and lastly concluding the work with Section VI.

II. Literature Review

A. Traditional Attestation Methods

The principles of remote attestation were discussed in [1]. This paper introduced the fundamental principles of attestation, including the terminologies related to attestation and the fundamental principles.

SPDM [2] is a protocol by DMTF for providing attestation evidence. A study on securing hard drives with SPDM is discussed in [2]. Though this study is focused on storage devices, similar concepts are applicable to other devices which form part of a larger composite assembly. A formal analysis of the security of SPDM protocol message exchanges is presented in [4]. This analysis provides visibility into the security of the message exchanges, and there is no other significant analysis of attestation at a system level.

A detailed study on secure compute enclave-based attestation is covered in [5]. Various aspects of attestation, such as identities, models of attestation, and composite device attestation, are discussed in detail in this paper. Another study on remote attestation schemes with taxonomy and review is presented in [6]. The literature primarily focuses on component-level attestation. Attestation at a system level is an open topic of research which needs further study.

Architecture for remote attestation is worked on by the IETF Remote Attestation Task Group [7]. Defining the actors involved in the attestation architecture and the data flow paths is a major contribution to this architecture. The definition of data flows is limited to conceptual messages without defining the data structures and schema for the message exchanges.

A device in its lifecycle can be compromised at any point in the supply chain, operation and decommissioning. Considering this possibility, attestation frameworks need to consider the device lifecycle and onboarding mechanism. Various onboarding solutions are proposed by different standards considering the needs of the industry segment. OPCUA Device boarding [7] is targeted for industrial device onboarding to an operational network. Bootstrapping Remote Secure Key Infrastructure [9] provides solutions for device onboarding for scenarios considering different connectivity options. These include internet-connected devices, disconnected scenarios with suitable security assumptions. Secure Zero Touch Provisioning [10] considers provisioning the operating system and configurations apart from hardware provisioning. A provisioning solution specifically meant for IoT devices is FIDO Device Onboarding [11]. While these onboarding solutions consider hardware identities, they do not consider attestation to ensure that the firmware and configuration are trustworthy.

B. Machine Learning Based Attestation

Traditional firmware attestation methods often depend on having an authorized copy of the firmware, which is frequently unavailable due to its status as the manufacturer's intellectual property (IP). Existing machine learning (ML) classifiers described in [12] are based on analysis of memory dumps from IoT devices, increasing the computational complexity of remote verifiers assessing the integrity of the devices' internal state. This technique collects memory traces from the device main memory and converts them to grayscale images. Convolutional Neural Network based classifiers are applied on these images to detect any tampering with the device firmware.

Different ensemble learning methods like bagging, voting, stacking, aggregating based on parametric sensitivity analysis were first studied in [22] providing support to the results of better performance of ensemble models over the individual ML models in anomaly detection. This paper comprehensively discusses the development and performance of a verifier-like runtime attestation framework for multiclass IoT malware detection using optimized ensemble learning.

III. Characteristics of Dataset

We have used "Edge-IIoTset Cyber Security Dataset of IoT & IIoT" [13] for training the base models in our ensemble model. It is an inclusive and realistic cybersecurity dataset designed to support machine learning-based intrusion detection systems (IDS) in Internet of Things (IoT) and Industrial IoT (IIoT) environments. The testbed is organized into a multi-layered architecture and this paper focusses on the IoT and IIoT Perception Layer. Distribution of attacks in the IoT and IIoT Perception Layer is shown in Figure 2.

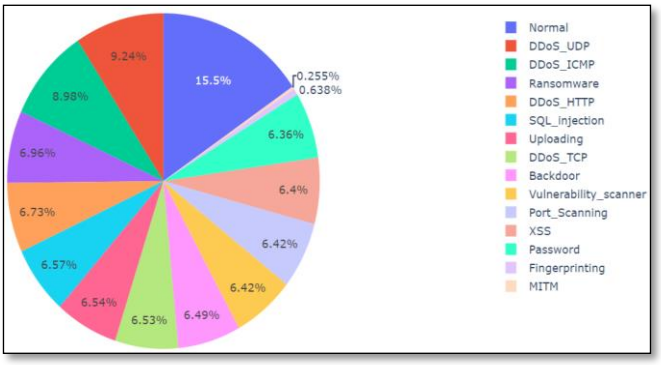

Figure 2 : Distribution of attacks in the IoT and IIoT Perception Layer.

The dataset is collected from over 10 different types of IoT devices such as low-cost digital temperature sensors, humidity sensors, ultrasonic sensors, water level detection sensors, pH meters, soil moisture sensors, heart rate sensors, to name a few. The dataset proposes fourteen distinct attacks targeting IoT and IIoT connectivity protocols which are further classified into five major threat categories: DoS/DDoS attacks, information gathering, man-in-the-middle attacks, injection attacks, and malware attacks.

Figure 3 visualizes 15 out of 62 attributes of the dataset using a heatmap highlighting redundancy and important feature interactions extracted from dataset. Few of the key observations drawn from the heatmap include strong interdependency of features like ICMP (Internet Control Message Protocol) checksum and sequence number, which are two important fields of ICMP message header and interdependency of TCP (Transmission Control Protocol) acknowledgement and connection synchronization, two crucial concepts in error-checked data delivery using TCP.

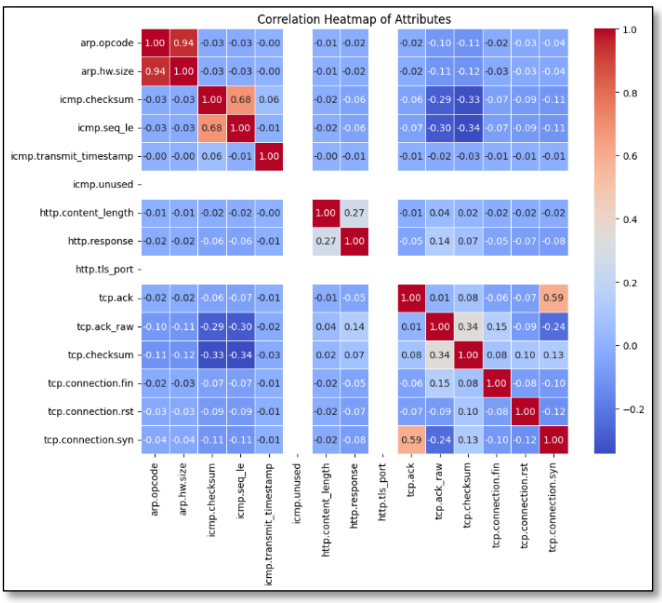

Figure 3 : Correlation Heatmap of 15 Features. The orange-colored cells highlight interdependency of features with a correlation coefficient between 0 and 1 while dark blue-colored cells highlight weak relationship of corresponding features with a correlation coefficient <1.

IV. Algorithms and Methodology

A. Data Analysis and Feature Engineering

Exploratory data analysis (EDA) is foundation to building robust intrusion detection systems in heterogenous IoT environments. Upon carrying exploratory data analysis (EDA) on the testbed with 62 features and 158k data records [13], the following observations and follow-ups were made:

- The null and duplicate values found in the dataset were cleaned and features were normalized and standardised using Standard and Min-Max Scaling.

X' : scaled value, X_{min} : minimum value in the dataset, X_{max} : maximum value in the dataset, σ : standard deviation of the dataset, μ mean of the dataset.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (1)$$

$$X' = \frac{X - \mu}{\sigma} \quad (2)$$

- Label encoding was applied to the categorical values like names of attack types and attack labels to convert them into one hot encoded feature. 14 different malware attacks were labelled from 1 to 14 and normal class was labelled as 0.
- The ratio of malware class to normal class was found to be 5.46, highlighting imbalanced classes and the need for synthetic sampling to be applied.

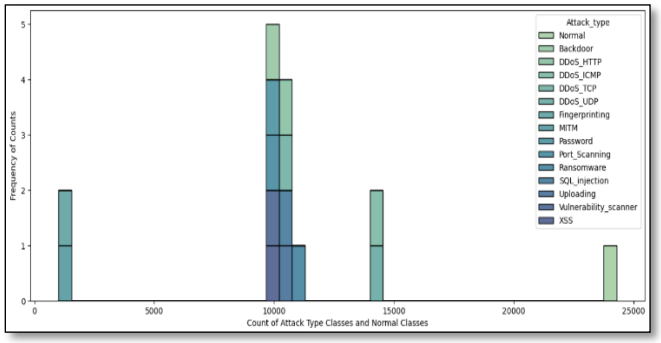

Figure 4 : Visual representation of the frequency distribution of the counts for different malware (attack types) classes and normal classes.

The observed imbalance in classes can lead to the ensemble model becoming biased towards the majority class, i.e. the class with significantly higher number of samples. Hence, the dataset was partitioned into training, validation and testing subsets followed by Synthetic Minority Oversampling (SMOTE) described in [14] to address imbalanced classes. The pseudocode for calculating the amount of smote (N) to be applied while generating synthetic examples for an imbalanced dataset based on k-nearest neighbours within the same class, described first by Nitesh Chawla, et al. [14] is provided below:

k: number of nearest neighbours, attrs: number of attributes, org[ ][ ]: an array for original minority class samples, count: initialized to 0 (keeps a count of the number of synthetic samples generated), synth[ ][ ]: an array for synthetic samples

Algorithm SMOTE(T, N, k)
Input: T: Number of minority class samples
N: Amount of SMOTE (%)
k: Number of nearest neighbours
Output: (N/100) * T synthetic minority class samples

1. **if** N < 100: Randomize the T minority class samples
2. T = (N/100) * T
3. N = 100
4. **else:**
5. N = (int)(N/100)
6. **for** each sample i from 1 to T:
Compute k nearest neighbours for i and save the indices in array
7. Call Populate(N, i, array)
8. **end for loop**

9. **function** Populate(N, i, array):
10. **while** N > 0:
Choose a random number between 1 and k, call it n
11. **for** each attr from 1 to attrs:
Compute diff = org[array[n]][attr] − org[i][attr]
Compute gap as a random number between 0 and 1
12. synth[count][attr] = org[i][attr] + gap * diff
13. **end for loop**
14. count = count + 1
15. N = N - 1
16. **end while loop**
17. **end function**
18. return synth[ ][ ]

Since the multi-dimensional dataset is broadly featured, dimensionality reduction of the original dataset without compromising bias or variance is essential for faster performance. Hence, dimensions of the dataset were reduced using Fisher's Linear Discriminant Analysis [15] that projects original multi-dimensional feature space to a new reduced space while retaining essential information and enhancing model performance. Figure 5 shows the feature space reduced and transformed to one-dimensional space, using Fisher's LDA.

Fisher's LDA discussed in [15] entails: (2) S_B as the scaled between-class scatter matrix, while (3) S_W is the scaled within-class scatter matrix based on (4) which is the summation of the product of the centralized input matrix and its transpose. D' (5) is the new reduced dimensional space. W is the optimal projection matrix formed with D' eigenvectors corresponding to D' largest eigenvalues in (6) i.e. maximising class separability. Through this dimensionality reduction we aim to maximise the separation of within-class and between-class variance.

K : number of classes, N_i : number of inputs in class C_i , m_i : local mean, m : global mean, n : original number of features.

$$S_B = \sum_{i=1}^{K} N_i (m_i - m)(m_i - m)^T \quad (2)$$

$$S_W = \sum_{i=1}^{K} S_i \quad (3)$$

$$S_k = \sum_{n \in C_i} (x_n - m_i)(x_n - m_i)^T \quad (4)$$

$$D' = min(K-1, n) \quad (5)$$

$$W = max_D(eig(S_w^{-1}S_B)) \quad (6)$$

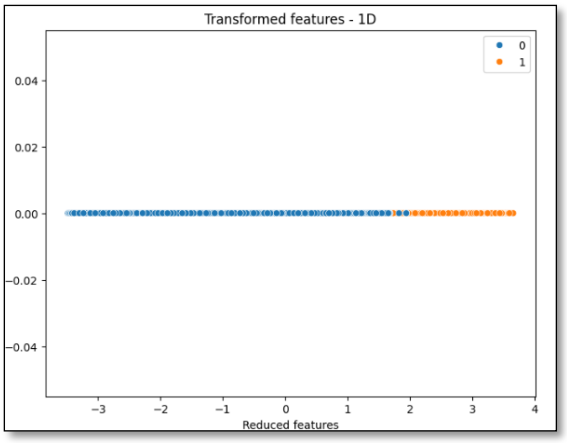

Figure 5 : Linear transformation to one-dimensional feature space with blue highlighting normal classes and orange for malware classes.

The pre-processed data was used to train the base models of maximum voting ensemble classifier and stack-based ensemble model with 20% of the data used as testing set.

B. Model Training

Ensemble Models such as maximum voting classifiers are typically effective in heterogeneous IoT environment, where they enhance generalization and mitigate overfitting risks [16]. With *soft voting*, probability estimates for each class are aggregated from each base classifier with the ensemble predicting the class with the highest average probability. A diagrammatic representation of maximum voting ensemble model with soft voting for binary classification is shown in Figure 6.

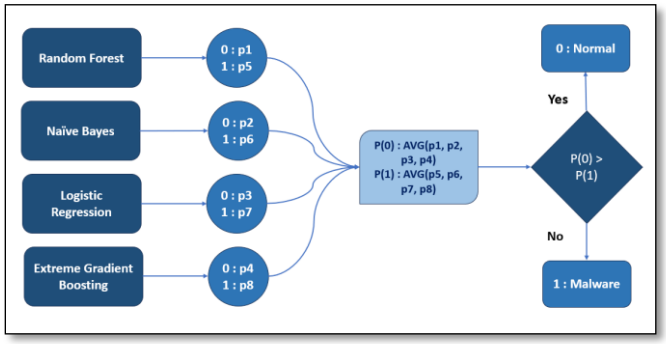

Figure 6 : Diagrammatic representation of maximum voting ensemble model with Soft Voting for binary classification.

Stacking ensemble technique leverages the strengths of multiple machine learning models [17]. This approach involves training each of the base learners on the dataset and then combining their predictions using a *meta-learner* to make the final prediction efficiently. The meta-learner/second level learner [18] is trained on the predictions obtained by integrating the capabilities of diverse algorithms or first-level learners known as individual learners and 20% of the original training set as validation data. By combining the diverse base models, ensemble models harness their individual strengths and mitigate their weaknesses, leading to a robust and highly accurate predictive model. Diagrammatic representation of stack ensemble model with XGBoost meta-learner for multi-class classification is shown in Figure 7.

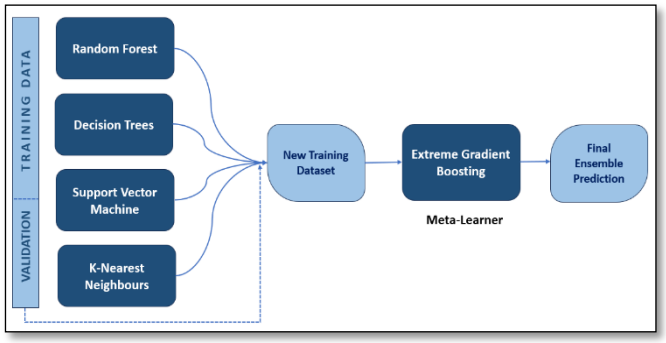

Figure 7 : Diagrammatic representation of stack ensemble model with XGBoost meta-learner for multi-class classification.

The proposed solution is based on a maximum voting classification ensemble model with random forest, logistic regression, extreme gradient boosting and gaussian naïve bayes as base models for binary classification. The multiclass classification is based on Stack ensemble model with decision trees, k-nearest neighbors (KNN), support vector machines (SVM) and random forest as base models and XGBoost as the meta-learner.

The base learners or models using in the solution include - extreme gradient boosting, whose property of iteratively boosting weak learners, enhances the model's ability to handle complex patterns; decision trees that contribute through their simplicity and interpretability, as base models; logistic regression that offer probabilistic predictions and insights into feature importance; naïve bayes that provide probabilistic classification using Bayes' theorem, assuming strong feature independence; random forest that aggregates the predictions from an ensemble of decision trees to improve generalization; support vector machine (SVM) which excels in finding hyperplanes that best separate classes in high-dimensional spaces; k-nearest neighbours (KNN) that provide a non-parametric approach that captures local data structures.

A convolutional neural network (CNN) leverages convolutional layers to capture hierarchical patterns and spatial features. A CNN model was also trained using the same feature-engineered training data with early stopping callback and Adam optimizer. When tested using the same test data, CNN model resulted in an accuracy of 81% with 80% precision for multi-class classification.

The hyperparameters of the base learners in the ensemble model were further fine-tuned to optimize the model training process and enhance the overall performance.

C. Hyperparameter Tuning

Selecting the optimal set of hyperparameters avoids the model from overfitting or underfitting the training data and performing better on new input values. In order to enhance the model performance, model parameters like learning rate, maximum depth of random forest, number of estimators of XGBoost, penalty, class wieght for logistic regression, variable smoothing in naïve bayes, to name a few, are properly tuned prior to model training. There are several hyperparameter tuning methods, like random search, grid search, bayesian optimization and hyperband. Random

search is a randomized search for the best hyperparamter combination, hence does not guarantee optimal solutions [19], while grid search is an exhaustive search that trains model for every hyperparameters combination found and thus, becomes computationally intensive for large number of hyperparameters [20]. Hyperband first described in [23] is a bandit-based tuning which presumes that the performance of hyperparamter configurations can be adequately estimated using a minimal amount of resources, which is not always feasible.

Bayesian optimization is a sequential model-based optimization (SMBO) that efficiently finds the optimal hyperparameters by iteratively using previously obtained hyperparameters for evaluating current combination of optimal hyperparameters [20]. Figure 8 shows iterative Bayesian optimization for hyperparameter tuning of random forest classifier. Bayesian optimization is inherently sequential, repeatedly applying equations (6) and (7) until convergence is achieved, as detailed in [21] :

D : observed dataset, f(x) : objective function at point x to be optimized, α(x/D) : acquisition function at point x.

$$x_{n+1} = \arg\max_{x} \alpha(x|\, D_n) \tag{7}$$

$$D_{n+1} = D_n \cup \{(x_n + 1, f(x_n + 1))\} \tag{8}$$

iter	target	colsam...	gamma	learni...	max_depth	n_esti...	subsample
1	0.2104	0.5247	4.754	0.2223	9.381	89.0	0.578
2	0.2102	0.3349	4.331	0.1843	10.91	55.15	0.985
3	0.2099	0.7995	1.062	0.06273	3.568	126.1	0.7624
4	0.2106	0.5592	1.456	0.1874	2.953	123.0	0.6832
5	0.2104	0.5736	3.926	0.06791	8.199	198.1	0.5232
6	0.2104	0.7752	1.622	0.2177	2.896	122.8	0.5724
7	0.2093	0.7891	1.336	0.1232	2.975	122.9	0.951
8	0.2104	0.4262	4.962	0.07203	10.81	173.9	0.6438
9	0.2107	0.7738	2.678	0.2695	7.403	155.6	0.8269
10	0.2104	0.7047	4.728	0.04113	11.0	173.6	0.7506
11	0.2157	0.7369	2.184	0.1323	1.426	68.91	0.8332
12	0.2157	0.5144	2.491	0.143	1.698	68.54	0.7988
13	0.2106	0.8503	2.392	0.2256	3.253	69.17	0.8814
14	0.2178	0.5748	1.681	0.01504	1.732	67.92	0.7147
15	0.2146	0.4132	1.093	0.2651	1.029	66.78	0.501

Figure 8 : Bayes_opt iteratively finds optimal hyperparameters (highlighted ones) in a Random Forest Classifier.

V. Results and Analysis

All base models built on the Bayesian optimized set of hyperparameters were trained on the same training set from the pre-processed data and the final ensemble model was evaluated on the test set. The experimental results and observations made are presented below.

A. Feature engineered dataset

Reduction to one-dimensional feature space supports real-time applications and contributes to more efficient data analysis pipelines, with the following key points :

1) Class Separation: The linear transformation maximizes class separation by projecting data onto a single dimension, highlighting differences between classes as observed in Figure 5.

2) Enhanced Interpretability: Simplifies complex data, making it easier to visualize and interpret class distinctions in the transformed feature space.

3) Performance Evaluation: Demonstrated notable performance improvements in subsequent classification tasks, with clear class boundaries facilitating faster training, reducing processing time.

B. Binary Classification

With soft voting, max voting classifier combined the predictions from naïve bayes, random forest, XGBoost, and logistic regression base models and achieved an overall accuracy of 78% and a precision of 82% for binary classification of malware and normal classes. The ensemble model leveraging the strengths of individual models, enhanced predictive performance through aggregated probabilities. Soft voting balanced bias and variance trade-offs yielding improved precision without significantly compromising accuracy.

C. Multiclass Clasification

The final prediction observed from the stack ensemble with XGBoost meta learner after leveraging the power of all the individual learners, namely decision trees, random forest, SVM, and KNN, was an accuracy of 93% with 92% precision. Stacking base models with a meta-model utilizes layered learning for efficiently balancing the bias-variance trade-off by merging high-bias, low-variance models with low-bias, high-variance models. Multi-class classification's accuracy and precision calculations were based on the confusion matrix shown in Figure 9 and evaluation metrics shown below.

TP : true positives or instances of predicted class matching the actual class, FP : false positives or all incorrect predictions for a given class, TN: true negatives or correctly predicted classes excluding the target class, FN: false negatives or instances of a class that are predicted incorrectly.

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

$$Recall = \frac{TP}{TP+FN} \tag{10}$$

$$F1\text{-}score = \frac{2*Precison*Recall}{Precision+Recall} \tag{11}$$

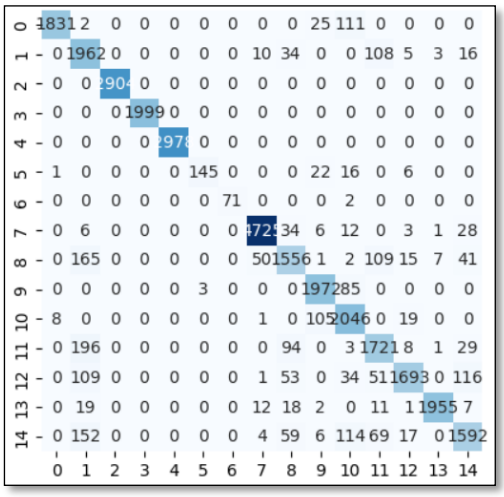

Figure 9 : Multi-class classification confusion matrix. 0-14 denote the 15 classes including 14 different malware attack classes and a normal class.

The performance comparison of ensemble learning with deep learning led to the outcome of stacking ensemble outperforming CNN model in terms of accurate multi-class classification. The diversity in the base learners enhanced the overall performance by mitigating individual weaknesses and stacking individual strengths. CNN model's accuracy on training dataset vs testing dataset across 15 epochs is shown in Figure 10.

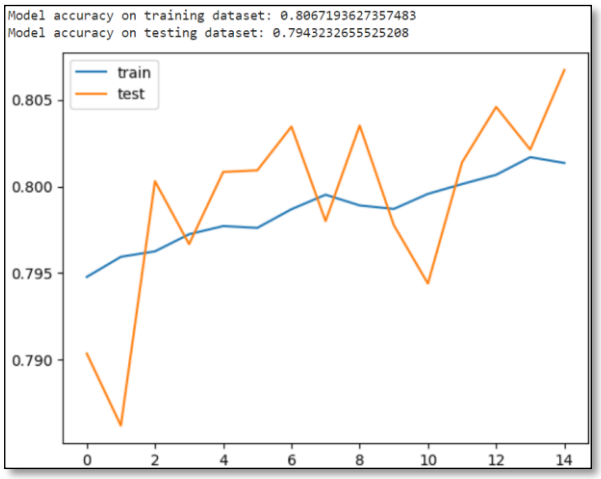

Figure 10 : CNN model's accuracy on training dataset vs testing dataset.

The proposed attestation framework utilizes advanced ensemble learning techniques to enhance the performance of both binary and multiclass classification tasks. These results underscore the effectiveness of ensemble approaches in leveraging the complementary strengths of various models, outperforming traditional attestation as well as deep learning methods and demonstrating their potential for complex classification problems.

VI. Conclusion

This paper proposes a novel runtime attestation framework for prediction of malicious IoT devices and malware attack types, attaining a secure IoT network. Utilizing a Bayesian optimized maximum voting classification ensemble model and a stack ensemble model, the framework achieves impressive accuracy and precision rates of 93% and 92%, respectively. The proposed solution effectively harnesses the strengths of diverse base models with XGBoost meta-learner to eliminate the need for a legitimate copy of the original firmware for multiclass classification and bridge gaps of traditional attestation methods. The efficacy of these ensemble models in handling intricate patterns and enhancing predictive performance is evident from the experimental results on a realistic cybersecurity dataset covering various IoT and Industrial IoT Perception Layer attacks.

References

[1] G. Coker *et al.*, "Principles of remote attestation,"*International Journal of Information Security*, vol. 10, no. 2, pp. 63–81, Apr. 2011.

[2] Cas Cremers *et al.*, "Formal Analysis of SPDM : Security Protocol and Data Model", Mar. 2023.

[3] R. C. A. Alves, B. C. Albertini, and M. A. Simplicio, "Securing hard drives with the Security Protocol and Data Model (SPDM)," in 2022 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Nicosia, Cyprus: IEEE, Jul. 2022, pp. 446–447.

[4] Cremers, Cas, Alexander Dax, and Aurora Naska. "Formal Analysis of {SPDM}: Security Protocol and Data Model version 1.2." In 32nd USENIX Security Symposium (USENIX Security 23), pp. 6611-6628. 2023.

[5] A. Niemi, Sampo Sovio, and J.-E. Ekberg, "Towards Interoperable Enclave Attestation: Learnings from Decades of Academic Work," Apr. 2022.

[6] W. A. Johnson, S. Ghafoor, and S. Prowell, "A Taxonomy and Review of Remote Attestation Schemes in Embedded Systems," IEEE Access, vol. 9, pp. 142390–142410, 2021.

[7] Remote ATtestation ProcedureS (Rats), n.d.

[8] OPC Unified Architecture Specification Part 21: Device Onboarding, 2022.

[9] M. Pritikin, M. Richardson, T. Eckert, M. Behringer and K. Watsen, "Bootstrapping remote secure key infrastructures (brski). Internet-Draft draft-ietf-anima-bootstrapping-keyinfra-45", November 2020.

[10] K. Watsen, I. Farrer and M. Abrahamsson, "Secure zero touch provisioning (sztp)", RFC 8572 Internet Engineering Task Force (IETF), April 2019.

[11] Wital Bartsch, *et al.*, FIDO Device Onboard Specification 1.1, April 2019.

[12] M. N. Aman, H. Basheer, J. W. Wong, J. Xu, H. W. Lim, and B. Sikdar, "Machine Learning Based Attestation for the Internet of Things Using Memory Traces," IEEE Internet of Things Journal, pp. 1–1, 2022.

[13] Ferrag, Mohamed Amine, et al, "Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning", 2022.

[14] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," 2011.

[15] Stamatios-Aggelos Alexandropoulos,Christos K. Aridas,Sotiris Kotsiantis, and Michael N. Vrahatis,"Chapter 7-Classification: a Tour of the Classics", 2020.

[16] Sangapati Pavan, "Ensemble Learning Techniques," Kaggle, 2020.

[17] Stamatios-Aggelos Alexandropoulos,Christos K. Aridas,Sotiris Kotsiantis, and Michael N. Vrahatis,"Stacking Strong Ensembles of Classifiers", May 2019.

[18] Rama Jayapermana,Aradea Dipalokareswara and Neng Ika Kurniati,"Implementation of Stacking Ensemble Classifier for Multi-class Classification of COVID-19 Vaccines Topics on Twitter",May 2022..

[19] Lavanya Gupta," Comparison of Hyperparameter Tuning algorithms: Grid search, Random search, Bayesian optimization", Nov 2020.

[20] Siddharth Ghosh," Hyper Parameter Tuning Techniques — Grid Search, Bayesian & Halving— Wonders of ML Realm", 2024,.

[21] Kenji Kawaguchi, Leslie Pack Kaelbling, and Tom´as Lozano-P´erez," Bayesian Optimization with Exponential Convergence", 2015.

[22] Tin Lai, *et al.*, "Ensemble learning based anomaly detection for IoT cybersecurity via Bayesian hyperparameters sensitivity analysis", Jun. 2024.

[23] Li Lisha, *et al.*, "Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization", Jun. 2018.