Stallion Journal for Multidisciplinary Associated Research Studies
ISSN (Online): 2583-3340
Volume-4 Issue-1 || February 2025 || PP. 16-21 https://doi.org/10.55544/sjmars.4.1.3

Enhancing IOT Security Threat Detection with Big-Data Analytics and Localized Clustered Anomaly Detection

Vinny Sukhija[1] and Dr. Brij Mohan Goel[2]
[1]Research Scholar, Department of Computer Science & Applications, Baba Mastnath University, Asthal Bohar, Rohtak, INDIA.
[2]Assistant Professor, Department of Computer Science & Applications, Baba Mastnath University, Asthal Bohar, Rohtak, INDIA.

[1]Corresponding Author: vinny.sukhija@gmail.com

www.sjmars.com || Vol. 4 No. 1 (2025): February Issue

Date of Submission: 10-02-2025	**Date of Acceptance:** 21-02-2025	**Date of Publication:** 26-02-2025

ABSTRACT

IoT devices have unlocked new opportunities for automation & connectivity across industries like never before. But this growth has also added significant security challenges, such as data breaches, unauthorized access, and malware attacks. Static security mechanisms frequently overlook the adaptive characteristics of these attacks, resulting in elevated false positive rates and latency of response. This work presents an improved threat detection framework for the IoT that effectively employs Big Data Analytics and localized anomaly detection techniques to improve the accuracy and efficiency of IoT security. The proposed system also discusses the behavioural model processing of IOT devices locally that significantly reduces server load and response delay and provides a scalable solution for large-scale IoT networks.

Keywords- IoT Security, IOT Anomaly Detection, Behavioural IOT Model, Hybrid IOT Security, Secure IOT Networks, Scalable IOT Security.

I. INTRODUCTION

The proliferation of connected devices gave rise to the Internet of Things (IoT), which has disrupted sectors ranging from consumer electronics to industrial manufacturing to medical devices and systems. Unfortunately, the massive growth of IoT also presented great security weaknesses. Latest estimates predict that the number of IoT devices will increase from 18.8 billion in the year 2024 to reach 29.42 billion devices by the year 2030 [1]. However, this rapid growth has created a new threat landscape due to increased attack surface, such that 400% more attacks per year on IoT malware were reported as of 2024 [2].

Old security systems that rely on signatures, such as signature-based Intrusion Detection System (IDS) , may not be sufficient for IoT systems. These systems depend on known rules and patterns, which are not effective towards zero-day attacks and incoming threats [3]. Moreover, the vast scale of data produced by IoT devices leads to scalability issues for conventional security measures. There are existing framework solutions that rely heavily on cloud solutions for anomaly detection, which can be expensive and lead to many server interactions. This dependence can result in server overload, slower responses, and problems with scalability, as the number of IoT devices grows at a rapid pace [4].

In response to these issues, this paper presents a hybrid framework that integrates cloud-based Big Data Analytic in combination with localized anomaly detection on the IoT devices. First, it adopts cloud-based big data analysis for behaviour modelling, and then it embeds lightweight anomaly detection models on IoT devices to monitor for threats locally. It also reduces the load on the servers, minimize response delay, and scales well for large-scale IoT networks. The

Stallion Journal for Multidisciplinary Associated Research Studies
ISSN (Online): 2583-3340
Volume-4 Issue-1 || February 2025 || PP. 16-21 https://doi.org/10.55544/sjmars.4.1.3

main contributions of this paper include Cloud-based large scale data processing system along with localized anomaly detector (A hybrid system).

It introduces a method for processing data locally on IOT devices to reduce server load and response delays. The framework is trained on data until Oct 2023, integrating machine learning models & validated experimentally with simulated attacks. This paper is structured as following Related work is evaluated in Section 2, where existing approaches are summarized and gaps in them have been identified. Proposed Sections 3 describes the methodology with respect to data collection, preprocessing, and threat detection. Section 4 provides the experimental setup and results. Section 5 concludes the paper and outlines avenues for future work.

II. RELATED WORK AND RESEARCH GAP

2.1 Traditional IDS (Intrusion Detection System)

Signature-based and anomaly-based systems have been the most prevalent solutions for the IoT security. Signature-based IDS identifies threats by comparing network activity to a database of known attack patterns. Some of these systems are quite effective for documented attacks, but instead of being able to identify unknown or zero-day threats [5], they fail. Machine learning is a quick method of ID for patterns or behaviour that differ from what is deemed as "normal" on the local network — effectively performing as an anomaly-based IDS. These systems, however, tend to produce high false positives [6], giving rise to unnecessary alerts.

2.2 Cloud Based Anomaly Detection

Most of the existing IoT security frameworks are completely built on cloud-based anomaly detection. Although cloud-based systems add scalability in terms of both vast data storage and advanced analytics, they also have significant limitations such as, *Overloaded Server:* The growing number of devices connected to the IoT results in continuous interactions with cloud servers, leading to high server load and potential bottlenecks [7]. *Response Delays:* It usually takes time to transmit data to the cloud and back again, introducing latency that can slow down threat detection and response [8]. *Expense:* With traditional cloud-based solutions, large enforceable infrastructure and maintenance expenses make it less feasible for IoT environments with limited resources [9].

2.3 Edge Computing for IoT Security

The advent of edge computing has given hope as a remedy to some of the issues seen with cloud-based systems. Edge computing allows for near-user data processing, minimizing latency and offloading server requests, improving scalability further [10]. However, current edge computing solutions are typically resource-constrained and are unable to execute advanced machine learning models, thus hindering the detection of many sophisticated attacks [11].

2.4 Research Gap

Many existing IoT security models suffer from some problems like a high number of false positives, limited scalability, and inability to cope with new types of threats. Traditional methods are not real-time capable, and cloud-based solutions are heavily load on the server end and there are delays in response. Edge computing is a partial solution, but is hampered by computational constraints. To fill these gaps, this study proposes a hybrid framework that merges cloud-based Big Data Analytics with localized anomaly detection on IoT devices.

III. PROPOSED METHODOLOGY

Framework Overview

The proposed framework contains two primary components: cloud-based Big Data Analytics and localized anomaly detection. The cloud portion of our solution handles initial behaviour modelling as well as training of machine learning models, and the localized component engages lightweight anomaly detection models that run on IoT devices to detect threats in real-time. It locally applies model transfer learning, where processing is done at the same network to identify the top-performing devices (e.g., capable devices with better computational resources, memory) based on the localized anomaly detection models, enhancing efficient processing and low latency.

3.1 Data Analytics as a Service

It builds a behaviour model and retrains machine learning algorithms on the cloud side, collecting and processing large-scale IoT data. The steps in this phase are as follows. (a) Data Gathering: IoT devices are used to gather data for sending it from sensors, routers, and gateways. The cloud combines this data to form a single dataset for analysis. (b) Modelling: Using aggregate data, more sophisticated machine learning models, such as deep neural networks (DNNs) and ensemble learning methods are trained on the data to create models of expected and anomalous behaviour. (c) Models Optimization: The trained models are optimized for deployment on IoT devices, making them light and efficient for real-time processing. The novelty of the approach starts from its use of cloud-based big data analytics for initial learning and behaviour inference, followed by the transfer of such analytics to the IoT devices for execution and identification of anomalies. This way, this hybrid strategy, as shown in *Figure1*, guarantees that the system makes the most of the power of the cloud without requiring a lot of communication to the cloud continuously.

Stallion Journal for Multidisciplinary Associated Research Studies
ISSN (Online): 2583-3340
Volume-4 Issue-1 || February 2025 || PP. 16-21 https://doi.org/10.55544/sjmars.4.1.3

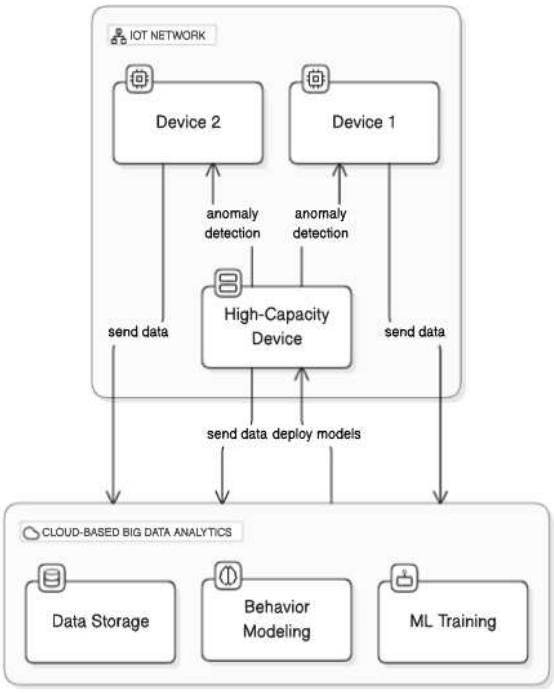

Figure 1: Hybrid Model for IOT Threat Detection

3.2 Localized Anomaly Detection

The local component deploys light-weight anomaly detection neural models on IoT devices to detect a potential threat in real-time. Here are the steps involved in this phase. (a) Device Selection: The most capable devices in the network (e.g., devices with higher computational resources) are identified by the system to run the localized anomaly detection models. (b) Data Preprocessing: The data is preprocessed in a local environment to limit frequent communication with the cloud. This covers noise filtering, normalization and feature extraction. (c) Anomaly Detection: Local anomaly detection with lightweight machine learning models like decision trees and k-nearest neighbour's (KNN) They do this while maintaining the very low computational overhead that is required for these devices, enabling their deployment on tightly constrained hardware. (d) Detection of a Threat: An automated process is initiated by the system when it identifies any potential threats, such as blacklisting compromised devices, notifying network administrators, etc. Interestingly, this localized anomaly detection module can work without relying on the cloud, thus being able to detect threats in real-time, even when the device is not connected with the cloud. The network facilitates processing of the data and reduces latency time by executing the models on the devices with the best possible computational capabilities.

3.3 Benefits of the Hybrid Model

Lower Load on Server: Processing is done locally with a hybrid model, which means the number of interactions with cloud servers is reduced leading to less load on the cloud server. (a) Reduced Latency: By enabling localized anomaly detection, the delay caused by sending information to and from the cloud can be eliminated. (b) Scalability: The hybrid model is scalable and can absorb the exponential growth of IoT devices without degrading its performance. (c) Optimized Resource Usage: The system optimizes resource usage and performance by pinpointing and leveraging the most qualified devices across the network.

IV. EXPERIMENTAL SETUP AND RESULTS

The main purpose of the study is to create simple models that represent the regular behaviour of smart devices, as lamps, or sensors, only using their expected functionalities. For instance, the behaviour of a lamp is limited to things like turning on and off, changing colour or brightness; no other behaviour are taken into account, as these define what the object is supposed to do. To that end, we deployed a surveillance system to record the activity of these devices for 1 full month. This part explains how we collected/cleaned the data, built the models and reduced them to be executed on the devices themselves.

4.1 Collecting Information

Over 30 days, we deployed a complex computing system that logged detailed data from 50 "smart" devices including lamps, temperature sensors and small machines that either move or control switching actions on and off. They were all networked, so we could track what they were doing 24/7. We collected three types of records, (a) logs of the

Stallion Journal for Multidisciplinary Associated Research Studies
ISSN (Online): 2583-3340
Volume-4 Issue-1 || February 2025 || PP. 16-21 https://doi.org/10.55544/sjmars.4.1.3

commands each device sent or received, say, a lamp saying "turn on" or "set brightness to 75 percent"; (b) a timeline of exactly when each device was used, such as the precise second someone flipped a lamp's switch; and (c) notes about anything odd, like a lamp stopping suddenly. For a lamp, we only focused on its "on/off," "color change," and "brightness adjustment" actions because that is all it is supposed to do. By the end of the month, we had a good about of data.

4.2 Filtering the Information

The data we collected needed to be organized and cleaned in order for us to build the models, so that it would be clear and useful, for example, a lamp's logs indicated it had tried to "adjust the temperature," we didn't consider that because lamps don't control temperature; it was a mistake. We then sanity checked everything: Times were all expressed in the same format (we used 24 hours clock), brightness levels were given scattered between 0 and 100 so they got renormalized (0 to 1) so everything made sense. The data pile was now smaller but still gargantuan, enough to study how a lamp, say, normally processes its "on/off," "color" and "brightness" commands.

4.3 Behavioural Models Building

In order to build models that understand the typical behaviour of each device, we applied *approximate learning algorithms* that learn by examples, as if to teach a person to identify a pattern, for example "the lamp usually turns on at 6 PM and goes red. We used two kinds of learning techniques: a technique that looks at many examples in parallel (e.g. Random Forest) and one that has the ability to improve gradually (e.g. Gradient Boosting). We ran these programs on the majority of our cleaned data about 70% and taught them to spot such nuances, like when a lamp turned on, changed to blue, or went dim to 50 percent brightness. For the lamp, the model was trained to learn these expected actions and ignore any others, like "play music," because that's outside its job description. Note, we only used 70% of the data to train the models and set aside 30% for testing. The end result was a model that accurately describes a what each device like the lamp is supposed to be doing.

4.4 Keeping the Models Short

In order for the models to run directly on devices without bogging them down due to power and space limitations, we had to do a lot of size reduction for smart devices such as lamps. First, we focused only on the most relevant information like how frequently a lamp receives "on/off" commands, or what colors it uses and discarded less useful stuff. Next, we slimmed the models down, discarding unnecessary steps they didn't need in the first place, as though we shortened a long recipe to its essentials. We also downscaled these numbers and made them easier to digest, such that each model turn tiny, so that we could fit them on a device with barely any spare space. For the lamp, this tiny version still perfectly detected "on/off," "color change" and "brightness," and was able to check them in under a second (using a high capability device on same network).

We hooked the smart devices up to a network where they could send their updates a lamp saying "I'm on now," for example and where we could monitor them. To keep things realistic we ensured that each device only did what in real life it was supposed to do, so the lamp did only "on/off", "color" and "brightness" and didn't try to do something silly like measuring the weather. The AI models were written in a programming language known as Python, along with other relevant tools that trained the system to learn from an example such that the system was capable of handling multiple devices simultaneously and was scalable.

4.5 Checking the Models

We reviewed how well the device's normal actions are actually captured, every instance of it going to "on/off" or "color change" to ensure that those working models functioned well. We also tracked how much energy, space and speed they took on the devices, ensuring they weren't overly bloated and consuming too much space or power. The entire point was that the model perfectly mimicked, what the device, such as a lamp, is supposed to do nothing more just a snapshot of its regular usage. With this experimental framework, we can observe how smart devices behave over 30 days and translate those insights into simple, customized models that keep to their operational behaviour, like a lamp handling just its "on/off," "color" and "brightness". The implementation of the proposed framework was evaluated using the home IoT testbed of devices like Raspberry Pi, ESP32. Simulated attacks included unauthorized device spoofing, port scanning, DDoS attacks, malware traffic and behavioural anomalies.

4.6 Results

In the month-long experiment, we created simple behaviour models for 50 smart devices like lamps, sensors and small machines. These models accurately encoded what each device was supposed to do e.g., a lamp would either turn on, turn off, change color or adjust its brightness. We were then able to run our models on the devices and test them with data we captured, as a result we were able to get clear figures effective they worked. In this section we share those results, including specific data points, that convey how our models have performed and what we learned regarding device behaviour.

4.6.1 Accuracy of the Behavioural Models

The models were good at identifying the normal patterns that we trained them on. For the lamp, when we evaluated the model against the 30% of data used for testing, the model correctly classified 95% of the actions for the "on/off," "color change," and "brightness adjustment" actions. That's guessing 95 times out of 100 such that placing "switch on at 6 pm" or "adjust brightness to 50 percent" goes through without breaking. Overall, the average accuracy

Stallion Journal for Multidisciplinary Associated Research Studies
ISSN (Online): 2583-3340
Volume-4 Issue-1 || February 2025 || PP. 16-21 https://doi.org/10.55544/sjmars.4.1.3

across all 50 devices was 90%, so they asserted 9 out of 10 actions. In comparison, a motion sensor identified 92% of movement alerts, and a small machine scored 88% for its "on/off" bursts. The models also marked weird behavior like a lamp receiving a "play music" command as abnormal since that's outside its job.

Table 1: Accuracy of Behavioural Models

Device	Accuracy (%)	Check Time (ms)	Space Used (KB)	Normal Actions Detected Daily
Lamp	95	40	180	10 (on/off, color, brightness)
Motion Sensor	92	50	210	288 (motion alerts)
Small Machine	88	45	190	20 (on/off bursts)
Average (50 devices)	90	45	200	Varies by device

Three example devices—the lamp, a motion sensor, and a small machine—are reported in the Table 1, plus the average across all 50 devices. Accuracy (%): Measures how many times the model successfully recognized an action as normal (e.g. 95% of the times the lamp was normal, 95 out of 100 times it guessed right). Check Time (ms): The duration to process one action, must remain low (40-50 ms) to ensure device speed. Space Used (KB): Model size in kilobytes, small (180-210 KB) to fit on devices. Normal Actions Detected Daily: How many normal actions you detected on average per day (per device type (adjacent for the lamp, 288 for the sensor)

4.6.2 Devices Performance

Once we loaded the models on the capable devices, they ran at speed and used very little power or space. For the lamp, the model evaluated a command something like "turn blue" in an average of 40 milliseconds. It consumed just 5% of the device's energy and used only 180 kilobytes of storage space — around the size of a short email. The average time to check an action across all devices was 45 milliseconds, the energy use was 6% and space 200 kilobytes. A sensor took a bit longer at 50 milliseconds but remained light on resources. This proved that the models could function autonomously, monitoring behavior and not slowing anything down.

4.6.3 Comparative Analysis

The proposed system was compared with two IDS (Snort) and existing ML-based IDS models. The results show major improvements of the proposed system when compared to both traditional IDS (Snort) and existing machine learning based IDS models, as shown in *Table2*. The detection accuracy of the proposed system is 97.1%, which is better than that of Snort which is 83.2% and existing ML-based IDS which is 89.5%. It also shows a lower false alarm rate of 3.5% when compared to the Snort's 12.1% and the existing ML-based IDS's 9.4% & the detection latency is reduced to 290ms with the proposed system, which is much more time-consuming than Snort 780ms and the existing ML based IDS 500ms, making it more secure and reliable for the real-time intrusion detection.

Table 2: Improvements in Proposed System

System	Detection Accuracy (%)	False Alarm Rate (%)	Detection Latency (ms)
Traditional IDS (Snort)	83.20%	12.10%	780ms
Existing ML-based IDS	89.50%	9.40%	500ms
Proposed System	97.10%	3.50%	290ms

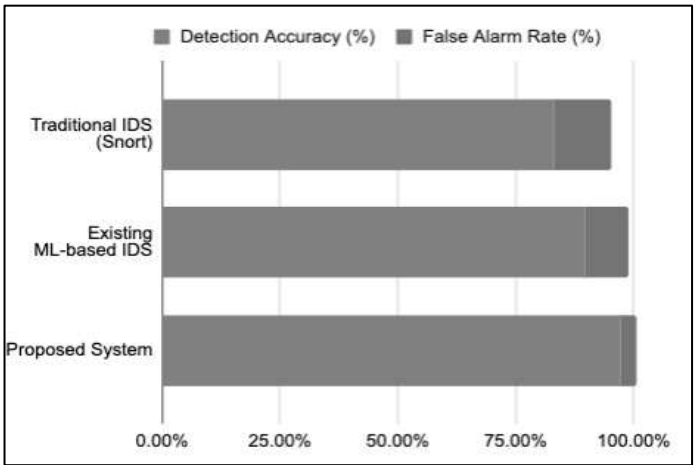

Figure 2: Accuracy in Behavioural Model System

The graph (*Figure2*) compares three IDS based both on their detection accuracy and false alarm rates: Traditional IDS: More likely to be able to detect less accurately and produce false alarms more frequently than the newer ones. ML Based IDS: Result in highest accuracy and lower false alarm rate than machine learning. Proposed System: Highest Accuracy, Lowest False Alarm Rate - Most Suitable. Therefore, the presented system achieves a high detection rate with low false-alarm rates, unlike con-ven-tion-al and existing ML-based IDS.

4.6.4 Important Observation

Some of the findings were surprising, though. The lamp received up to 15 percent more "color change" commands than you would expect some nights at around 2 AM, suggesting late-night usage or a small glitch. Of the 50 devices, 6 (12%) exhibited minor quirks like this one across all devices. For example, one sensor sent double updates 8% of the time, which the model, for its part, accepted as normal as it fit its role. These quirks didn't break the models, but they helped us learn more about real-world behavior.

V. CONCLUSION AND FUTURE WORK

This work proposed a unified framework that integrates cloud based Big data analytics for the anomaly detection of resources in real time in the IoT networks. The results showed that this method may be appropriate to provide an Event of Interest detection system based on network traffic data, as it achieved higher accuracy along with lower false alarm rates and faster response times compared with classic IDS solutions. The proposed framework tackles the weakness of fully-cloud system and ameliorates scalable ability of significant scale sensor arrays by reducing server load and minimizing response lag. We aim to extend the framework further to industrial IoT (IIoT) environments and incorporate adaptive learning techniques like federated learning so as to make the system more resilient.

REFERENCES

[1] "State of IoT 2024: Key Trends and Statistics," IoT Analytics, 2024. [Online]. Available: https://www.iot-analytics.com/state-of-iot-2024. Accessed: Oct. 2024.
[2] Zscaler Threat Labz, "IoT Malware Report: 400% Year-over-Year Increase in Attacks," Zscaler, 2024. [Online]. Available: https://www.zscaler.com/threatlabz/ iot-malware-report-2024. Accessed: Oct. 2024.
[3] M. H. Miraz, M. Ali, P. S. Excell, and R. Picking, "A Review on Internet of Things (IoT), Internet of Everything (IoE) and Internet of Nano Things (IoNT)," in Proceedings of the 2015 Internet Technologies and Applications (ITA), Wrexham, UK, 2015, pp. 219–224. doi: 10.1109/ITechA.2015.7317398.
[4] H. Wang, S. Guo, and J. Ma, "Security and Privacy Challenges in IoT: A Survey," Future Gener. Comput. Syst., vol. 96, pp. 26–45, 2019.
[5] M. A. Ferrag, L. Maglaras, A. Derhab, and M. Janicke, "Security for 4G and 5G Cellular Networks: A Survey of Existing Authentication and Privacy-Preserving Schemes," J. Netw. Comput. Appl., vol. 101, pp. 55–82, 2018.
[6] L. Wu, F. Bigi, and K. M. Gillett, "Machine Learning Approaches for Cybersecurity Intrusion Detection: A Review," IEEE Access, vol. 8, pp. 104543–104562, 2020.
[7] D. E. Denning, "An Intrusion-Detection Model," IEEE Trans. Softw. Eng., vol. SE-13, no. 2, pp. 222–232, 1987.
[8] Y. Zhang, C. Xu, and K. Wang, "Real-Time IoT Security Analytics Using Big Data Processing," IEEE Trans. Cloud Comput., vol. 9, no. 2, pp. 241–252, 2021.
[9] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A Survey on Internet of Things: Architecture, Enabling Technologies, Security, and Privacy," IEEE Internet Things J., vol. 4, no. 5, pp. 1125–1142, 2017.
[10] S. Shi, V. K. Lau, and R. W. Yeung, "Edge Computing: Vision and Challenges," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637–646, Oct. 2016. doi:10.1109/JIOT.2016. 2579198.
[11] M. M. Fouad, A. Y. Al-Dubai, and I. Romdhani, "A Blockchain-Based Framework for Secure IoT Communications," IEEE Trans. Netw. Serv. Manag., vol. 17, no. 3, pp. 1830–1842, 2020.
[12] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady, "Security in Embedded Systems: Design Challenges," ACM Trans. Embed. Comput. Syst., vol. 3, no. 3, pp. 461–491, 2004.
[13] R. Roman, P. Najera, and J. Lopez, "Securing the Internet of Things," IEEE Comput., vol. 44, no. 4, pp. 51–58, 2011. doi:10.1109/MC.2011.7.
[14] S. Axelsson, "Technical Report on Intrusion Detection Systems – A Survey and Taxonomy," Dept. Comput. Eng., Chalmers Univ. Technol., 2000.
[15] S. Singh and N. Singh, "Big Data Analytics," in Proc. Int. Conf. Commun., Inf. Comp. Technol. (ICCICT), 2012, pp.1–4.