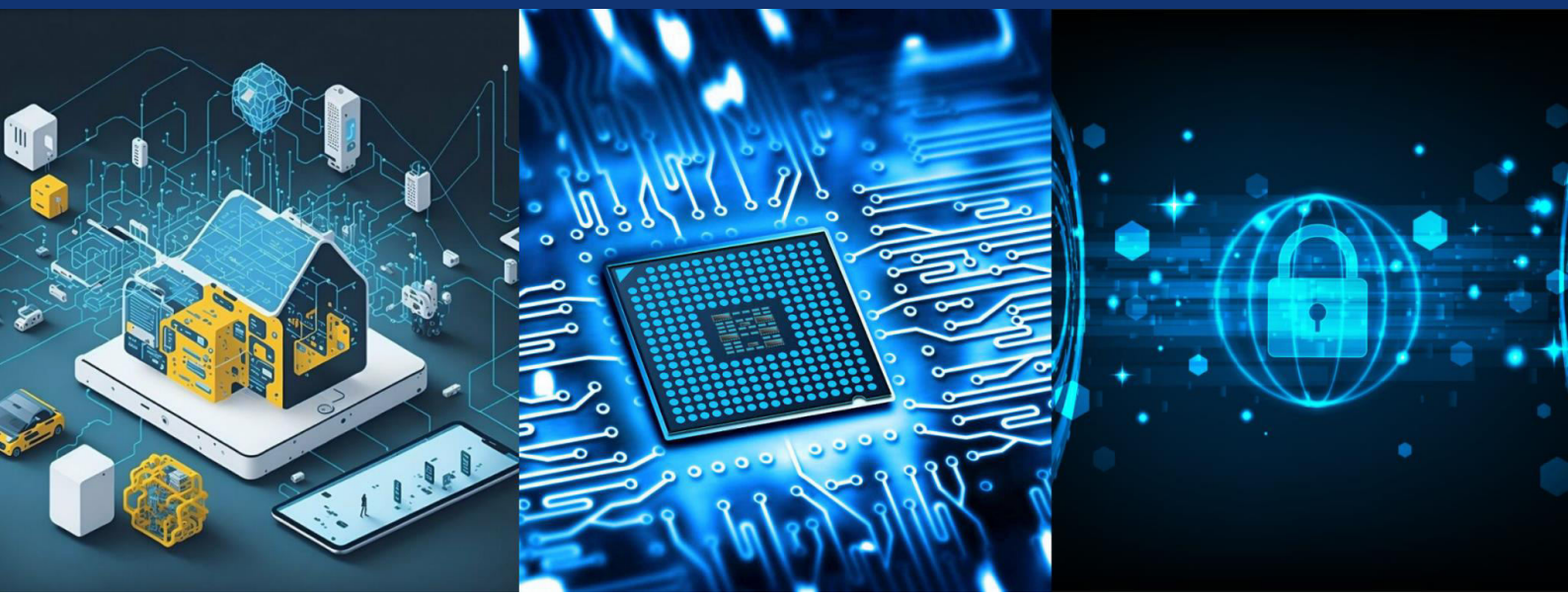
IJIRCCE

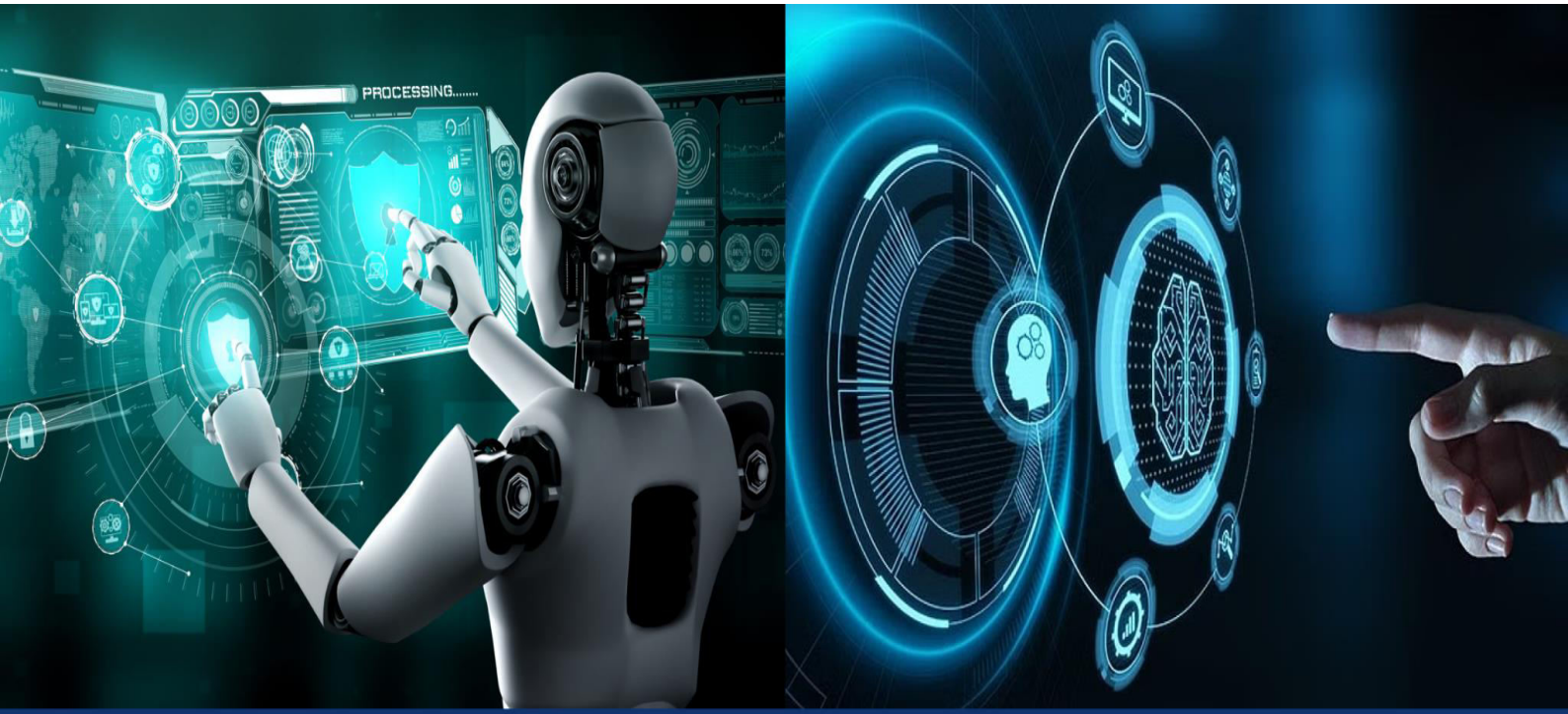
ISSN(O): 2320-9801

ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 9.927

Volume 14, Issue 5, May 2026

© 2026 IJIRCCE | Volume 14, Issue 5, May 2026 DOI: 10.15680/IJIRCCE.2026.1405040

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

BlueHunt: A Forensic Framework for Detecting Rogue Bluetooth and IoT Devices in Sensitive Environments

Madhumitha M, Mr. Sankara Narayanan S T

M.Sc. Cyber Forensics & Information Security, Dept. of Cyber Security, Dr. M.G.R. Educational and Research Institute, Maduravoyal, Chennai, Tamil Nadu, India

Assistant Professor, Dept. of ISDF, Center of Excellence in Digital Forensics, Perungudi, Chennai, Tamil Nadu, India

ABSTRACT: BlueHunt is a passive framework to detect rogue Bluetooth and IoT devices in sensitive environments. BlueHunt features real-time Bluetooth Low Energy (BLE) scanning, manufacturer identification based on IEEE OUI numbers, a threat scoring engine, and an Isolation Forest machine learning model to detect rogue devices in real time. The system uses a Flask-SocketIO web dashboard to provide monitoring of rogue devices, management of alerts and whitelists, and generation of forensic reports of detected devices. Experimental results show that BlueHunt can detect 100% of rogue devices in a sensor network with zero false positives for high-risk device classifications.

KEYWORDS: Bluetooth Low Energy (BLE); IoT Security; Rogue Device Detection; Isolation Forest; Anomaly Detection; Passive Scanning; Forensic Framework; Wireless Security.

I. INTRODUCTION

Bluetooth is already being used on the 2.4 GHz ISM band so they are a lot of other wireless protocols sharing the same band. BLE devices send out adevertisement packets as often as every 20 milliseconds and in passive listening mode, any nearby bluetooth adapter can detect them. This does not stop, however, from no widely-used enterprise security solution routinely monitoring the BLE spectrum for unauthorised devices. The forensic implications are enormous: it could be weeks or months before a rogue device is discovered and the presence of rogue device could only be confirmed by a physical inspection rather than by electronic monitoring.

In this paper, the authors propose Bluehunt, an integrated passive forensic framework to tackle this security problem. The framework is forensically transparent and undetectable as it passively collects BLE advertisement packets and WI-Fi probe requests without sending any data on the wireless spectrum. Discovered devices are analysed by a dual-layer detection pipeline: a rule-based heuristic scoring engine that scores 7 weighted risk indicators, and an Isolation Forest machine learning model that is able to discover statistical anomalies in device behaviour patterns. Results are displayed on a real-time web dashboard and one-click forensic report generation.

The main findings of this research are summarized below. The passive BLE scanning architecture is implemented on Windows without the need of special privileges or hardware and, for the first time, using the Bleak asynchronous library. Second, having a complete OUI-based manufacturer identification system would match the first three octets of each device's MAC address to the IEEE list of manufacturers, which would be a first "risk signal". Third, a seven factor weighted heuristic scoring engine categorizes the device as LOW, MEDIUM or HIGH risk, depending on manufacturer registration, proximity to the signal, presence of the device name, historical novelty, proximity to service UUID profile, MAC randomisation and compound behaviour patterns. Fourth, a second layer of detection is provided by an Isolation Forest anomaly detector, which was trained on feature vectors of devices that have been observed, detecting threats that do not match any known rule pattern. Fifth, a Flask-SocketIO dashboard provides real-time device discovery feeds, alert management, whitelist administration and automatic HTML and JSON forensic reporting.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

II. LITERATURE SURVEY

Becker et al. [1] provided essential research showing that the BLE advertisement packets could be passively received and tracked, allowing for constant tracking of the device even when not paired to other devices or over time. They found that early BLE implementations had MAC addresses which were static, meaning that a specific device could be tracked across different operating contexts, thereby raising privacy concerns. The results of this research formed the technical foundation for passive monitoring via BLE and inspired the use of MAC address randomisation in consumer devices as well as the creation of security-monitoring frameworks like BlueHunt that exploit passive monitoring To investigate MAC address randomisation as a privacy protection method in BLE devices, Fawaz et al. [2] carried out a systematic study on it. They showed that it is possible to be able to re-identify them despite address randomisation due to consistent patterns in the manufacturer-specific advertisement data, service UUID profiles, and timing characteristics. This is highly relevant to BlueHunt's threat model: As per the analysis, a random MAC address is considered a threat indicator because it could be a sign of active anonymisation by a rogue device, whilst a random-looking MAC address may be a legitimate device identified by its consistent behaviour which can then be added to the whitelist.

Ryan [3] did an early and extensive study on the security flaws of BLE and showed how to perform the following attacks in practice: passive eavesdropping, man-in-the-middle and device impersonation. His research showed that although the BLE pairing and encryption model is better than classic Bluetooth, advertising traffic is still disabled with no encryption. BlueHunt takes advantage of this property to do monitoring; BLE advertisements are broadcast by nature.

Das et al. [4] suggested a signature based Bluetooth intrusion detection system which stores the database of known malicious device signatures and checks them against the signatures found in Bluetooth scanning. While their approach is highly accurate for known threat signatures, it is not capable of detecting new rogue devices with characteristics different from any of the signatures. This restriction directly drives BlueHunt to include unsupervised anomaly detection, which is the ability to detect statistically atypical devices without having to know their characteristic properties.

Liu et al. [5] presented a new algorithm called Isolation Forest for anomaly detection in high dimensional data. They put forward the principle that unusual observations can be separated more readily by using random partitioning than can normal observations, a principle that has been found to be useful in a wide variety of phenomena. The work by Chen et al. [6] further extended the use of Isolation Forest to network intrusion detection, achieving better performance than one-class SVM and Local Outlier Factor on small networks traffic datasets that were unlabelled, a direct similarity to the BlueHunt deployment scenario.

III. METHODOLOGY / APPROACH

The BlueHunt is implemented as a modularised Python framework comprised of seven modules, arranged in a four-layer pipeline. The architecture of the overall system is shown in figure 1. The framework can be executed on Windows 10/11 with a Bluetooth adapter and does not need any special hardware, custom firmware or higher system privileges compared to the average user.

A. System Architecture Overview

The data are pipelined in BlueHunt. The BLE Scanner and Wi-Fi Scanner modules are used to receive advertisement data from wireless spectrum and convert it into DeviceRecord objects which are structured data containers containing all the attributes observed by the device. The DeviceRecord objects are sent to the ThreatScorer for heuristic analysis and then to the AnomalyDetector for evaluation using machine learning. The AlertManager processes the scored records and alerts if a high-risk finding is detected. All results are stored in the Database module using a local SQLite store. The Dashboard module provides real-time visualization via a browser interface and creates forensic reports on-the-fly.

B. Passive BLE Scanning

The scanner is implemented in python asynchronously. The BleakScanner context manager registers an advertisement callback function to be called for every advertisement packet received. The callback is used to retrieve the following fields from each packet: MAC address (48-bit device identifier); device name (from Complete Local Name or

© 2026 IJIRCCE | Volume 14, Issue 5, May 2026 DOI: 10.15680/IJIRCCE.2026.1405040

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Shortened Local Name advertisement data type); RSSI (dBm); Manufacturer Specific Data (company identifier and payload bytes); Complete List of 16-bit, 32-bit and 128-bit UUIDs; and TX Power Level (if present).

C. Wi-Fi Probe Request Capture

Complementary to BLE scanning, BlueHunt captures Wi-Fi probe request frames using the Scapy packet manipulation library with Npcap for Windows packet capture. Probe requests are management frames (type 0, subtype 4) broadcast by Wi-Fi devices searching for previously connected networks. They contain the source MAC address and optionally the SSID of the target network. Probe request capture enables detection of devices that are not currently advertising via Bluetooth but are actively scanning the Wi-Fi spectrum, expanding the framework's detection coverage.

D. MAC Address Analysis

MAC address analysis is a central component of the BlueHunt threat assessment pipeline. The 48-bit MAC address is examined at two levels. At the OUI level, the first three octets (24 bits) are matched against the IEEE OUI registry. An unregistered OUI indicates either an uncertified device or a deliberately fabricated MAC address, both of which warrant elevated suspicion.At the individual address level, the locally-administered bit (bit 1 of the most significant octet) is examined to detect MAC randomisation. A set locally-administered bit indicates the address was generated locally rather than assigned by the manufacturer, which is the mechanism used by iOS, Android, and Windows for privacy-preserving MAC randomisation. The detection is implemented as: is_random = bool(int(mac[0:2], 16) & 0x02). While MAC randomisation is employed legitimately by consumer devices for privacy, its combination with other suspicious indicators significantly elevates the risk profile of a device.

E. Heuristic Threat Scoring Engine

The ThreatScorer module implements a weighted additive scoring system that evaluates each DeviceRecord against seven independent heuristic rules. The rules are applied sequentially, and their penalty scores are accumulated. The total score is capped at 100 to normalise the output range. Table I presents the complete rule set with associated penalties and rationale.

Heuristic Rule	Score	Security Rationale
Unknown OUI	+30	MAC prefix unregistered with IEEE; indicates uncertified or spoofed device
Randomised MAC	+20	Locally-administered address suggests deliberate anonymisation
No Device Name	+20	Legitimate devices typically broadcastidentifiable names
Strong Signal >-50dBm	+15	Unusually strong RSSI indicates device concealed in close proximity
New Device	+10	Device has no observation history in the database
Suspicious Service UUID	+15	Nordic UART, Eddystone beacon, or raw serial protocols detected
Compound: Close+Unnamed+Unknown	+20	Combination characteristic of deliberately concealed covert devices

Table I: Heuristic Threat Scoring Rules

The final score maps to three risk tiers: LOW (0-29), MEDIUM (30-59), and HIGH (60-100). AlertManager is triggered by devices that are MEDIUM or HIGH. Whitelisted devices are routed completely bypass and always assigned to the LOW category, allowing operators to pre-register authorised infrastructure devices for deployment without having to be score.

© 2026 IJIRCCE | Volume 14, Issue 5, May 2026 DOI: 10.15680/IJIRCCE.2026.1405040

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

F. Isolation Forest Anomaly Detection

The algorithm creates a number of isolation trees formed by repeatedly choosing a random feature and a random split value in that feature's range. The path length between a node and the root of the tree is used as an approximation of the anomaly score: a short path length means that the node can be easily isolated and thus that the probability of the existence of an anomaly is high. The final anomaly score is the normalised average path length of the ensemble of trees.

The anomaly detector uses six numerical features for each DeviceRecord in the database: (1) RSSI value in dBm; (2) MAC randomisation indicator (binary); (3) device name presence indicator (binary); (4) manufacturer registration status (binary); (5) service UUID count; and (6) TX power level in dBm. The use of z-score normalisation on features is to standardise the features so that features with a larger absolute range do not dominate the partitioning process, such as RSSI.The model is set up with 100 estimator trees, contamination parameter set to be 0.1 and fixed random seed is chosen to be 42 for reproducibility. Training on all the devices that are collected in the database, minimum of 10 observations before the model is reliable. The model is automatically re-trained after each scan session and persisted to disk using Python pickles to allow the model to be deployed as soon as it has been re-trained in the next scan session without retraining. The result of the decision function ranges from around -0.5 (strongly anomalous) to +0.5 (strongly normal) and is linearly transformed to a 0-100 anomaly score to be displayed in a consistent manner with the heuristic scoring system.

G. Alert Management and Database

The AlertManager module is an implementation of alerting based on thresholds with per device cooldown suppression. All devices with MEDIUM and HIGH risk will trigger an alert on initial detection. Every subsequent scan cycle of persistently broadcasting devices is prevented from triggering alerts by the optional cooldown window (default 10 minutes). All alerts are saved to the SQLite alert table with extended details such as MAC address, risk level, score, flag list, message and time. The dashboard can be fed alert events through Socket with an optional callback interface.IO for browser notification in realtime.

H. Dashboard and Forensic Reporting

The dashboard is created using Flask web application and Socket.IO for bidirectional, real-time communication. The scanner thread flushes DeviceRecord updates to all the browser clients that are connected through Socket.IO events on devices as discovered – no polling latency, device list is always up to date based on scan state, within sub-second latency. The dashboard includes four functional panels: Devices (live risk sorted device table with filtering by risk level), Alerts (live alert feed with colour coding by risk level) and Whitelist (device registration interface with add and remove operations) and Reports (forensic report generation and download).ReportGenerator module generates forensic reports in HTML and JSON format, with the Jinja2 templating engine.

IV. RESULTS & DISCUSSION

BlueHunt test case was conducted in a controlled office space to determine the accuracy of detection, system performance, real-time response time, and to compare with state-of-the-art tools. The testing area consisted of a 40 sq m office with 12 testable devices (laptop, smartphones, wireless headset, a smartwatch and a wireless keyboard and mouse). The following rogue devices were secretly added: a commercial BLE beacon module with a randomised MAC address and no device name, a Bluetooth serial communication module (HC-08) broadcasting the Nordic UART Service UUID, and a commercial IoT environmental sensor using an unregistered OUI.

A. Detection Accuracy

Detected and classified as HIGH risk all three planted rogue devices were detected during the first scan cycle (10 seconds of scanning). The complete detection results of all the devices observed during the evaluation is shown in Table II.

Device	Risk	H-Score	ML Flag	Primary Flags
BLE Beacon (rogue)	H	85/100	Y	RANDOMMAC,NO_NAME, UNKNOWN_OUI, COMBO
HC-08	H	75/100	Y	SUSPICIOUS_

© 2026 IJIRCCE | Volume 14, Issue 5, May 2026 DOI: 10.15680/IJIRCCE.2026.1405040

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

BT Module (rogue)				UUID, NO_NAME, NEW_DEVICE
IoT Sensor (rogue)	H	70/100	Y	UNKNOWN_ OUI, STRONG_SIGNAL, NEW_DEVICE
Developer Laptop	L	10/100	N	NEW_DEVICE
iPhone (randomised)	M	30/100	N	RANDOM_MAC
Android Phone	L	20/100	N	RANDOM_MAC
Wireless Headset	L	10/100	N	NEW_DEVICE
Smartwatch	L	15/100	N	NEW_DEVICE
BT Keyboard	L	10/100	N	—

Table II: Detection Results for Evaluation Environment

Whereas: H-High , M- Medium, L-LOW, Y-Yes, N-No
The heuristic scorer was able to classify all three rogue devices as HIGH. The BLE beacon, with no signal strength penalty, because the beacon was placed across the room, got the highest score of 85/100, due to the combination of unknown OUI (+30), randomised MAC (+20), absent device name (+20) and the compound covert device flag (+20). The HC-08 module earned the 75/100 score mainly for its suspicious Nordic UART UUID (+15), no device name (+20), and new device status (+10) features, as well as bonus compound scoring. The IoT sensor achieved 70/100 with an unregistered OUI (+30), a good proximity signal (+15) and new device status (+10).

B. Performance Metrics

To evaluate the utilisation of system resources, continuous scanning was performed on a test system consisting of an Intel Core i5-1135G7 processor and 8GB of RAM and running Windows 11. The CPU utilization during the BLE scanning phase was below 4% on average, even reaching a maximum of 8% in a few cases during the write operation of the database after high density scan cycles. Memory used during long scans remained fairly constant at around 85MB. All these numbers show that there is nothing wrong with running BlueHunt on a lightweight laptop running concurrently with other work.

BLE advertisement packet reception was measured by dashboard responsiveness defined as the time between reception of BLE advertisement packet and the appearance of the device in the browser interface. 100 measurements were taken, with a mean latency of 147ms and a 95th percentile of 218ms. This latency includes BLE stack latency, latency due to Python callbacks, OUI lookup latency, heuristic score latency, isolation forest prediction latency, sqlite write latency, socket latency.IO event emission. The average latency is less than 200ms, which is a true real-time response for operational use.

The training time for the Isolation Forest model was evaluated with different datasets sizes. It took 0.31 seconds to complete the training that was done on 15 device observations. The time needed for training was 0.48 seconds when observing 50 times. The time needed to train on 200 observations was 1.2 seconds. All training times are suitable for retraining after operations without disrupting operations.

V. CONCLUSION

BlueHunt has a four-layer detection pipeline that includes Passive BLE and Wi-Fi scanning, IEEE OUI-based manufacturer identification, a seven-factor weighted heuristic threat scoring engine, and an unsupervised statistical anomaly detection machine learning model known as Isolation Forest. The dual-layer detection architecture offers complementary coverage: the heuristic layer can efficiently detect the devices that match the known suspicious patterns; the Isolation Forest layer can detect the statistical outliers that are out of the known rule patterns, which reduces false negative and the operational blind spots.

The passive operation of BlueHunt is a forensically important property, as it does not send probe requests and does not create any connections with the devices it is scanning, thus it will not affect the wireless environment or be detected by

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

the devices being scanned. The feature is ideal for hiding in places such as high-security facilities for continuous environmental monitoring, as well as in the aftermath of an incident for forensic investigation and for covert security assessments.BlueHunt will be expanded on in a number of ways in the future. The GPS coordinate tagging will allow the spatial correlation of device detections with the physical map of the facilities, thus allowing fast physical localization of rogue devices. Time-series behavioural analysis will be able to detect devices that are only seen during specific timeframes, including work hours, which is typical of insider threat situations. The ability to integrate with enterprise SIEM solutions such as Splunk and ELK will allow for a single pane of glass view of security monitoring for multiple BlueHunt sensors deployed concurrently. Last but not least, a light-weight Android mobile application will allow handheld scanning for physical security sweeps in facilities where the infrastructure for Bluetooth adapters is not fixed.

REFERENCES

[1] J. Becker, N. Asokan, and J. Bhatt, "Tracking Anonymized Bluetooth Devices," Proceedings on Privacy Enhancing Technologies, vol. 2019, no. 3, pp. 5-21, Jul. 2019.
[2] K. Fawaz, K. H. Kim, and K. G. Shin, "Protecting Privacy of BLE Device Users," in Proc. 25th USENIX Security Symposium, Austin, TX, USA, Aug. 2016, pp. 1205-1221.
[3] M. Ryan, "Bluetooth: With Low Energy Comes Low Security," in Proc. USENIX Workshop on Offensive Technologies (WOOT), Washington, D.C., USA, Aug. 2013.
[4] A. Das, N. Borisov, and M. Caesar, "Do You Hear What I Hear? Fingerprinting Smartphones Through Embedded Sensors," in Proc. ACM CCS, Scottsdale, AZ, USA, Nov. 2014, pp. 441-452.
[5] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation Forest," in Proc. IEEE International Conference on Data Mining (ICDM), Pisa, Italy, Dec. 2008, pp. 413-422.
[6] T. Chen, X. Gao, and G. Chen, "The Application of Classification Methods to Detect Network Intrusions," in Proc. IEEE International Conference on Big Data, Washington, D.C., USA, 2016, pp. 1291-1299.
[7] A. K. Sikder, G. Petracca, H. Aksu, T. Jaeger, and A. S. Uluagac, "A Survey on Sensor-based Threats to Internet-of-Things Devices and Applications," IEEE Communications Surveys and Tutorials, vol. 23, no. 2, pp. 1125-1159, 2021.
[8] S. Sciancalepore, G. Oligeri, and R. Di Pietro, "ROGUE: Rogue Access Point Detection Using Randomness," in Proc. ACM WiSec, Boston, MA, USA, Jul. 2017, pp. 168-178.
[9] M. Kershaw, "Kismet Wireless Network Detector," [Online]. Available: https://www.kismetwireless.net. [Accessed: Mar. 2026].
[10] Bluelog, "Bluetooth Scanner and Logger," [Online]. Available: https://github.com/MS3FGX/Bluelog. [Accessed: Mar. 2026].

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi crossref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details