Forensic Science International: Digital Investigation 48 (2024) 301686

Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

DFRWS EU 2024 - Selected Papers from the 11th Annual Digital Forensics Research Conference Europe

So fresh, so clean: Cloud forensic analysis of the Amazon iRobot Roomba vacuum

Abdur Rahman Onik [a,b,*], Ruba Alsmadi [a,b], Ibrahim Baggili [a,b], Andrew M. Webb [b]

[a] Baggili(i) Truth (BiT) Lab, Center of Computation & Technology, Baton Rouge, LA, USA
[b] Division of Computer Science & Engineering, Louisiana State University, Baton Rouge, LA, USA

ARTICLE INFO

Keywords:
IoT forensics
Autonomous vacuum
Smart home devices
Roomba
Cloud forensics

ABSTRACT

The advent of the smart home has been made possible by Internet of Things (IoT) devices that continually collect and transmit private user data. In this paper, we explore how data from these devices can be accessed and applied for forensic investigations. Our research focuses on the iRobot Roomba autonomous vacuum cleaner. Through detailed analysis of Roomba's cloud infrastructure, we discovered undocumented Application Program Interfaces (APIs). Leveraging these APIs, we developed PyRoomba – an open-source Python application that acquires a Roomba's complete mission history and navigational data. From this information, PyRoomba generates detailed mission logs and maps of navigated spaces, informing the user about mission duration, detected objects, degree of coverage, and encrypted image captures. We compared the outcomes of PyRoomba with Roomba's mobile application across six navigation runs in two environments of different sizes. We found that PyRoomba provides more detailed environmental information. A simulated crime scene case study demonstrated PyRoomba's ability to detect environmental changes, such as bodies and knives, which were identified as hazards or obstacles. PyRoomba offers a more forensically sound approach to cloud acquisition compared to Roomba's standard mobile application, minimizing the risk of inadvertently triggering the device during a crime scene investigation.

1. Introduction

The world's first autonomous robot vacuum, the Electrolux Trilobite, was developed in 1996. Since then, the technology surrounding these devices has rapidly and continually evolved[1]. Autonomous vacuums can now wirelessly connect to the internet, detect obstacles within centimeters, and—as we argue in this paper—help solve crimes.

As IoT devices become more ubiquitous and integrated into our daily lives, the data that they collect has raised several privacy concerns (Arabo et al., 2012; Buil-Gil et al., 2023; Chen et al., 2021). For example, the Amazon iRobot Roomba collects data to create a 3D home layout (Bugeja et al., 2021; Bettini et al., 2018) including furniture location and obstacles. An inadvertent leak of this data could reveal private aspects of the user's lifestyle and habits (Bettini et al., 2018) or enable behavioral profiling and object identification by law enforcement (McAmis and Kohno 2023).

As of 2022, iRobot's Roomba vacuum cleaners dominated the U.S. robotic vacuum market with a staggering 78 % market share. Their revenue in Q4 2022 showed a significant shift, with mid-tier and premium robots constituting 84 % of total sales, a rise from 81 % the previous year (Bedford 2023). Despite a minor dip, iRobot continued its global leadership, boasting a 46 % market share in 2020 (Department, 2023b) and a substantial 75 % share in North America in 2020 (Department 2023a).

The extent to which these devices facilitate home surveillance remains largely unexplored. Although Amazon asserts that they do not share this data with third parties, concerns persist about potential uses like targeted advertising (Guo 2022). Addressing these concerns, Bugeja et al. (2021) noted that Roomba has introduced a privacy policy on its mobile application and website, which details data collection and usage. Users may opt out of data sharing by disabling the "Clean Map" feature in the Roomba mobile application. Providing awareness of these privacy risks is vital for consumers. Companies must implement robust privacy and security measures to safeguard user data. This data, while raising privacy concerns, may prove indispensable for digital forensic investigations.

Investigators increasingly rely on IoT devices as digital evidence sources in cases. For example, Amazon Alexa was used to investigate

* Corresponding author. Baggili(i) Truth (BiT) Lab, Center of Computation & Technology, Baton Rouge, LA, USA.
E-mail addresses: aonik1@lsu.edu (A.R. Onik), ralsma1@lsu.edu (R. Alsmadi), baggili@gmail.com (I. Baggili), andrewwebb@lsu.edu (A.M. Webb).
[1] https://thamtus.com/blogs/blog/the-history-of-robot-vacuum-cleaner.

https://doi.org/10.1016/j.fsidi.2023.301686

Angie White's murder by her husband, Daniel White. Incriminating voice commands stored in the cloud helped establish a timeline and provide crucial evidence against Daniel White (BBC 2023). Similarly, Echo smart speakers have been used in murder investigations, with judges ordering Amazon to provide recordings and device data as potential evidence (Whittaker 2018).

We developed a cloud-based method for digital investigators to extract and analyze data from autonomous vacuums at crime scenes. Through studying Roomba's network communications, we designed a forensic tool to visualize the vacuum's navigation over time. This tool was validated using a simulated crime scene, proving its ability to detect environmental alterations related to crime.

Our contributions are as follows.

- We conducted the primary forensic analysis of the Roomba cloud infrastructure by analyzing its communication methods to access undocumented and hidden APIs for extracting artifacts. This allowed us to obtain operational data without directly interfacing with the physical device.
- We created PyRoomba, a Python-based open-source application that utilizes the APIs we discovered. This tool allows secure, authenticated access to Roomba's cloud infrastructure, enabling the retrieval of artifacts in a manner that maintains forensic integrity. We've made PyRoomba available to the public on GitHub .[2]
- We conducted a simulated crime scene case study to showcase potential forensic applications. The study presented how PyRoomba can reconstruct events, detect environmental changes, and provide insights unavailable via the Roomba mobile application.

This paper is organized as follows. Section 2 provides current research regarding autonomous robot vacuums. We discuss our procedure in Section 3, and we present the artifacts we retrieved as APIs in Section 4. Section 5 introduces the Python application based on our findings. Section 6 evaluates the results and includes a case study, while Section 7 presents the evaluation discussion. Section 8 presents the Limitation of our work. Finally, Section 9 discusses conclusions.

2. Related work

Previous research has addressed forensic challenges associated with the extraction and analysis of digital traces from IoT devices through the introduction of novel data extraction methods, the development of frameworks and tools to standardize practices, and the exploration of security and privacy issues.

2.1. Data extraction methods

Servida and Casey (2019) introduced a six-step forensic methodology, emphasizing the importance of data collection from IoT devices, cloud providers, and hardware analysis. The study successfully extracted traces from various sources, including system logs, user commands, device memory, networks, and smartphone applications, with recovered credentials enabling cloud data access.

Kim et al. (2020) addressed challenges in extracting meaningful data from diverse smart home devices. They focused on devices like Google Nest Hub, Samsung SmartThings, and Kasa Cam for forensic purposes, encompassing device data, movements, voice commands, and call history. Through correlational analysis, they derived a comprehensive framework with enhanced data accuracy for smart home data forensics.

Chung et al. (2017) presented a novel approach to digital forensics within the Alexa ecosystem. Their approach combined cloud-native and client-side forensics to support investigations. They introduced the Cloud-based IoT Forensic Toolkit (CIFT) tool as a proof-of-concept for identifying, acquiring, and analyzing artifacts from cloud and local devices in the Alexa ecosystem.

Zhou et al. (2022) extracted and analyzed Roomba's operational logs, installation details of the control system, and application usage records, all sourced from the memory of a smartphone. While this study offered valuable insights into smartphone-based data acquisition, our research differed in its approach and focus. Our work focused on the cloud forensic analysis of Roomba vacuums, leveraging undocumented APIs for a more comprehensive and forensically sound method of data extraction.

2.2. Frameworks and tools

Meffert et al. (2017) proposed Forensic State Acquisition from Internet of Things framework (FSAIoT) to address issues associated with standardization, storage limitations, and diverse communication protocols. Wu et al. (2019) presented findings from an IoT forensics survey, emphasizing challenges in acquiring and analyzing IoT data and devices while shaping a clear understanding of the field. Dorai et al. (2018) introduced the Forensic Evidence Acquisition and Analysis System (FEAAS), an open-source tool designed to assist digital forensics in smart home IoT scenarios. Additionally, Baggili et al. (2015) detailed preliminary forensic analysis of popular smartwatches, and they highlighted their sluggish security in other work (Ricci et al., 2017). Pace et al. (2023) recently developed an open-source tool called Tile Artifact Parser (TAP) to parse forensic data from Bluetooth trackers. The research highlighted the complexity of IoT device forensics because of issues like standardization, limited historical data, and constant connectivity challenges.

There has been a development of open-source projects implementing Roomba's cloud APIs, contributing to the broader understanding of IoT device control and interaction (Koalazak 2021, Clown0503, 2014). This past work examined and implemented Roomba's operational framework, allowing users to control Roomba devices via an alternative to the official application. While this past work was interesting, it did not acquire data from Roomba's cloud infrastructure in a forensically sound manner. This past work changes the integrity of the evidence by controlling the device and its operations.

2.3. Security and privacy issues

Panwar et al. (2019) and Bugeja et al. (2021) addressed smart home security and privacy concerns, highlighting potential risks tied to features like Roomba's mapping technology. They emphasized the importance of frameworks such as Privacy Risk Analysis of Smart Homes (PRASH) for systematically identifying privacy threats and aiding digital forensics. Similarly, Hartzog (2014) discussed privacy and data security concerns, particularly regarding Roomba's mapping technology. While prior studies explored privacy and security concerns for data collected by Roombas and other IoT devices, this work uniquely studies the forensic value of Roomba's data collection.

3. Methodology

Our methodology consisted of four phases: preparation, identification of cloud APIs, tool creation, and validation.

3.1. Preparation

The preparation phase involved obtaining the necessary hardware components and installing the required software on the devices (see Table 1). We first acquired a Roomba vacuum cleaner (Roomba J Series) and an Android smartphone (Samsung Galaxy s10+). The official Roomba mobile application was downloaded from the Google Play Store and logged in using user credentials. The acquired Roomba was previously linked to another account, requiring that we reset the device in

[2] https://github.com/BiTLab-BaggiliTruthLab/PyRoomba.

Table 1
List of apparatus.

Hardware/ Software	Use	Company	Software/Model Version
Galaxy s10+	Roomba Companion Device	Samsung	Android 11.0.0
Windows PC	Roomba Companion Device	Microsoft Corporation	Windows 10.0.22621
Roomba J Series - j715020	Tested Roomba Device	iRobot Corporation	sapphire+23.12.4 + 2023-06-07-cca733b60c4+Firmware-Production+150
APK-MITM Tool	Modification of Android APK files to enable Hypertext Transfer Protocol Secure (HTTPS) inspection	Open Source	v1.2.1
MITMProxy Tool	An interactive HTTPS proxy tool for intercepting and analyzing network traffic	Open Source	mitmproxy 10.1.0

accordance with the official Roomba guidelines (iRobot Corporation, 2023). Concurrently, *apk-mitm* and *mitmproxy* tools were installed on our Windows PC. Additionally, the *mitmproxy* certificate was successfully integrated into our Android phone and Windows PC.

We used *apk-mitm* to inspect the Roomba mobile application's HTTPS traffic (Shroudedcode 2019) (see Fig. 1a). To track the HTTPS traffic of the mobile application, we connected our test phone to our Windows PC using the mobile hotspot feature. Both the test phone and the PC must be on the same network. Next, we proceeded to install both *mitmproxy* and the modified Roomba application onto our test phone. Once installed, we authenticated the app using our credentials and established a connection with the Roomba (see Fig. 1b).

3.2. Identification of undocumented cloud APIs

In the second phase, we used the *mitmweb*, a web-based interface of *mitmproxy* tool to view and monitor the HTTPS traffic between the iRobot mobile application and Amazon's web server (see Fig. 1c). This was essential as the traffic was Transport Layer Security (TLS) encrypted, necessitating the use of a Man-in-the-Middle (MitM) approach to decrypt and analyze the data. We analyzed the network traffic to identify all the Application programming interface (API) requests. Our criteria for identification focused on the name of the APIs and the JSON responses this APIs returned. If the JSON contained information such as timestamps, user metadata, coordinates, or any map-related details, we considered that APIs relevant. All detected API requests, headers, query parameters, and responses were documented. We saved these interactions as Flows—captured sequences of network requests and responses that are viewable and inspectable via the *mitmweb* interface. Next, we used these findings (APIs) to develop a forensic tool.

3.3. Tool creation: PyRoomba

We developed *PyRoomba*, a forensic web application built with Python 3.11, that extracts Roomba's navigational mission data via undocumented Roomba APIs without direct device interaction, ensuring forensic integrity (refer to Fig. 5bd). This application not only generates maps from the navigation data, as detailed in Section 5, but also offers timestamps, detected objects, and coverage details. While Roomba captures and labels images during navigation, Amazon encrypts them for security. PyRoomba grants users access to these encrypted images via direct download links, along with crucial cryptographic information like the encryption and hash keys, as well as the initialization vector (IV).

3.4. Validation

To validate PyRoomba, we set up two controlled environments: one larger room and one smaller room, both filled with common household objects to replicate real-world scenarios. In the larger room, we deployed the Roomba four times, while in the smaller room, we ran it twice. Based on the API data, we used our proposed algorithm to construct a map of the room. We compared the map produced by PyRoomba against the actual scene. Additionally, we evaluated the map created by PyRoomba in comparison to the map generated by the native Roomba mobile application (which offers fewer details than our forensic application).

4. Undocumented Roomba APIs

We performed intensive traffic analysis using *mitmproxy* to gain insight into Roomba's cloud services. Our analysis revealed transfers of JSON-formatted artifacts between the Roomba device, mobile application, and cloud services. Although Roomba does not publish official APIs documentation, our analysis revealed undocumented APIs used by their mobile application. By capturing and interpreting JSON responses from the cloud, we discovered six APIs related to device configuration and usage data. Table 2 outlines the features of the undocumented Roomba APIs and their potential significance for forensic investigations. Overall, this analysis provided valuable insight into the cloud services and data formats used by Roomba devices. With an in-depth analysis of undocumented APIs, our application can access artifacts without going through the official mobile application, limiting the risk of accidental

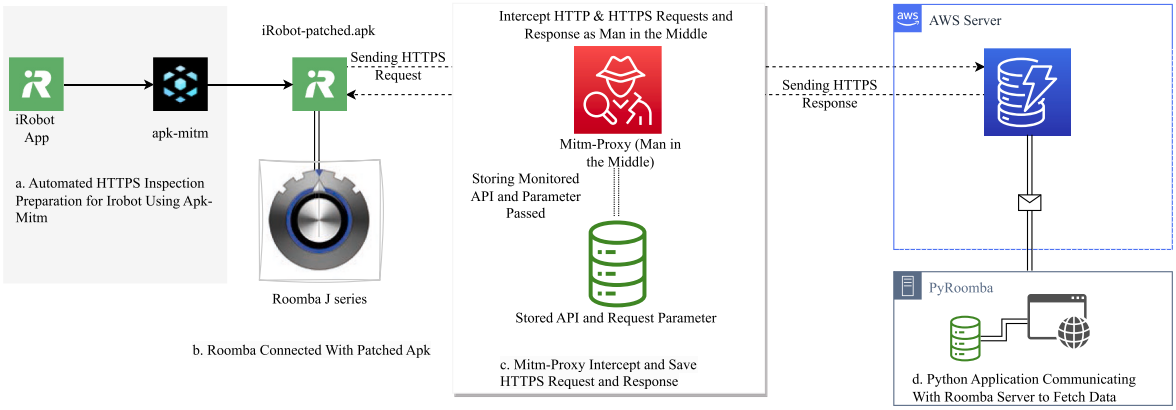


Fig. 1. Network traffic analysis of amazon roomba.

Table 2
Roomba endpoint APIs and forensic relevance.

API Category	Undocumented Roomba APIs	Description	Forensic Relevancy
Platform Configuration	https://disc-prod.iot.irobotapi.com/v1/discover/endpoints?country_code = US	Configuration details about the iRobot IoT platform	Provides insight into the infrastructure and potential data storage locations.
Account Provider	https://accounts.us1.gigya.com/accounts.getAccountInfo	User account and profile information	Offers a timeline of account activity, login methods, and verification status, which can be used to track user interactions and potential unauthorized access.
Mission History	https://auth3.prod.iot.irobotapi.com/v1/robotid/missionhistory?filterType = app_id	Comprehensive dataset about Roomba's activities	Offers insights into Roomba's operations, including Wi-Fi connections, charging times, and cleaning durations. This can help establish timelines and identify anomalies.
Roomba Map Detail Coordinates	https://auth3.prod.iot.irobotapi.com/v1/robotid/pmaps/pmap_id/versions/pmapv_id/	Information regarding Roomba's navigational coordinates to reconstruct the map	Offers a digital footprint, enabling the reconstruction of events, verification of objects, detection of environmental changes.
Obstacle Detected	https://auth1.prod.iot.irobotapi.com/v1/robots/340EE4928078487E853BF9F3180A3898/imageupload/metadata	Details about the type of obstacle	Forensically relevant in reconstructing events, understanding environmental dynamics, and establishing potential evidence in investigations.
Image Retrieval	https://auth1.prod.iot.irobotapi.com/v1/robots/340EE4928078487E853BF9F3180A3898/imageupload/imagesurl	Link to download Encrypted Images captured by Roomba	Images to see the Obstacles presented in an Environment.

activation by investigators.

It's crucial to note that the official Roomba mobile application does not offer comprehensive data. Specifically, our application provides obstacle types identified on the map and discloses the names of obstacles imaged by Roomba. Additionally, using the official app poses a risk in forensic investigations, as unintentional activation of the Roomba is possible. Our application mitigates this risk.

4.1. Roomba's Platform Configuration API

We examined a Platform Configuration API that yields configuration specifics of Roomba's IoT platform. For the "U.S." region, the API identified three deployments under the `deployments` key, likely meaning staging, testing, and production settings. The common configuration parameters include.

- `awsRegion`: Region where the Roomba server is located.
- `httpBase`: Base URL for unauthenticated Hypertext Transfer Protocol (HTTP).
- `httpBaseAuth`: Base URL for authenticated HTTP.
- `mqtt`: This highlights Roomba uses Message Queuing Telemetry Transport (MQTT) protocol. The messaging system is designed for devices with limited memory and bandwidth.

4.2. Roomba's Account Provider API

Our analysis discovered an Account Provider API to access account management information. Roomba uses *gigya*, a customer identity management platform. The JSON data provided by this API includes first and last name, email address, country, account registration timestamp, last login timestamp, user preferences, and account status. In a forensic investigation, this data helps establish user behavior, verify identity, and correlate with other pieces of evidence. The returned JSON object contains the following properties.

- `timestamps`: Contains registered timestamp, signature timestamp, created timestamp, last login timestamp, and last updated timestamp. This provides a details timeline of an account activity.
- `loginProvider`: Indicates whether the user is logged in via a social network or a site.
- `newUser`: Either `True` or `False`. A value of `True` means the account was recently created. A value of `False` means this account has existed for some time.
- `isVerified`: Information about Accounts email Verification.
- `verifiedTimestamp`: Provides the timestamp for when the account was verified. This will help to establish the timeline of when the account was created.

4.3. Roomba's Mission History API

In the Roomba Mission History API, detailed information on cleaning missions is provided. This detailed documentation includes the following parameters.

- `chrgM`: Charge duration (in minutes) while navigating.
- `chrgs`: Total charging time (in seconds).
- `cmd`: Command types (e.g., train, start, resume, pause).
- `initiator`: Roomba's start method (remoteApp, localApp, or manual).
- `dirt`: Refers to the coordinates of dirt detected by the Roomba during its operation.
- `dockedAtStart`: Whether Roomba began from its dock.
- `durationM`: Navigation duration (in minutes).
- `missionId`: Unique identifier for the mission.
- `nMssn`: Total mission count.
- `pmap`: IDs (pmap_id and pmapv_id) for navigation data, used to reconstruct Roomba's path as described in Section 4.4. This information is employed to reconstruct the robot's navigation map, offering a comprehensive view of its movement and coverage during its navigation.
- `runM`: Total runtime (in minutes).
- `softwareVer`: Roomba's software version.
- `sqft`: Cleaned area (in square feet).

4.4. Roomba Map Detail Coordinates API

This API, utilizing `pmap_id` and `pmapv_id`, fetches detailed Roomba navigation specifics. The key "maps" in the response contain.

- `map_header`: Metadata about the map including name, creation time, and mission count.
- `border`: Outer boundary coordinates detected by Roomba.
- `regions`: Segmented area coordinates identified by Roomba.
- `layers`: Coordinates for map information like coverage, and dirt. Coverage points offer coordinates that allow us to map the area the Roomba has cleaned in a room.
- `points2d`: Contained two-dimensional points representing specific locations or features on the map

- `pose2d`: Coordinates of Roomba's docker position, start and end position.
- `objects`: Types of Objects and their coordinates Roomba detected.
- `hazards`: Objects Roomba detected but couldn't identify the type were categorized as hazards and provided coordinates.
- `escape_events`: Coordinates of brush caught into something and it stopped.
- `door_coords`: Coordinates of doors.

4.5. Roomba Obstacle Detection API

The Roomba Obstacle Detection API provides information that can help in investigations. In JSON response, some files have encrypted images. All images are labeled with the name of the object in them.

4.6. Image Retrieval API

This API offers links to download images taken by Roomba during its navigation. Besides images, the API includes files with encryption keys, hash values, and IV specifics. Additionally, a file lists all the JSON details obtained from the Obstacle Detection API. It's important to note that these downloadable links are available for a limited duration. During our analysis, we observed that the link remained active for approximately 36 h. Interestingly, if the images captured by the Roomba are reviewed, this link disappears.

5. PyRoomba: a roomba forensics application

PyRoomba, developed in Python 3.11, serves as a cloud forensic tool that surpasses the standard Roomba mobile applications in data extraction capabilities. It ensures secure data retrieval from the cloud, a contrast to potential risks posed by mobile applications that may inadvertently activate the Roomba or face data loss due to uninstallation or phone resets. This approach safeguards forensic integrity, offering reliable and secure cloud-based data access. Noteworthy features include the extraction of detailed `Mission History` (depicted in Fig. 2a), offering insights into Roomba's missions, and `Mission Details` (Fig. 2b), providing specific mission details. The `Navigational Map` (refer to Fig. 2c) presents a comprehensive 2D map with detected objects, timestamps, and environmental insights. A distinctive attribute of PyRoomba lies in its ability to download data of identified objects, enhancing the depth of forensic analysis. The tool incorporates real-time checks during login and data fetching, ensuring adaptability to API changes and maintaining consistent reliability. Through historical analysis of Roomba's API, PyRoomba demonstrates enduring consistency, particularly in login endpoints, instilling confidence in its long-term utility.

The key features of PyRoomba are.

- **Amazon Web Services (AWS) Authentication:** Uses AWS authentication to ensure that the data downloaded from the cloud is authentic and reliable. With the inherent data integrity and security features of AWS, the data retrieved from this application is safeguarded against tampering or manipulation, thereby enhancing its reliability.
- **Full Mission History and Details:** Provides a timeline record of where and when the Roomba was active. For forensic investigations, it can help determine the presence or movement of individuals or objects in a particular area at a specific time.
- **2D Map Generation From Roomba Mission Details:** Visualizes a 2D map of where the Roomba has cleaned or traversed. If there are specific areas the Roomba didn't access or unusual patterns in its movement, this might indicate obstacles, changes in the environment, or other interruptions that could be related to an incident.

5.1. Authentication

Our application requires AWS authentication to collect cloud-based data associated with a Roomba. We implemented an authentication process based on the guidelines provided in the official AWS documentation.[3]

5.2. Mission history and details

The Mission History documents all navigations executed under a specific account linked with the Roomba. It's important to note that data from any previous accounts cannot be retrieved if a factory reset has occurred. However, the total number of missions run by the Roomba since its inception can be accessed, as the mission count is always recorded from its start. Additionally, the Roomba mobile application provides 30 navigation histories, but PyRoomba allows retrieval of all navigation histories. Using PyRoomba, users can access a range of mission details such as the mission number, whether the Roomba charged during the mission, dirt detection, starting point (whether it began from the dock or not), task completion status, mission duration, start and end times, bin cleaning status, the initiating entity, pause duration, and the covered area in square feet. For a more visual representation of these details, users can select individual mission numbers under the "Mission History" section of the application. Fig. 2 presents a visual representation of the"Mission Details" interface in the application.

5.3. PlotMap: navigational map generation

PyRoomba's PlotMap feature provides a 2D map visualization of Roomba's navigated areas and detected objects. This visualization is derived from data obtained through the Mapping Metadata API. Key elements of the data include room boundaries, the positions and orientations of objects, and specific locations such as docking stations, entrances, exits, and hazards. The PlotMap process involves plotting these data points onto a 2D map, effectively representing Roomba's movement, cleaning coverage, and interactions with obstacles. For those interested in the detailed implementation of the PlotMap feature, the complete code is available in our code repository for further exploration.

The visualization starts by outlining the room's borders and then marking key locations like the starting point, docking station, and endpoints of the Roomba's journey. Doors are represented by magenta lines, highlighting entrances and exits. Hazards and obstacles, such as furniture, are also plotted to show areas Roomba navigates around. Our algorithm focuses on detailing the areas cleaned by Roomba, indicated by the density of coverage points within the mapped area. Additionally, it highlights locations where Roomba encountered difficulties or was stuck, providing a comprehensive view of its operational efficiency and navigational challenges.

6. Evaluation

This section presents our experimental analysis comparing PyRoomba's forensic capabilities against the standard Roomba mobile application. We focused on four core aspects.

- **Accuracy of Navigational Data:** How does PyRoomba compare to the Roomba mobile application in accurately and comprehensively capturing navigational data?
- **Performance Consistency:** Evaluating PyRoomba's effectiveness in different environments, specifically in rooms of varying sizes.

[3] https://docs.aws.amazon.com/AmazonS3/latest/API/sig-v4-authenticating-requests.html.

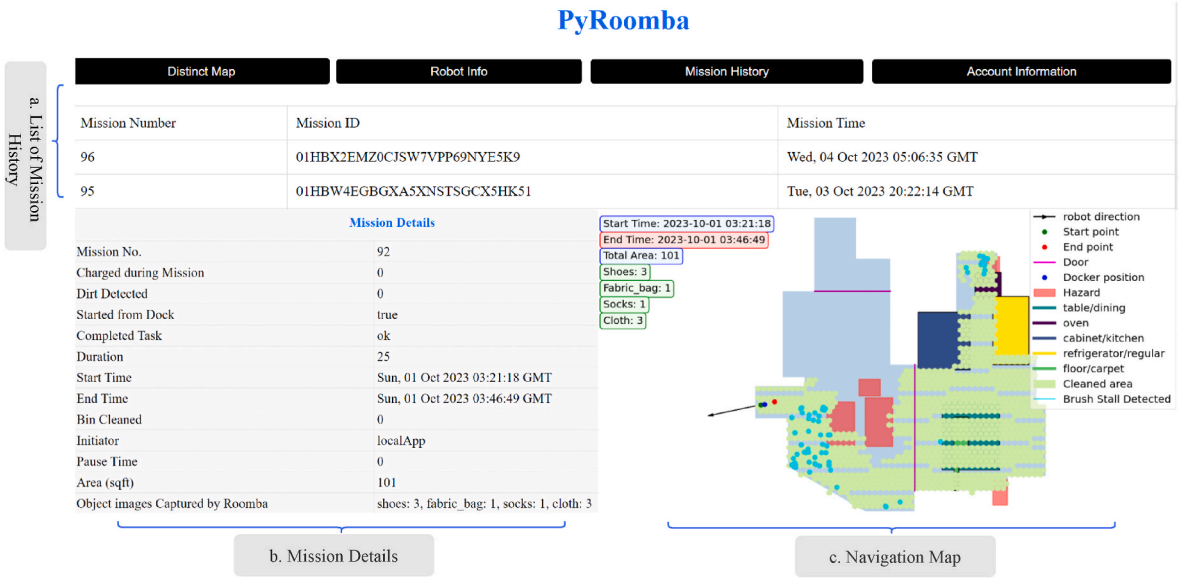


Fig. 2. Visualization of the main features of the PyRoomba application.

- **Reliability After Resets:** Testing PyRoomba's ability to retrieve and analyze data post-factory resets, simulating real-world scenarios.
- **Crime Scene Case Study:** Applying PyRoomba in a simulated crime scene to detect environmental alterations and identify potentially significant unusual objects for forensic analysis.

Our controlled experiments aim to understand the comparative effectiveness of our tool in forensic contexts, providing insights into its potential and limitations.

6.1. Procedure

Our evaluation consisted of performing multiple iterations of the cleaning routine with the Roomba in two separate environments with different sizes and object density: a small bedroom and a larger studio apartment. Following each iteration, we constructed maps using both

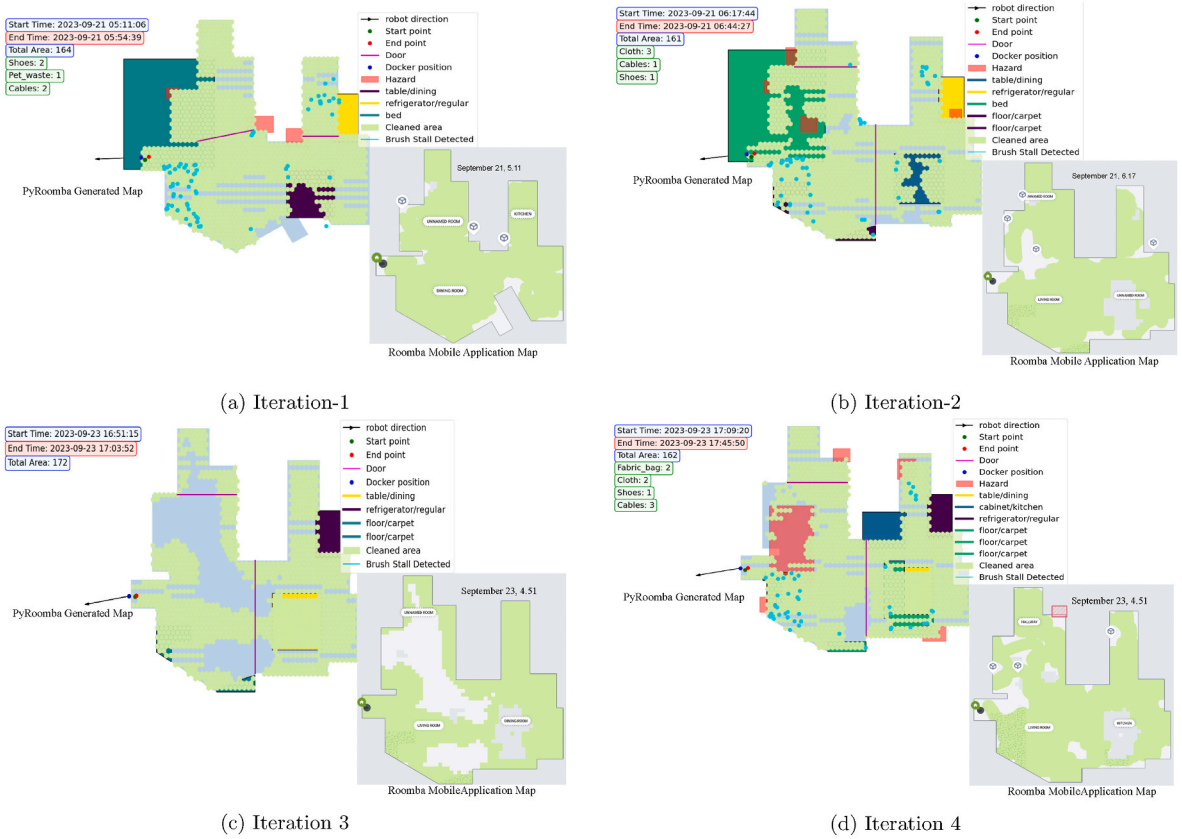


Fig. 3. Maps generated during iterations 1–4 by pyroomba and the roomba mobile application for the studio apartment environment. For each iteration, Pyroomba's map is shown on the left, and the roomba App's map is shown on the right.

PyRoomba and the Roomba mobile application (see Figs. 3 and 4). Additionally, we executed factory resets on the Roomba. We did this to emulate potential user behavior in real-world scenarios where devices are reset for various reasons, such as resale or troubleshooting. After each reset, a new ID was utilized to log into the Roomba, ensuring that any retained data could not be attributed to continuous sessions or cached user profiles.

6.1.1. Studio apartment exploration

The studio apartment consisted of multiple objects, including a bed, kitchen area, refrigerator, shoe rack, kitchen cabinet, two pairs of shoes, an oven, numerous cables, and fabric bags. We conducted four runs in this environment over two days (two runs per day).

6.1.2. Small bedroom exploration

The small bedroom was furnished with a bed, reading table, ottoman, a single pair of shoes, and cables. Notably, this room also contained a distinct cabinet with a door. We conducted two runs in this environment.

6.1.3. Objectives

Our primary objective was to measure how accurately and consistently the Roomba can recognize objects and detect hazards or obstacles. Further, we sought to determine if subsequent runs would provide new insights, detect previously unnoticed hazards, or fail to recognize obstacles identified during previous runs. Finally, running the Roomba in the small bedroom allowed us to understand its behavior in a more restricted environment.

6.2. Results

The results from the Roomba iterations provide insightful observations into its performance, object recognition, and cleaning patterns in both environments.

The maps for the studio apartment exploration, iterations 1 to 4, are presented in Fig. 3. We observe differences regarding the Roomba's coverage and object detection capabilities. In the first iteration, as depicted in Fig. 3a, the Roomba operated for 43 min, covering an expanse of 164 sqft. It identified 5 objects: 2 shoes, 1 pet waste, and 2 cables. In contrast, iteration 2 (Fig. 3b) was shorter at 26 min and covered 161 sqft. It detected 5 objects: three pieces of cloth, a cable, and a shoe. In iteration 3 (Fig. 3c) the Roomba ran for 36 min, covering 162 sqft. It detected 8 objects: 2 fabric bags, 2 cloths, a shoe, and 3 cables. The final run, iteration 4, is illustrated in Fig. 3d. The Roomba operated for 33 min and covered 148 sqft. It detected 11 objects: 5 cables, 3 fabric bags, 2 shoes, and a cloth.

The map outcomes for the small bedroom exploration, iterations 5 and 6, are depicted in Fig. 4a and b. In iteration 5, the Roomba ran for 21 min and covered 112 sqft. It detected no objects. We suspect this is due to navigational obstructions. In iteration 6, the Roomba ran for 15 min and covered 92 sqft. It detected 4 objects: 2 shoes, a cable, and pet waste.

6.3. Case study: crime scene simulation

We prepared a room with standard items as shown in Fig. 5a. The Roomba was activated to navigate and document the area. This initial scan began at 20:22:14 GMT on Tue, 03 Oct 2023, and ended at 20:52:00 GMT, with a duration of 29 min, and covered an area of 156 sqft. During its run, the Roomba identified one cloth, five cables, two shoes, two fabric bags, one toy, and one pet bowl. The map generated by PyRoomba is shown in Fig. 5b. After running this initial baseline scenario, we simulated a crime, positioning a dead body on the floor and accompanying it with knives. We then reactivated the Roomba to map the modified scene.

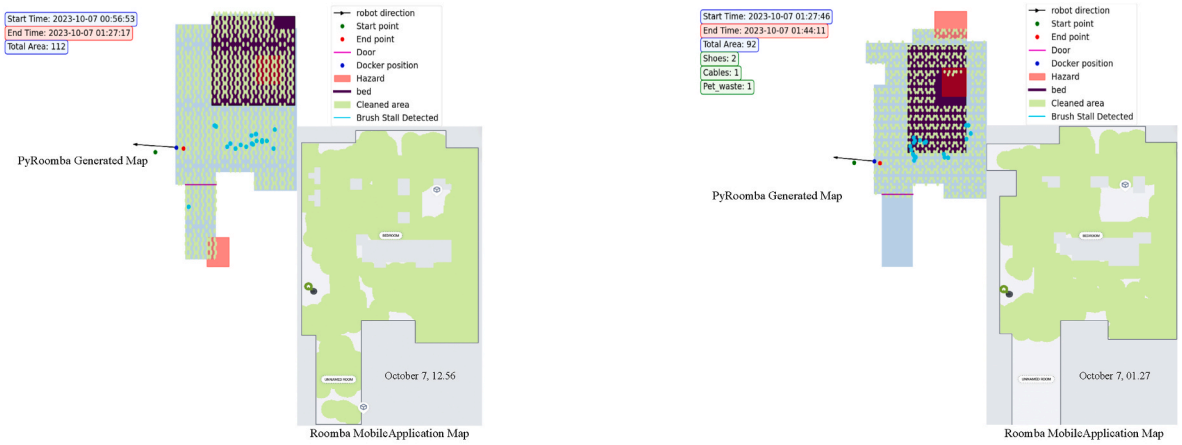
During the second run, the Roomba documented an area of 126 sqft. This mapping was initiated at 05:06:36 GMT on Wed, 04 Oct 2023, and completed at 05:34:08 GMT, lasting 27 min. The updated room's setup can be seen in Fig. 5c. The crime scene PyRoomba map, presented in Fig. 5d, demonstrated changes in identified items: three clothes, three cables, four shoes, two fabric bags, and one toy. Most significantly, the Roomba flagged two new "hazards." Upon manual inspection, these hazards were linked to the simulated dead body and the knives.

7. Discussion

From the Roomba's navigation of the two environments, we observe different object detection behaviors. In the studio apartment exploration, the Roomba detected a diverse range of objects, with dining, carpet, and refrigerator being detected more consistently across iterations.

Additionally, the data suggests that the Roomba may be improving its detection algorithms with each iteration as it discovered the Kitchen cabinet in the fourth iteration. The same decision can be concluded for iterations 5 and 6. Initially, it overlooked shoes and cables within the room. However, in subsequent iterations, it identified these obstacles. We find that the Roomba performed better in a small bedroom compared to the larger studio apartment. The Roomba obtains better coverage in a smaller space and spots objects more accurately.

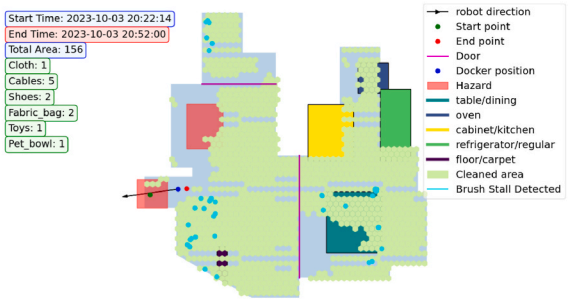
When we compared the maps generated by our system with those from Roomba's mobile application, it became clear that our maps



(a) Iteration 5 (b) Iteration 6

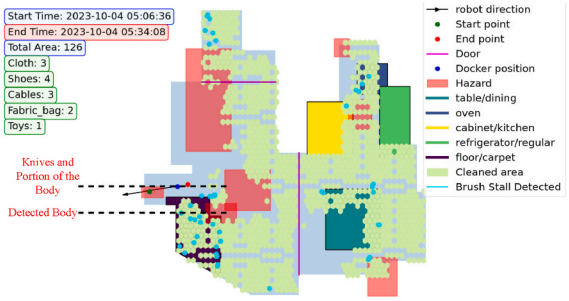
Fig. 4. Maps generated by pyroomba and roomba mobile application for the small bedroom environment, iterations 5–6.

(a) A Room Setup Before Crime



(b) Room Map Generated by PyRoomba

(c) A Crime Scene With Body and Knife



(d) Crime Scene Map Generated by PyRoomba

Fig. 5. Crime scene creation and comparison of PyRoomba maps between two scenarios.

contained more detailed environmental information. This added detail not only highlights the efficiency of our constructed maps but also provides the advantage of not being entirely reliant on the Roomba mobile application. From a forensic standpoint, this data is valuable. PyRoomba maps can be pivotal for scene reconstruction. The timestamps can assist in creating event timelines, and differences in object recognition may suggest the removal of items.

The Roomba's ability to identify changes in the environment, such as a decrease in navigable area from 156 sqft to 126 sqft, indicates its potential in forensic contexts. The decrease in the mapped area indicated the dead body, potentially limiting the Roomba's navigation space. While the Roomba successfully identified new items like the body and knives, it classified them as "hazards" without detailing their specific nature. This highlights that while the Roomba can note changes in an environment, human inspection is important in interpreting these modifications.

8. Limitations

While our approach is robust, it exhibits several limitations in retrieving all forensic artifacts from Roomba devices. Firstly, it is centered on cloud acquisition and does not encompass device-level acquisition. Secondly, it necessitates user credentials. Thirdly, our findings are specific to the Roomba J series model; we have not conducted tests on all Roomba models. To decrypt the encrypted images, necessary components included the encryption key (the password for scrambling images), the hash (a unique code to verify image integrity), the IV for ensuring unique encryptions, and the master key (a top-level key for securing the encryption key) (Bozorg-Haddad et al., 2017). However, due to our inability to access the master key, we couldn't decrypt the images. Nonetheless, the image labels helped identify their content without needing to decrypt them.

9. Conclusions

Despite extensive work on IoT forensics, Roomba's cloud forensics have remained unexplored in peer-reviewed literature. Using HTTPS interception and web API analysis, we successfully extracted rich forensic data from Roomba. In our research, we recognized the ethical complexities in using undocumented APIs. These APIs, although not publicly documented, are part of the application's accessible interface. We ensured our methods were ethically sound and legally compliant, particularly in handling private user data. This highlights the importance of ethical responsibility in cloud forensics. Our creation, PyRoomba, is a forensically robust Python application for this purpose. Our approach and tool establish a framework for IoT cloud forensics, particularly in cases of limited device data access.

Our future work aims to develop a generalized model that categorizes data across various automated vacuum robots, potentially broadening our framework to include multiple brands and models, thus enhancing its applicability and depth.

References

Arabo, A., Brown, I., El-Moussa, F., 2012. Privacy in the age of mobility and smart devices in smart homes. In: '2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing', pp. 819–826.

Baggili, I., Oduro, J., Anthony, K., Breitinger, F., McGee, G., 2015. Watch what you wear: preliminary forensic analysis of smart watches. In: 2015 10th International Conference on Availability. Reliability, and Security', IEEE, pp. 303–311.

BBC, 2023. Swansea: Man Charged with Murder of Woman Named as Angie White. URL: https://www.bbc.com/news/uk-wales-63367246.

Bedford, 2023. 'irobot Reports Fourth-Quarter and Full-Year 2022 Financial Results'. **URL:**. https://investor.irobot.com/news-releases/news-release-details/irobot-reports-fourth-quarter-and-full-year-2022-financial.

Bettini, C., Shmueli, E., Lanzi, A., Riboni, D., Lepri, B., 2018. The Privacy Implications of Cyber Security Systems: A Technological Survey.

Bozorg-Haddad, O., Solgi, M., Loáiciga, H.A., 2017. Meta-heuristic and Evolutionary Algorithms for Engineering Optimization. John Wiley & Sons.

Bugeja, J., Jacobsson, A., Davidsson, P., 2021. Prash: a framework for privacy risk analysis of smart homes. Sensors 21 (19), 6399.

Buil-Gil, D., Kemp, S., Kuenzel, S., Coventry, L., Zakhary, S., Tilley, D., Nicholson, J., 2023. The digital harms of smart home devices: a systematic literature review. Comput. Hum. Behav. 145, 107770.

Chen, B., Liu, Y., Zhang, S., Chen, J., Han, Z., 2021. A survey on smart home privacy data protection technology. In: '2021 IEEE Sixth International Conference on Data Science in Cyberspace (DSC)', pp. 583–590.

Chung, H., Park, J., Lee, S., 2017. Digital forensic approaches for amazon alexa ecosystem. Digit. Invest. 22, S15–S25.
Clown 0503, 2014. Pyroomba. https://github.com/Clown0503/PyRoomba.
iRobot Corporation, 2023. 'Find Answers | iRobot'. **URL:**. https://homesupport.irobot.com/s/article/9047.
Department, S.R., 2023a. Robotic Vacuum Cleaner Market Revenue Share in the North American Market from 2017 to 2020, by Brand. **URL**: https://www.statista.com/statistics/934290/north-america-robotic-vacuum-cleaner-revenue-share/.
Department, S.R., 2023b. Robotic Vacuum Cleaner Market Share Worldwide from 2014 to 2020 by brand'. **URL**: https://www.statista.com/statistics/934089/worldwide-robotic-vacuum-cleaner-market-share/.
Dorai, G., Houshmand, S., Baggili, I., 2018. I know what you did last summer: your smart home internet of things and your iphone forensically ratting you out. In: Proceedings of the 13th International Conference on Availability. Reliability, and Security', pp. 1–10.
Guo, E., 2022. 'A Roomba Recorded a Woman on the Toilet. How Did Screenshots End up on Facebook?'. **URL:**. https://www.technologyreview.com/2022/12/19/1065306/roomba-irobot-robot-vacuums-artificial-intelligence-training-data-privacy/.
Hartzog, W., 2014. Unfair and deceptive robots. Md. Law Rev. 74, 785.
Kim, S., Park, M., Lee, S., Kim, J., 2020. Smart home forensics—data analysis of iot devices. Electronics 9 (8).
Koalazak, 2021. 'dorita 980'. https://github.com/koalazak/dorita980.
McAmis, R., Kohno, T., 2023. The writing on the wall and 3d digital twins: Personal information in (not so) private real estate. In: 32nd USENIX Security Symposium (USENIX Security 23). USENIX Association, Anaheim, CA, pp. 2169–2186.
Meffert, C., Clark, D., Baggili, I., Breitinger, F., 2017. Forensic state acquisition from internet of things (fsaiot) a general framework and practical approach for iot forensics through iot device state acquisition. In: Proceedings of the 12th International Conference on Availability. Reliability, and Security', pp. 1–11.
Pace, L.R., Salmon, L.A., Bowen, C.J., Baggili, I., Richard III, G.G., 2023. Every step you take, i'll be tracking you: forensic analysis of the tile tracker application. Forensic Sci. Int.: Digit. Invest. 45, 301559.
Panwar, N., Sharma, S., Mehrotra, S., Krzywiecki, Ł., Venkatasubramanian, N., 2019. 'Smart Home Survey on Security and Privacy'. *arXiv preprint arXiv:1904.05476*.
Ricci, J., Baggili, I., Breitinger, F., 2017. Watch what you wear: smartwatches and sluggish security. In: Managing Security Issues and the Hidden Dangers of Wearable Technologies. IGI Global, pp. 47–73.
Servida, F., Casey, E., 2019. Iot forensic challenges and opportunities for digital traces. Digit. Invest. 28, S22–S29.
Shroudedcode, 2019. 'Github - Shroudedcode/apk-Mitm: A Cli Application that Automatically Prepares Android Apk Files for Httpsinspection'. URL: https://github.com/shroudedcode/apk-mitm.
Whittaker, Z., 2018. Judge Orders Amazon to Turn over Echo Recordings in Double Murder Case. URL: https://techcrunch.com/2018/11/14/amazon-echo-recordings-judge-murder-case/.
Wu, T., Breitinger, F., Baggili, I., 2019. Iot ignorance is digital forensics research bliss: a survey to understand iot forensics definitions, challenges and future research directions. In: Proceedings of the 14th International Conference on Availability. reliability, and security', pp. 1–15.
Zhou, H., Deng, L., Xu, W., Yu, W., Dehlinger, J., Chakraborty, S., 2022. Towards internet of things (iot) forensics analysis on intelligent robot vacuum systems. In: '2022 IEEE/ACIS 20th International Conference on Software Engineering Research, Management and Applications (SERA)', pp. 91–98.