Forensic Investigation of Humanoid Social Robot: A Case Study on Zenbo Robot

Farkhund Iqbal*
College of Technology Innovation, Zayed University, United Arab Emirates
Farkhund.Iqbal@zu.ac.ae

Abdullah Kazim
College of Technology Innovation, Zayed University, United Arab Emirates.
M80007502@zu.ac.ae

Aine Mac Dermott
School of Computer Science and Mathematics, Liverpool John Moores University, UK.
A.M.MacDermott@ljmu.ac.uk

Richard Ikuesan
College of Technology Innovation, Zayed University, United Arab Emirates
Richard.Ikuesan@zu.ac.ae

Musaab Hassan
Emirates College for Advanced Education, United Arab Emirates
Musaab.Hasan@ecae.ac.ae

Andrew Marrington
College of Technology Innovation, Zayed University
Andrew.Marrington@zu.ac.ae

ABSTRACT

The Internet of Things (IoT) plays a significant role in our daily lives as interconnection and automation positively impact our societal needs. In contrast to traditional devices, IoT devices require connectivity and data sharing to operate effectively. This interaction necessitates that data resides on multiple platforms and often across different locations, posing challenges from a digital forensic investigator's perspective. Recovering a full trail of data requires piecing together elements from various devices and locations. IoT-based forensic investigations include an increasing quantity of objects of forensic interest, the uncertainty of device relevance in terms of digital artifacts or potential data, blurry network boundaries, and edgeless networks, each of which poses new challenges for the identification of significant forensic artifacts. One example of the positive societal impact of IoT devices is that of Humanoid robots, with applications in public spaces such as assisted living, medical facilities, and airports. These robots use IoT to provide varying functionality but rely heavily on supervised learning to customize their utilization of the IoT to various environments. A humanoid robot can be a rich source of sensitive data about individuals and environments, and this data may assist in digital investigations, delivering additional information during a crime investigation. In this paper, we present our case study on the Zenbo Humanoid Robot, exploring how Zenbo could be a witness to a crime. In our experiments, a forensic examination was conducted on the robot to locate all useful evidence from multiple locations, including root-level directories using logical acquisition.

CCS CONCEPTS

• **Digital Forensics**; • **Humanoid Robot**; • **Internet of Things Forensics**; • **Logical Acquisition**; • **Zenbo**;

*Corresponding Author

ARES 2024, July 30–August 02, 2024, Vienna, Austria

ACM ISBN 979-8-4007-1718-5/24/07
https://doi.org/10.1145/3664476.3670906

ACM Reference Format:
Farkhund Iqbal, Abdullah Kazim, Aine Mac Dermott, Richard Ikuesan, Musaab Hassan, and Andrew Marrington. 2024. Forensic Investigation of Humanoid Social Robot: A Case Study on Zenbo Robot. In *The 19th International Conference on Availability, Reliability and Security (ARES 2024), July 30–August 02, 2024, Vienna, Austria.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3664476.3670906

1 INTRODUCTION

We are living in an age full of technological advancements that make our everyday activities easier. A decade ago, we would not have believed that we could use a smart mobile device to monitor our health through a smartwatch on our wrist. IoT is a bridge between the physical and digital worlds; an enabling technology with transformative implications for the way we interact with our immediate physical environment. While the applications can vary, this human-centric sensing and collection of data contributes to e-health, assisted living, and e-wellness. One growing area of wellness and assisted living is the increasing use of robotics and artificial intelligence (AI), deemed "*the next evolution of wellness*". Some of the most interesting applications in the healthcare sector include care and socially assistive robots, rehabilitation systems supporting patient recovery and assistance, training, and support for workers through learning initiatives, and more.

Studies forecast that by 2021, there will be 20.8 billion connected things in use worldwide. Robotics is one of the adaptations of IoT and it is commonly used in manufacturing areas. The worldwide expenditure on robotics, and Humanoids, will grow to $188 billion per annum [1]. The challenges and opportunities stemming from AI and robotics applications are currently subject to debate. One of the key factors attributed to the successful adoption of these humanoid robots is their ability to mimic human behavior in comparison to more traditional robots. This comfort and adaptability are likely to fuel the trend of employing humanoid robots in homes and businesses [1]. In terms of IoT, the key attributes of humanoid robots include web service functions, AI and machine learning analytics and predictions, device data synchronization across platforms, and communication with other IoT devices and services. These robots collect and process huge amounts of data, which in turn can be a good source of evidence in any digital forensic investigation. The commonality among currently available humanoid robots in the

healthcare domain is their mutually underlying Android-based OS, though this may not always be the case. Zenbo [2], Pepper [3], and Sanbot [4] humanoids have Android OS. Zenbo can connect to many smart home devices and has features that appeal to people across different age ranges [5]. Real-time monitoring based on the information gathered from the connected devices provides large-scale connectivity and greater insight into patient care, individual habits, and routines [6]. Zenbo and similar humanoid robots – posited as a key in future e-health environments and potentially increased applications in society – raise questions over data collected and stored. The growing development of humanoid robotics and human interaction has provided us with the possibility to develop applications that can adapt to the multifarious demands of modern society. Noninvasive sensors allow the collection and use of the extracted information, which simultaneously creates the need to protect human data [3]. As a result, digital forensic analysis of Zenbo as a case study presents a great insight into the possible data collected and stored, highlighting forensic artifacts for current and future Android-based robots. The primary objectives and contributions of this paper are highlighted below:

- To enhance the field of humanoid robot forensics; there is currently a limited number of published research on the subject. We interact with the internal memory of the physical Zenbo device (Zenbo Junior-Expansive) to identify and extract forensic artifacts.
- We identify, reconstruct, and interpret forensic artifacts of interest from the main databases maintained by the Zenbo app stored on the device and synced with the robot.
- We analyze databases, interpret event logs, and decode proprietary activity files stored on the Zenbo device to reconstruct the chronology and sequence of events.

2 LITERATURE REVIEW

Existing forensic methodologies are designed for different types of evidence sources. In the IoT, objects of forensics interest may not always be available or accessible [7]. A new era of best practices and digital forensics techniques to simultaneously verify and leverage physical and digital evidence within a changing regulatory landscape is required. MacDermott et al. [8] present a forensic analysis of wearable devices: Fitbit, Garmin, and HETP watches. The main objectives of the project were to run several extraction methods across the devices and their linked applications. The device contained historical data specifically for testing purposes. MacDermott et al. [9] present an analysis of securing 'things' within the healthcare IoT, offering insight into the security vulnerabilities within the IoT ecosystem per interconnected network and layers. The key themes were: Most IoT devices used a flash-based storage device; Many IoT devices are built on proprietary hardware; Some IoT devices store their data in the Cloud; Analysis of the interaction between IoT devices is necessary to collect data; Much of the evidence collected on IoT devices comes from network traffic analysis. Using a similar robot, Yankson et al., [10] evaluate the security vulnerability of IoT-based Zenbo robots. Several opened ports and the configuration lapses were observed as potential vectors of security limitations, based on open-source security analysis tools. Abeykoon et al. [11] proposed a forensics methodology that can be used for Robot Operating System (ROS) based devices. Humanoid robots are particularly susceptible to cyberattacks, as a potential intrusion on the communication channels between humans and humanoid robots is thought to be one of the easiest attack vectors (due to the use of "open communication networks"). The authors focused on ROS in their forensics approach. While there is no clear detailed process conveyed in these works, they highlight the importance and role of data extraction and analysis.

In terms of artifacts that can be collected from IoT devices, Chung et al. [12] studied Amazon's Alexa ecosystem from a forensics perspective. They highlighted that while client-side study for Alexa-enabled devices is important for a forensics investigation, the focus should also be put on the cloud-based component in Alexa's ecosystem because when a device is connected to Alexa's cloud it produces better information. The authors produced a proof-of-concept tool: Cloud-based IoT Forensic Toolkit (CIFT). Azfar et al. [13] proposed classifications for Android forensics artifacts based on 30 Google Play applications. The classification was accomplished by creating logical acquisitions of the Android device using the XRY tool, followed by investigating the extracted data files from each application.

Alabdulsalam et al. [14] highlighted that robust security measures are rarely incorporated into IoT designs. This is due to the production focus balancing limited resources, product miniaturization goals, and required device functionality – the combination of which can make it challenging to include and maintain adequate security safeguards. Le-Khac et al. [15] discussed the matter of how smart vehicles differ from traditional vehicles in their connectivity and the amount of data that they store about their users. Li et al. [16] remarked that Android-based smartphones are the most popular phone OS in the world and because of this, the need for a sound digital forensic approach when dealing with Android smartphones was advocated. The research paper focused on Extended File System version 4 (Ext4) and considered both NAND and embedded Multi-Media Controller (eMMC) non-volatile memories for forensics analysis and physical data acquisitions while analyzing the differences between these two types of memory.

Saracino et al. [17] also remarked that Android occupies the largest market share of operating systems used in mobile phones and highlighted that the openness of the Android OS renders it more vulnerable to malware.

3 METHODOLOGY AND IMPLEMENTATION

This study adopted the IoT forensic model described by Li et. al [16] in a scenario where an IoT device is a witness to a crime, i.e. data stored on the device can directly implicate a person or tell a story as to what has occurred. We performed a set of controlled experiments that involved several scenarios, each one referring to a specific usage (e.g. initiating and receiving video calls, sending voice commands, getting alert reminders, using storytelling mode, etc.) during which a typical record of user activities has taken place. These activities enabled us to generate data to forensically examine the IoT device (the Zenbo robot), examine the companion app, and examine the companion network. Details of the investigative scenario and stages of methodology are described below:

3.1 Methodology

Because our research was conducted in a controlled environment the standard procedures that apply to securing the crime scene, location, and seizure of the device are not considered. Moreover, because the Zenbo robot is an IoT device running on a well-known operating system (Android version 6.0.1) our experiments fall into the category of device-specific forensics. Due to the sheer number of IoT devices available and their diversity, there is no standardized methodology that can be applied to all of them. However, during our research, we have decided to follow the general main stages: data generation, data acquisition, data analysis, and presentation of results. Refer to Figure 1.

We have set up a scenario that involved the use of the Zenbo Robot, Zenbo Master app, and Zenbo App Builder, two applications that interact with the device, which were also analyzed in this study.

Zenbo Master is an app developed by AsusTek Computer Inc. available on both Google Play and App Store. It allows the owner of Zenbo or associated family members and friends to interact with the robot remotely. Features include initiating and receiving video calls, sending voice commands to the robot, getting alert notifications on reminders and emergencies, capturing, and viewing photos stored on Zenbo, etc. ASUS does not provide any other online console to control and change the settings of Zenbo. Zenbo App Builder is a visual development platform designed by ASUSTeK to provide users with an interface to easily develop their apps that can make use of Zenbo's functionalities like movement expressions [18]. Zenbo App Builder is accessible through a browser and as an app for Android and iOS. The developed apps have the extension of "ZBA", and they can be copied to Zenbo over the Internet, by connecting both the device and robot to the same Wi-Fi access point and then enabling the Zenbo App Builder app on the robot. This means that Zenbo App Builder is available both on the devices used for development and the robot itself.

A point worth mentioning is that Zenbo App Builder allows for media upload as part of the developed apps. The app, when developed through browsers allows for non-media files, like executable files, to be added into the developed app, but the same is allowed when the app is developed on iOS and Android unless we change the extension of the files. Another point to note is that while building the app through a normal browser requires ASUS login credentials, the same is not true for developing apps through Zenbo App Builder on smartphones. The following points summarize the methodology:

- Data generation: (a) Reset the robot operating system, (b) Acquire initial forensic image from the device, (c) Install applications on chosen devices, (d) Initial setup, and (e) Operate the robot.
- Data acquisition: (a) Acquire the final forensic image from Zenbo, (b) Collect data from The Zenbo Master and Zenbo App Builder, (c) Use the hashes from previous acquisitions to identify changes and extract useful artifacts. This process ensures forensic soundness.
- Data analysis: (a) Analyze Zenbo Master app data, (b) Analyze Zenbo App Builder data, (c) Analyze Zenbo robot data Logical acquisition was used for all the extractions of artifacts and the Android Debug Bridge (ADB) was the chosen extraction method ([17]). The tool was installed on the Windows 10 forensic workstation, and since the USB port is used to connect Zenbo to the forensic workstation. Furthermore, the USB Debugging mode was enabled on the Zenbo OS. This forms the process of data acquisition and extraction. A variety of tools can be used to perform analysis of the data found in IoT devices. We apply mature solutions that are used by professional forensic examiners: Autopsy and OSForensics as well as two commercial solutions FTK 6.4 and Cellebrite UFED touch.

3.2 Implementation

Implementation followed closely the methodology described in the previous section. To ensure the authenticity of the results (and negate the possibility of previous data affecting the results) factory reset was performed on the robot's OS. Then an initial forensic image was obtained of the fresh OS, and the SHA-256 hash values were calculated for each of the files in the image for comparison purposes in a later stage. Applications were installed on an Android phone (running Android version 6.0) with the Zenbo Master app, and the Zenbo App Builder app installed on it. This process mimics the normal operation of a Zenbo robot in a typical smart home/health deployment. The initial setup involved the connection of the application with the robot and setting up generic test data such as user profile, user ID, user relationship, and user profile picture. To generate enough data for the analysis the robot was operated for two weeks while utilizing all the functions and features provided by both the robot and the phone applications. The actions performed included answering calls, making outgoing calls, using the voice command function, and utilizing the wide array of functionalities provided by Zenbo. This meant that when analyzing the results generated by forensic analysis of the devices, we had a clearer focus on where to explore. The Zenbo Master app and Zenbo App Builder, two applications that interact with hardware, were also analyzed in this study. Additionally, specific robot information and artifacts including call logs, calendar entries, friends and family listings, voice commands, and storytelling mode were analyzed for data remnants. After two weeks of generating data, a second forensic image was acquired, and the SHA-256 hash values were calculated for each of the files in the image. By comparing the change in the hash values of the files in both forensic images we were able to identify all the affected files without missing any data. The aim was to find and/or acquire database files and other files of interest that may hold user data and items of forensic interest. As there is both limited literature on analyses of humanoid robots and affiliated, viable data from such for forensic analysis/extraction of data, we will detail our findings from each table of relevance as the information may be useful for future examiners. Intrusive or rooting methods may reveal more items of interest on the target devices in cases of seeking to examine the deleted files, but such methods may leave new fingerprints. As such, no intrusive or rooting methods were used throughout the experiment and thus the integrity of the evidence remained intact. In our scenario no data was hidden or deleted with purpose, thus ADB was used to perform the acquisition of the identified files of interest. The robot was connected to our lab PC with USB Debugging mode enabled. After

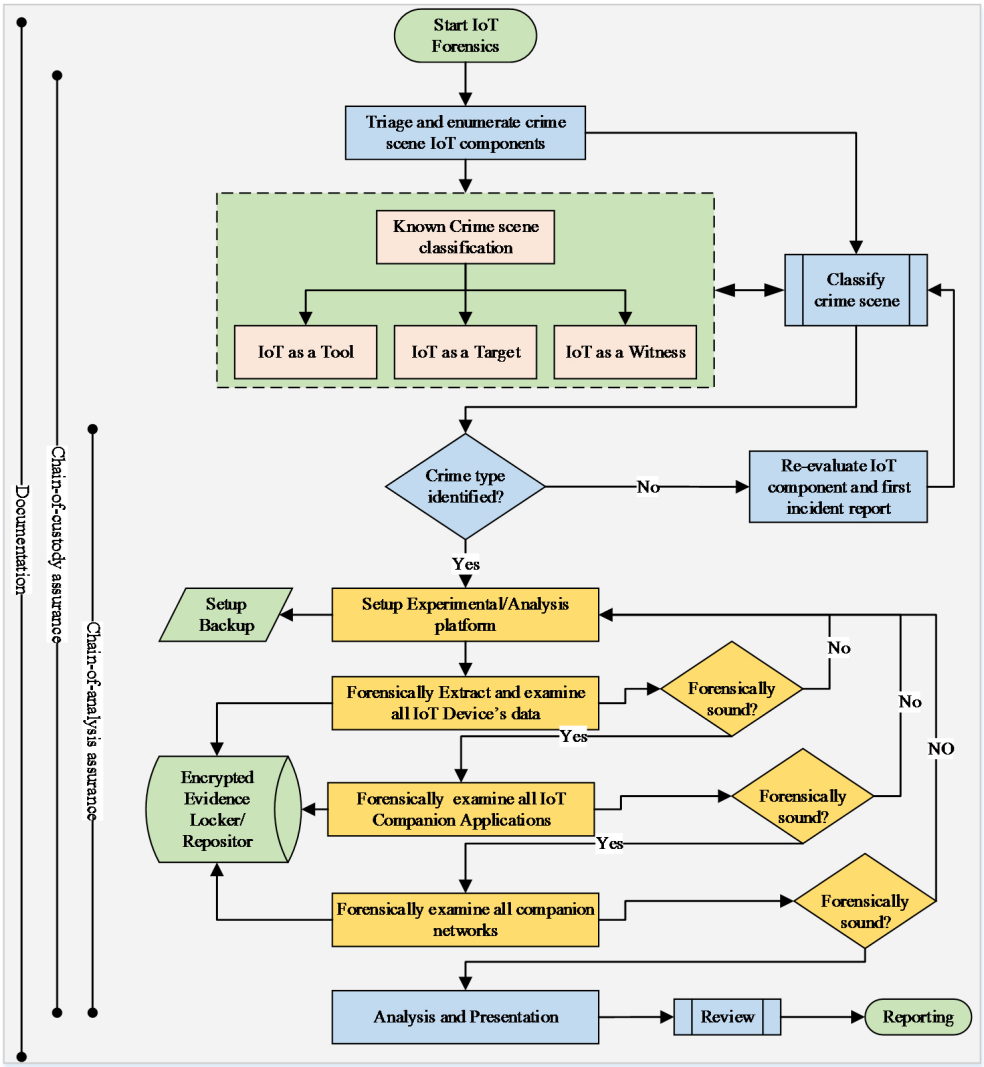

Figure 1: Investigative scenario

accessing the robot shell through ADB, the ADB pull command was able to extract all the directories that did not require the root privilege. The ADB pull command was not able to extract the applications' data since they are stored in the /data/data/ directory that requires root privilege. To overcome this issue, the ADB backup command was used to extract the applications' data without the need of rooting the robot and altering the system files. It must be noted that while there is a tendency for Android devices to be locked by a passcode/pattern/password, this forensic challenge is not present with Zenbo. The robot has a lock screen feature designed to protect the user interface like any normal Android tablet and smartphone. Zenbo owners may not want to use this feature because this will disable the Zenbo mode - the mode in which the robot is capable of facial expression and personal interaction.

4 ANALYSIS

It should be noted that in terms of compatibility, both Android and iOS smartphones can be linked with the Zenbo robot. In our experiments, we used the former (running Android version 6.0) with the Zenbo Master app and the Zenbo App Builder app installed on it. The aim was to find and acquire database files and other items of interest that may hold user data and information of forensic interest. There is limited literature on analyses of humanoid robots and affiliated, viable data from such devices. For forensic analysis/extraction of data, we will detail our findings from each table of relevance as this information may provide useful insight for future examiners. In terms of the procession, we followed a forensic acquisition of the items with an analysis stage. The findings from our experiments are presented in the subsequent sections. In short, we begin with Zenbo-specific applications that interact with Zenbo hardware and then discuss further findings on the actual Zenbo

robot that presents a good source of information for the forensic examination process.

4.1 Zenbo Master Findings

Analysis of the smartphone directories revealed that Zenbo Master. 2.1.17 is installed under the name "com.asus.robot.avatar". The identified package was extracted from the phone. Currently, Zenbo Master is considered the main method for remote interaction with Zenbo. Analysis of the package content revealed that the most valuable information is stored in the database file "asusRobotVideophone.db" including data from the Zenbo Master, the user, and interactions with the Zenbo robot. The most interesting findings from the "asusRobotVideophone.db" file are as follows:

- The table "authority_management_contacts" found in the mentioned database file stores the user profile name, user relationship, and the user profile picture. In the forensic process, these details are to match the data on the Zenbo robot to ensure the correct device is considered in the investigation.
- It was found that table "bind_robot_list" keeps records on the Zenbo Master user and the Zenbo robot that the application is connected to. Details found are similar to the data found in the previous table, however, it was also discovered that the name of the robot to which the app is connected is stored in this table. The name can be configured on the Zenbo robot during its initial setup.
- All calls initiated directly from the Zenbo robot as well as via the application are stored in the "call_logs" table. The records include the date and time of the call, the UID of the Zenbo robot, call duration (shown in seconds), and whether the call was answered or not. The column "Type" in the table has three main values (0,1,2); 0 indicates calls from the Zenbo Master app to the Zenbo robot, 1 indicates calls from the Zenbo robot to the Zenbo Master app, and 2 is for emergency calls or emergency alerts from the Zenbo robot to the Zenbo Master app. The "User ID" column was found empty when it was supposed to hold the Zenbo User ID.
- Table "contacts" holds the UID and the Zenbo Name data. These values were already recovered from other tables, but it helps to ensure the integrity of data by comparing what is recovered in this table to the data recovered earlier.
- The table "notification_center" stores data about all notifications sent to the Zenbo Master app, from the Zenbo robot, for user attention. Column "Title" has one of two strings – "Emergency notification" or "Successful System Update" – and the same is highlighted by value 1 or 2 respectively in the "Types" column. The "Content" column has the standard text "One of your family members needs help! Please confirm quickly!", which is sent in a case where the robot detects an emergency, for example, an injured family member nearby. The table timestamps for each record, and it has a column to indicate whether the notification was answered.

The "speak_out_log" is important because it holds logs of all speak-out commands sent to Zenbo through the Zenbo Master app shown in sequence. Speak out commands are text commands sent through the Zenbo Master app to the Zenbo robot for it to speak out. The table shows text that was sent as a command, user ID, Zenbo ID, and timestamp. Other files in the recovered package also reveal useful artifacts. For example, file "share_pref, 2E8585D2-AF184AE88E7B25CB4B71EF5.xml" provides time differences between server time and local time. Since Zenbo support is based in Taiwan, the server time is saved in the GMT+8 time zone. The location of the robot can therefore be inferred by the difference between server time and local time. The main forensic finding is that all XML files have the timestamp of when files were created/updated on the smartphone using Zenbo Master. Although the remote connection between Zenbo Master and Zenbo robot is made over the internet, there is no single IP address found in the package. With this in mind, forensic analysis of the Zenbo Master app alone is likely to be inadequate for most investigations but presents information that can complement findings from other sources. Table 1 summarizes the findings from this Zenbo Master package.

4.2 Zenbo App Builder

In our experiments, we used two different Android smartphones to develop apps and transfer them onto the Zenbo robot. The package name for Zenbo App Builder 1.0.3 on smartphones is "com.asus.appbuilder.editor". The analysis of the package did not reveal any valuable content that could become evidence or support data found in other sources. We were unable to retrieve any details about the apps that were developed on the device, no files, no database, etc. The recovered data does not link to any IP addresses of the user's smartphone, although data is copied to Zenbo over the Internet. So, even if an investigator seized the smartphone, there is no way of linking it to the apps imported into the robot. This means that in cases where malware was found on Zenbo, an investigator cannot confirm a suspect's involvement in installing that malware from their smartphone.

4.3 Zenbo Robot

As was expected the analysis of the robot provides the most valuable information. As a main source of data, it stores data that can be found on the connected apps but also much more viable information with the potential to become evident during forensic investigation. What follows are further subsections describing various locations where artifacts of forensic value can be found.

4.3.1 Zenbo App Builder Package. The package name for Zenbo App Builder 1.0.3 on the Zenbo robot is "com.asus.robot.appbuilder" and what follows are the findings from this package:

- The database file "BlocklyEngine.db" reveals all the properties of the apps that were copied onto Zenbo. The "ID" field has a sequential numeric value, and the highest number is the latest app imported onto Zenbo. This field needs to be compared with the field "Import Date" during analysis to ensure no tampering was done on the data because "Import Date" should be the latest date value for the highest "ID" number. The "Use Date" field reveals the date the app was last used on the Zenbo robot. Other data revealed in the database are all related to the set properties on the app at the time of development, like App Version and App Name.
- The folder "ZenboScriptFolderPlace" contains sub-folders where each refers to a single application imported on the device. The sub-folder names match the text in the "app_id"

Table 1: Summary of Zenbo Master app package findings

Filename	Findings
asusRobotVideophone.db	•User profile, user ID, user relationship, and user profile picture. •Call logs. •Information about Zenbo is linked to the app. •Logs of speak-out commands. •Notifications sent to Zenbo Master from Zenbo robot.
share_pref, 2E8585D2-AF18-4AE88E7B-25CB4B71EF5.xml	Server offset time.
SHARE_PREF.xml	•The Gmail account is used to configure the Zenbo Master app. •Configured name for Zenbo.
zenboImageVersion2E8585D2-AF184AE8-8E7B-A25CB4B71EF5	Robot version number.
allSpace2E8585D2-AF18-4AE88E7BA25CB4B71EF5	Total storage available on Zenbo.
useSpace2E8585D2-AF18-4AE88E7BA25CB4B71EF5	Space used on Zenbo. This is indicative as per the last time the app was connected to Zenbo.
zenbo_language.xml	The main language is configured on the app, and it indicates the main language of the user.
com.asus.robot.avatar_preferences.xml	Whether the user is an admin or a normal user.
battery, 2E8585D2-AF18-4AE88E7B-A25CB4B71EF5.xml	Shows the battery level of the Zenbo robot recorded at the time when the app was last connected to Zenbo.

field of BlocklyEngine.db. In case these two do not match, this may indicate data tampering. While the apps are developed as ZBA files, the data imported onto Zenbo are broken into folders with different contents that make up the apps. The timestamp of folders refers to the acquisition time rather than the creation time, but actual creation time can be extracted from the files within the folders.

- Each app folder has four main files: BlocklyManifest.xml, blocksStack.xml, executeScript.html, and icon.png. Additionally, the folder includes any media, audio, and other files that were packed with the app before importing onto Zenbo. This means that if a malicious file was imported on a device together with the app, it can be found in the folder.
- The file "executeScript.html" shows the full functionality of the app and this can be helpful in case any imported app is acting maliciously on the robot. A snapshot of the file is seen in Figure 2.
- The file "BlocklyManifest.xml" holds the properties of the app, many of which can be found in the BlocklyEngine.db database file. Comparing and matching the properties in these two files is important to ensure data integrity.
- The file "blocksStack.xml" also holds app functionality but in XML format. The file holds information about some of the changes that were made to the app in its earlier versions that may not exist anymore in the most recent version of the robot. This finding helps in case the app was made to run some malicious files on the robot and then to cover this up, another version was imported with modified functionality.

4.3.2 Zenbo Users and Call Logs. Package "robot.asus.com.robotprofileprovider" stores information about robot users. Such information can be added by the admin using the voice command

```
<html><head><title></title></head><body><script language="JavaScript">
Zenbo.init();
Zenbo.start('block_id_)XKFK{[/4L2+8GmV%3D`');
Zenbo.say(100,'Hi, my name is Zenbo', 'block_id_LAn2e|)mCch/=+_%aE37');
Zenbo.say(100,'How Are You Amera?', 'block_id_CBe;,H*zy}IDPFmG4ZV[');
Zenbo.setExpression(7, 'block_id_6X,rESXoE*98nr_b?c@f');
Zenbo.headMove(true,1,20,10, 'block_id_yjbx?s%ttEaM06#~#Iw.');
Zenbo.end();
 </script></body></html>
```

Figure 2: Snapshot of "executeScript.html" file

function. Information about relatives and friends can be very useful during the investigation. All contacts can be found in the database file asusRobot.db specifically in the table "user_profile". This table has details about all users that were configured on the Zenbo robot and the field "relative" specifies the type of relationship with Zenbo's owner/primary user. Furthermore, this database file also holds the profile picture of the "relative", which can help to positively identify the relative. Other details that can be found about these associates include their birthday, email address, gender, and whether they are an admin or a normal user of the Zenbo robot. In the table "call_logs", the list of all calls that were made to and from Zenbo was found. The calls can be made only between the robot and an application. Comparison of call logs data recovered from this table with data from the Zenbo Master's database asusRobotVideophone.db can provide information if data were tampered with as both files should contain the same information. While reading information from the "cus_id" field allows identification of the customer ID with whom the call was initiated it is not possible to identify the call origin or the IP address of the person.

4.3.3 Calendar and To-do List. Analysis of the calendar and to-do list items can help reconstruct user activity in an investigation.

Such details can be found in the package com.asus.robot.reminder 1.0.0. Zenbo reminds users of scheduled activities.

4.3.4 Voice Command Responses. The Zenbo robot receives voice commands and responds with confirmation voice responses followed by actions. Analysis of the robot reveals that in the path "/sdcard/Logs/DS" we have found many text files with the prefix "AsrSet", which contain different voice commands and the corresponding Zenbo function in response to the command. Initially, we thought that this might be all the voice commands issued by the user and Zenbo's corresponding action, but ultimately, we concluded that these were default text files since they can be found on Zenbo even after a factory reset. No traces of voice commands issued by the user were found, thus it was concluded that such information is not saved by the robot. Nevertheless, files "Record********TtsInfo.txt" where stars represent the year, month, and day are considered useful. These files are in the "/storage/self/primary/Logs/DS" directory and one file per day is being created. Moreover, each file holds all Zenbo answers that came in response to voice commands from the user. There are no timestamps available for each voice response from Zenbo thus it is not possible to group the responses in chronological order.

4.3.5 Storytelling. Storytelling for children is one of Zenbo's features in which the robot narrates built-in stories. It was possible to recover artifacts related to this functionality, such as images, video, audio files, and different JavaScript files for each story. One of the cases in which this artifact can prove to be important is when one of the stories was modified to function maliciously and activate the camera or audio recording to facilitate unauthorized video and/or audio monitoring of Zenbo's local environment. The storytelling artifacts are in the path "/system/media/storytelling".

4.3.6 System Apps. There are different apps installed on Zenbo, some of which provide access to the core functions of the robot to its user, while others provide system functions that are to be used only by the robot. Due to popularity of the Android devices, they are one of the most targeted platforms by cyber criminals [19]. Because of the relative ease of installation of third-party applications, they are especially susceptible to outside threats thus it is important to list artifacts relating to app installation. The findings of the apps are as follows:

- The "/system/app" directory contains system-related apps (APK files), which allow Zenbo to perform core system functions. Analysis of these applications can provide information if the app is infected by malware. Since we performed a logical acquisition, the timestamp of extracted apps in our experiments did not correspond to the app installation time, but rather the time when the package was extracted from the robot.
- The "/system/priv-app" directory also contains APK files, corresponding to apps that are supposed to be used by the system in the background, for example, apps for Google backup, uploading logs, and updating system configurations.
- While the first two mentioned paths provide the raw APK files, the database file "frosting.db", from package com.android.vending 17.9.17 provides a list of all installed packages with their actual installed paths. Figure 3 shows a snapshot of the database file.

PK	APK Path
jackpal.androidterm	/data/app/jackpal.androidterm-1/base.apk
com.google.android.tts	/data/app/com.google.android.tts-2/base.apk
com.asus.robot.storytelling	/system/app/RobotStoryTelling/RobotStoryTelling.apk
com.android.providers.telephony	/system/priv-app/TelephonyProvider/TelephonyProvider.apk
com.asus.robotframework.robotposmanualset	/system/app/RobotPosManualSet/RobotPosManualSet.apk
com.android.providers.calendar	/system/priv-app/CalendarProvider/CalendarProvider.apk
com.asus.zenbohichannel	/system/app/ZenboHichannel/ZenboHichannel.apk
com.android.providers.media	/system/priv-app/MediaProvider/MediaProvider.apk
com.asus.robotframework.sampling	/system/app/Sampling/Sampling.apk

Figure 3: List of installed packages and their paths

4.3.7 Other Forensics Artifacts. Even though most of the results were contained within the applications' data, the examination was extended to cover all files and directories that have been acquired from the robot to cover all possible artifacts. While exploring the "build.prop" file, we found useful build artifacts like the Android version, the ASUS version, and the last security patch installed on the device. It was also observed that the preferred network type set on the robot was WiFi (this is always true for the network connection for smartphones and tablets as they do not come with Ethernet ports for wired connection).

Knowing the robot's running time and the amount of time spent on the processing can provide valuable information to forensic examiners in confirming certain events and providing information on timeline events. The total time consumed by the CPU and those related to the user and system tasks can be found in the paths: (1) /acct/cpuacct.usage, (2) /acct/cpuacct.stat, and (3) /acct/cpuacct.usage_percpu. Moreover, multiple log files were identified and located, including system event logs, diagnostic logs, kernel logs, and ASUS server debugging logs. This information allows the identification of actions and the time they were performed by the robot. Like other devices based on Android, most media files were found in the DCIM and Download directories. The DCIM directory holds images and videos that are either taken by the robot or stored on it, while the Download directory holds various types of files downloaded from the Internet. It is worth mentioning that it is recommended to analyze the media files' metadata as they include a valuable source of information for forensic investigators. Recovering the built-in camera profile and settings may help in sourcing a specific photo back to Zenbo based on the image properties. For example, we can see the camera resolutions and supported image sizes with a Zenbo camera, and if we have a given image that doesn't match the acquired sizes, we can exclude Zenbo as the source for said image. A similar case can be seen with the findings on supported fonts on Zenbo, in the sense that a piece of writing with an unsupported font type may also be excluded from Zenbo as the source. No data was found about Zenbo Master on the robot itself. Table 2 summarizes all sources of data and possible findings recovered from the Zenbo robot.

5 RESULTS EVALUATION

The experiments carried out for this paper highlighted very interesting results. Zenbo Master revealed richer data than Zenbo App Builder, which, unlike Zenbo Master, has limited data about its

Table 2: Zenbo findings

Source	Findings
BlocklyEngine.db	Details of apps imported on Zenbo
ZenboScriptFolderPlace folder	
robot.asus.com.robotprofileprovider	Information about Zenbo users Details about calls
com.asus.robot.reminder	Information about activities scheduled by the user
/sdcard/Logs/DS	Zenbo default responses
/storage/self/primary/Logs/DS	Created Zenbo responses to commands in a text format
/system/media/storytelling	Details of robot's storytelling feature
/acct/cpuacct.usage	CPU-related details like a robot CPU up time and processing details.
/acct/cpuacct.stat	
/acct/cpuacct.usage_percpu	
/system/etc/fonts.xml	Font-related data like the supported fonts.
/system/etc/fallback_fonts.xml	
/system/etc/camera_ddr.sh	Camera profiles and settings.
/system/etc/camera_profiles.xml	
/system/etc/camera_realsense.gmin.xml	
/system/etc/camera_realsense.xml	
/system/etc/audio_effects.conf	Audio-related configurations and profiles.
/system/etc/audio_policy.conf	
/system/etc/event-log-tags	Tags that can be seen in event logs and correspond to specific events.
/system/etc/permissions	This folder contains different hardware and software features that Zenbo provides. For example, "android.hardware.bluetooth" has Bluetooth features.
/sdcard/Logs/last_kernel_logs	All the messages of the kernel.
/sdcard/Logs/UploadRecord.txt	Date and time value that gets updated every 24 hours. It may be related to syncing to server.
com.android.documentsui	File "recent.db" has timestamps of when apps were last used.
com.google.android.apps.photos	Files "gphotos0.db" and "local_trash.db" show details where photos are stored and the location of deleted photos.
com.android.providers.settings	Screen brightness, dim screen setting, auto time detection, notification setting when new Wi-Fi access point is available, USB mass storage enabled/disabled.

functions (recorded in allocated space) on both the robot and smartphone. The main limitation we have found is that although these two apps connect to Zenbo over the Internet, Zenbo does not record any UID from the smartphone source device, nor does it record the IP address – in essence, there is no network data stored on Zenbo and affiliated apps, making it difficult to trace specific source devices back to Zenbo, (e.g., to determine how particular malware was installed on the robot). Other Zenbo-specific artifacts covered were related to the Friends & Family settings and Storytelling functionalities previously addressed. We highlighted the importance of these two artifacts for a digital forensics investigation and how they can add value to it, by identifying the associates of Zenbo's owner/primary users. Moreover, we highlighted different directories, files, and packages that contain some potentially interesting artifacts depending on the case being investigated. Abeykoon et al. [11] were useful in suggesting data files and their importance when analyzing ROS. The authors conveyed how ROS can be composed of pictures, cloud data, database files, and file/memory logs. Our analysis found each of these files and confirmed that the Zenbo OS is more like an Android phone device rather than a specialist closed-off ROS. Our work involved logical analysis of the Zenbo robot but expanding it to explore the acquisition and analysis of the Random-Access Memory (RAM) image to locate other data fragments will be considered in the future. We have used standard open-source forensics tools such as Autopsy, OSForensics, and commercial tools including FTK 6.4 and Cellebrite UFED touch.

6 CONCLUSIONS AND FUTURE WORK

The rate of development, deployment, and integration of IoT devices into everyday life in modern societies has dramatically increased over the past decade. The nature of humanoid robots is intended to help operate near humans, which will necessitate more data being captured and accessed by said robots to enhance and customize their services, especially when robots have a long-term primary user. This can make them very useful in digital investigations for gathering evidence. In this paper, we examined Zenbo, one of the relatively inexpensive humanoid robots available on the market. We performed a logical acquisition of Zenbo, and smartphones synchronized with Zenbo and analyzed it along with its mobile apps to locate interesting findings pertinent to forensic examination. Further exploration and examination could extend this work by expanding it to explore the acquisition and analysis of the Random Access Memory (RAM) image to locate artifacts specifically related

to the robot rather than the normal Android platform. Moreover, options for the physical acquisition of Zenbo (e.g. through installing a rootkit) should be investigated. This would allow for the physical image with all the different available partitions to be analyzed, which may increase the number of useful artifacts that can be detected by adding the contents of unallocated space.

ACKNOWLEDGMENTS

This study is supported with Research Incentive Funds (activity code: R23064), Research office, Zayed University, Dubai, United Arab Emirates.

REFERENCES

[1] IDC. Worldwide spending on robotics will reach $188 billion in 2020 fueled by new use cases and expanding market acceptance, 2020. Available at: https://www.idc.com/getdoc.jsp?containerId$=$prUS45800320

[2] Yankson, B., Iqbal, F., AlMaeeni, F.: Social Robots Privacy Enhancement Using Colored Petri Net (CPN) for Behavior Modeling: A Case Study of Asus Zenbo Robot. Int. Conf. Cyber Warf. Secure. 18, 457–464 (2023). https://doi.org/10.34190/iccws.18.1.1018

[3] Barra Paola, Bisogni Carmen, Rapuano Antonio, Abate Andrea Francesco, and Iovane Gerardo, "HiMessage: An Interactive Voice Mail System with the Humanoid Robot Pepper," *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, Fukuoka, Japan, 2019, pp. 652-656, doi: 10.1109/DASC/PiCom/CBDCom/CyberSciTech.2019.00123,

[4] Mail System with the Humanoid Robot Pepper," 2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), Fukuoka, Japan, 2019, pp. 652-656, doi: 10.1109/DASC/PiCom/CBDCom/CyberSciTech.2019.00123

[5] Nicolas Oros and Jeffrey L Krichmar. Android™ based robotics: Powerful, flexible and inexpensive robots for hobbyists, educators, students and researchers. Cognitive Anteater Robotics Laboratory. University of California, Irvine, 2013. Available from: https://www.socsci.uci.edu/$\sim$jkrichma/ABR/

[6] Zenbo Intelligent Robot, ASUS, 2016. Available from: https://www.asus.com/CommercialIntelligent-Robot/Zenbo/

[7] Heather Mahalik, Rohit Tamma, and Satish Bommisetty. Practical Mobile Forensics. Packt Publishing Ltd, 2016

[8] Aine MacDermott, Thar Baker, and Qi Shi. IoT forensics: Challenges for the ioa era. In 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp. 1–5. IEEE, 2018

[9] Aine MacDermott, Phillip Kendrick, Ibrahim Idowu, Mal Ashall, and Qi Shi. Securing things in the healthcare internet of things. In 2019 Global IoT Summit (GIoTS), pp. 1–6. IEEE, 2019

[10] Benjamin Yankson, Tyler Loucks, Andrea Sampson, Chelsea Lojano: Robots Security Assessment and Analysis Using Open-Source Tools. Int. Conf. Cyber Warf. Secur. 18, 449–456 (2023). https://doi.org/10.34190/iccws.18.1.1019

[11] Iroshan Abeykoon and Xiaohua Feng. A forensic investigation of the robot operating system. In 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pp. 851–857. IEEE, 2017

[12] Hyunji Chung, Jungheum Park, and Sangjin Lee. Digital forensic approaches for amazon alexa ecosystem. Digital Investigation, 22:S15–S25, 2017

[13] Abdullah Azfar, Kim-Kwang Raymond Choo, and Lin Liu. Forensic taxonomy of android productivity apps. Multimedia Tools and Applications, 76(3):3313–3341, 2017

[14] Saad Alabdulsalam, Kevin Schaefer, Tahar Kechadi, and Nhien-An LeKhac. Internet of things forensics–challenges and a case study. In IFIP International Conference on Digital Forensics, pages 35– 48. Springer, 2018

[15] Nhien-An Le-Khac, Daniel Jacobs, John Nijhoff, Karsten Bertens, and Kim-Kwang Raymond Choo. Smart vehicle forensics: Challenges and case study. Future Generation Computer Systems, 109:500–510, 202017. fd

[16] Zhi Li, Bin Xi, and Shunxiang Wu. Digital forensics and analysis for android devices. In 2016 11th International Conference on Computer Science & Education (ICCSE), pp 496–500. IEEE, 2016

[17] Andrea Saracino, Daniele Sgandurra, Gianluca Dini, and Fabio Martinelli. Madam: Effective and efficient behavior-based android malware detection and prevention. IEEE Transactions on Dependable and Secure Computing 15, no. 1 (2016): 83-9720. ds

[18] Sarah Buhr. An amazon echo may be the key to solving a murder case. Tech Crunch, December, 27:2016, 2016

[19] TaeGuen Kim, BooJoong Kang, Mina Rho, Sakir Sezer, and Eul Gyu Im. A multimodal deep learning method for android malware detection using various features. IEEE Transactions on Information Forensics and Security 14, no. 3 (2018): 773-788.