Prototyping an Anomaly-Based Intrusion Detection System for Internet of Medical Things Networks

Georgios Zachos
Instituto de Telecomunicacoes
Aveiro, Portugal
Faculty of Engineering and Science
University of Greenwich
Chatham Maritime, UK
g.zachos@av.it.pt

Georgios Mantas
Instituto de Telecomunicacoes
Aveiro, Portugal
Faculty of Engineering and Science
University of Greenwich
Chatham Maritime, UK
gimantas@av.it.pt

Ismael Essop
Faculty of Engineering and Science
University of Greenwich
Chatham Maritime, UK
i.a.essop@greenwich.ac.uk

Kyriakos Porfyrakis
Faculty of Engineering and Science
University of Greenwich
Chatham Maritime, UK
k.porfyrakis@greenwich.ac.uk

Jose C. Ribeiro
Instituto de Telecomunicacoes
Aveiro, Portugal
jcarlosvgr@av.it.pt

Jonathan Rodriguez
Instituto de Telecomunicacoes
Aveiro, Portugal
Faculty of Computing, Engineering and Science,University of South Wales
Pontypridd, UK
jonathan@av.it.pt

***Abstract*—Over the past few years, the Internet of Things (IoT) is transforming the healthcare sector through the introduction of the Internet of Medical Things (IoMT) technology, whose purpose is the improvement of the patient's quality of life. Nevertheless, IoMT networks are still vulnerable to a wide range of threats because of their heterogeneity and resource-constrained characteristics. Thus, novel security mechanisms, such as accurate and efficient anomaly-based intrusion detection systems (AIDSs), taking into consideration the inherent limitations of the IoMT networks, are required to be developed before IoMT networks reach their full potential in the market. In our previous work, we presented the system architecture for a novel hybrid AIDS for IoMT networks. In this paper, we expand it by presenting details of the implementation process that led to a prototype of the proposed AIDS. Our target is this work to serve as a guidance for other researchers or engineers to develop their own specific implementations of AIDSs for IoMT networks.**

Keywords—Internet of Medical Things (IoMT), intrusion detection system (IDS), machine learning algorithms, anomaly-based intrusion detection, IoMT AIDS implementation

I. INTRODUCTION

The existence of the Internet of Things (IoT) is transforming the healthcare sector with the introduction of the Internet of Medical Things (IoMT) technology, which aims to improve the patient's quality of life by enabling personalized e-health services without limitations on time and location [1]–[5]. However, the wide range of different communication technologies (e.g., WLANs, Bluetooth, Zigbee) and types of IoMT devices (e.g., medical sensors, actuators) incorporated in IoMT edge networks are vulnerable to various types of security threats, and this, in turn, raises many security and privacy challenges for such networks, as well as for the healthcare systems relying on these networks [6]–[9] . For instance, an adversary may compromise IoT-based healthcare systems through their IoMT networks in order to manipulate sensing data (e.g., by injecting fake data) and cause malfunctions to the compromised IoT-based healthcare systems that, in turn, will jeopardize the integrity or the availability of the healthcare services provided by these systems [2]. Consequently, security solutions protecting IoMT networks from attackers are essential for the acceptance and wide adoption of such networks in the coming next years.

Nevertheless, the high resource requirements of complex and heavyweight conventional security mechanisms cannot be afforded by (a) the resource-constrained IoMT edge devices with limited processing power, storage capacity, and battery life, and/or (b) the constrained environment in which the IoMT devices are deployed and interconnected using lightweight communication protocols [10]. Therefore, novel security mechanisms are necessary to be developed in order to address the pressing security challenges of IoMT networks in an effective and efficient manner, considering their inherent limitations stemming from their resource-constrained characteristics, before IoMT networks gain the trust of all involved stakeholders and reach their full potential in the market [8], [11].

Toward this direction, the industry and research community currently foresee anomaly-based intrusion detection as a promising security solution that can play a significant role in protecting IoT networks, as long as novel lightweight anomaly-based intrusion detection systems (AIDSs) are developed [10], [12]–[15]. Currently, in the literature, there are only two related works on AIDSs for IoMT networks. The first IoMT AIDS is presented in [16] and the second one was proposed by us in [17]. In particular in [17], we presented the system architecture for a novel hybrid AIDS for IoMT networks, leveraging host-based and network-based techniques to reliably monitor and collect log files from the IoMT devices and the gateway, as well as traffic from the IoMT edge network, while simultaneously considering the computational cost. The detection process of the proposed AIDS is to be implemented by the detection engine running on the gateway of the IoMT edge network and relying on machine learning (ML) techniques, considering the computation overhead, in order to detect

This research work is supported by the European Regional Development Fund (FEDER), through the Regional Operational Programme of Lisbon (POR LISBOA 2020) and the Competitiveness and Internationalization Operational Programme (COMPETE 2020) of the Portugal 2020 framework [Project IEoT with Nr. 69537 (POCI-01-0247-FEDER-69537)]; This research work was also sponsored in part by the NATO Science for Peace and Security Programme under grant SPS G5797.

abnormalities in the collected data and thus identify malicious incidents in the IoMT network.

Therefore, in this paper, the main objective is the expansion of our work in [17] through the presentation of details of the implementation process that led to a prototype of the proposed AIDS in [17]. Our target is this work to serve as a guidance for other researchers or engineers to develop their own specific implementations of AIDSs for IoMT networks.

Following the introduction, this paper is organized as follows. Section II briefly reviews the architecture of the AIDS for IoMT networks proposed in [17]. Section III presents the details of our developed prototype implementation. Finally, Section IV concludes this paper and provides hints for future work.

II. System Architecture Overview

The proposed AIDS consists of two main components, as illustrated in Fig. 1: (a) a Monitoring and Data Acquisition (MDA) component running on each IoMT device of the IoMT network, and (b) the Central Detection (CD) component (i.e., detection engine) running on the gateway.

III. Prototype Implementation of Components

In this section, we describe the details of the prototype implementation of the two main components (i.e., MDA, CD) of the proposed IoMT AIDS in [17]. It is worthwhile mentioning that the prototype implementation was developed for linux-based IoMT devices and gateways as linux-based OSes (e.g., ucLinux OS) are commonly used in IoT network deployments [18], [19].

A. MDA Component Implementation

The MDA component is deployed on each IoMT device connected to the gateway. The MDA component monitors the behavior of the IoMT device hosting it and collects relevant device behavior data (e.g., CPU usage, CPU processes, memory usage, disk usage) during a specific MDA period (i.e., sampling period). Moreover, the MDA component transmits the gathered data to the gateway as an MDA report. The MDA component was implemented using the C programming language. Fig. 2 depicts the run-time operation of the MDA component, consisting of the following steps:

Step 1: The configuration parameters are read from a configuration file.

Step 2: Memory resources are allocated and internal files are initialized.

Step 3: The communication of the MDA component to the CD component in the gateway is established based on two configuration parameters related to the IP address (i.e., "gwIPAddr" parameter) and the port (i.e., "gwPortNum" parameter) of the gateway.

Step 4: The process waits for "samplingPeriod" seconds. The "samplingPeriod" variable is a configuration parameter describing the time in seconds elapsing between consequent collections of behavior data by the MDA component.

Step 5: The "data_collect" function accesses the "/proc" directory that is present in a linux-based OS in order to gather IoMT device behavior data and to create a record comprising three distinct types of information. The first type is the timestamp of the OS when the data collection occurs. The second type relates to CPU and system statistics originating from the "/proc/stat" file. In particular, the "/proc/stat" file includes data regarding (a) the number of cycles that the CPU remained in different modes (e.g., "user" mode, "nice" mode, "system" mode, "idle" mode), (b) the total number of processes that were initiated since booting the OS, and (c) the number of currently running processes. The third type consists of internal memory statistics originating from the "/proc/meminfo" file. In particular, the "/proc/meminfo" file provides data related to (i) the total internal memory of the IoMT

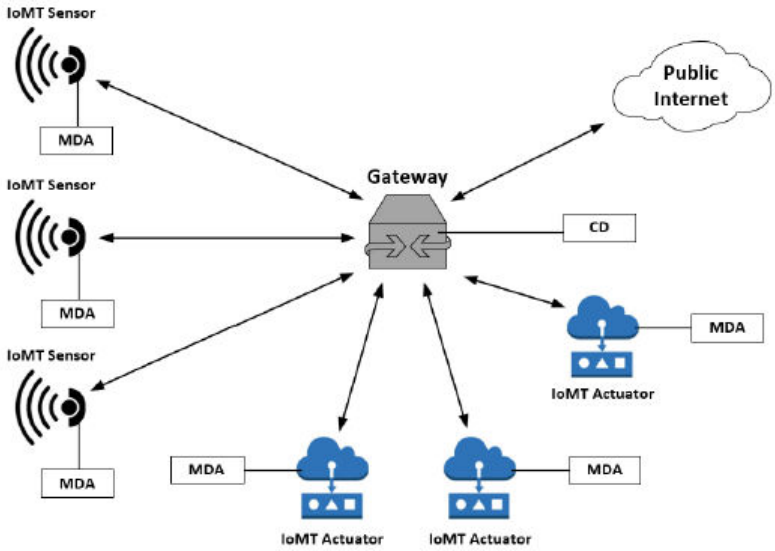

Fig. 1. The MDA and CD components of the proposed AIDS.

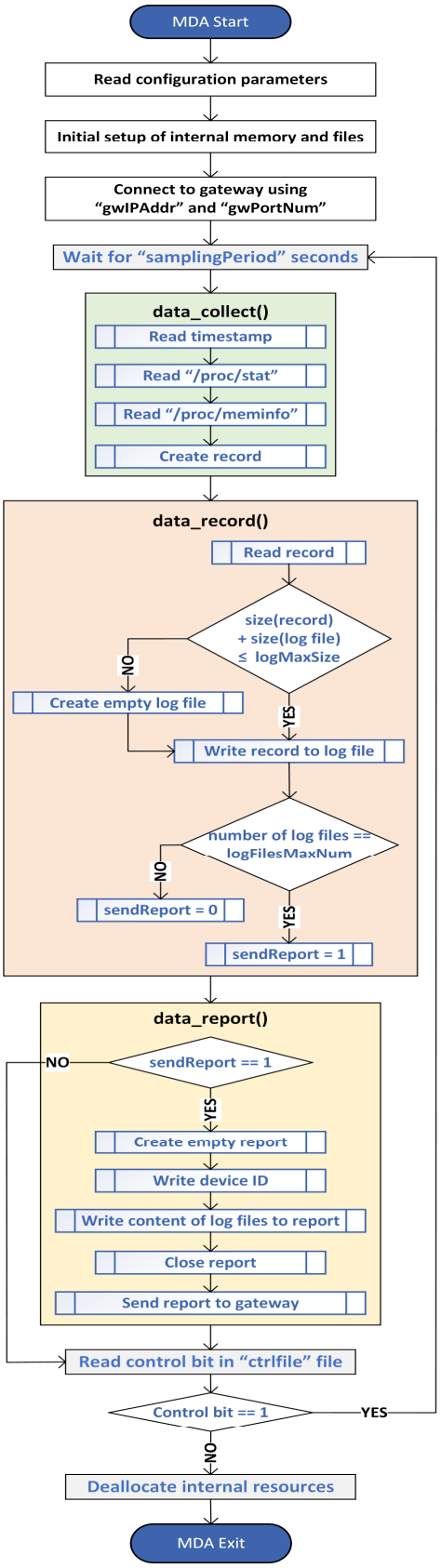

Fig. 2. Run-time Operation of the MDA Component.

device, (ii) the free internal memory of the IoMT device, (iii) the available internal memory of the IoMT device, and (iv) the cached internal memory of the IoMT device. A record in CSV format containing the three mentioned types of information is the output of the "data_collect" function.

Step 6: The "data_record" function receives the record from the "data_collect" function as input, and writes the record in a log file. In case that the size of the current log file exceeds the size permitted by the maximum size of the log file (i.e., defined by the "logMaxSize" parameter), then the "data_record" function closes and stores the current log file, and opens a new empty log file. In addition, apart from writing records to log files, the "data_record" function outputs a signal (i.e., "sendReport" signal) so that the "data_report" function knows whether a specific number (i.e., defined by the "logFilesMaxNum" parameter) of log files has been accumulated in order to create a report and send it to the gateway.

Step 7: The "data_report" function receives the "sendReport" signal, and based on this signal, the "data_report" function creates a report and writes the IoMT device ID in the report. Then the content of the accumulated log files is also included in the created report. Afterwards, the created report, as shown in Fig. 3, is sent to the gateway. In case that there is not a specific number of log files in order to create a report and send it to the gateway, then the execution continues with the next step.

Step 8: This step of the execution is meant to serve as a control step so that the operation of the MDA component can be terminated. In this step, a control file (i.e., "ctrlfile") is accessed and one bit is read from it. If the read bit is equal to 1, the execution continues from Step 4. Otherwise, the deallocation process is performed along with the termination of the operation of the MDA component. Thus, it is evident that the operation of the MDA component can be easily terminated by changing the control bit inside the control file.

```
1  0123456789
2  1653760850159,109562,738,20524,1456724,877,0,286,0,0,0,7339,1,2030968,141068,637232,619136
3  1653760910110,109762,738,20570,1462400,879,0,286,0,0,0,7373,4,2030968,134516,630756,621184
4  1653760911110,109788,738,20578,1462466,879,0,286,0,0,0,7377,2,2030968,130696,626940,624260
5  1653760912111,109816,738,20584,1462529,879,0,286,0,0,0,7385,1,2030968,120632,616876,633904
6  1653760913111,109818,738,20586,1462624,879,0,286,0,0,0,7394,1,2030968,120656,616900,633904
7  1653760914111,109821,738,20588,1462718,879,0,286,0,0,0,7406,1,2030968,120380,616624,633904
8  1653760915112,109823,738,20588,1462813,879,0,286,0,0,0,7415,1,2030968,120404,616648,633904
9  1653760916113,109825,738,20589,1462908,880,0,286,0,0,0,7415,2,2030968,122672,618932,633904
10 1653760917113,109827,738,20590,1463004,880,0,286,0,0,0,7415,1,2030968,137540,633800,619296
11 1653760918115,109828,738,20591,1463101,880,0,286,0,0,0,7415,1,2030968,137540,633800,619296
12 1653760919115,109830,738,20591,1463198,880,0,286,0,0,0,7415,1,2030968,137540,633800,619296
```

Fig. 3. Example of a report created by the MDA Component.

B. CD Component Implementation

The CD component runs on the gateway and its purpose is to: (i) monitor the behavior of the gateway hosting it and collect relevant behavior data (e.g., CPU usage, CPU processes, memory usage, disk usage) during a specific monitoring period (i.e., sampling period), (ii) monitor the network traffic passing through the gateway and gather

relevant network traffic data (e.g., source IP address or destination IP address or information about active connections) during a specific monitoring period, (iii) receive the reports transmitted by the MDA components running on the IoMT devices that are connected to the gateway, and (iv) leverage the aforementioned data to identify whether an malicious incident has occurred in the gateway, the IoMT devices or the IoMT network, and trigger a corresponding security alert. The CD component was implemented using the Java programming language. An execution example of the CD component is depicted in Fig. 4. In addition, Fig. 5 depicts the run-time operation of the CD component, consisting of the following steps:

Step 1: The configuration parameters are read.

Step 2: Memory resources are allocated and internal files and network sockets are initialized.

Step 3: The "gwBehavThread" thread is executed. The "gwBehavThread" thread is meant to handle the behaviour data (e.g., CPU usage, CPU processes, memory usage, disk usage) regarding the gateway device, starting from the collection of these data, continuing with the processing of these data and ending with performing intrusion detection based on these data.

Step 4: The "netThread" thread is executed. the "netThread" thread is meant to handle the network data (e.g., source IP address or destination IP address or information about active connections) related the network traffic passing through the gateway device, starting from the collection of these data, continuing with the processing of these data and ending with performing intrusion detection based on these data.

Step 5: The "iomtDevThread" thread is executed. The "iomtDevThread" thread is meant to: (i) perform the accepting of the connections of the MDA component deployed on the IoMT devices connected to the gateway, and (ii) create and execute separate threads for handling the information received by each of the connected MDA components.

Step 6: The process reads the input of the user from the keybord.

Step 7: If the input is "stop", then the execution continues with Step 8. Otherwise. The execution continues from Step 6.

Step 8: The "gwBehavThread" thread is stopped.

Step 9: The "netThread" thread is stopped.

Step 10: The "iomtDevThread" thread is stopped.

Step 11: The deallocation process is performed along with the termination of the operation of the CD component.

```
Program started!
Current directory: /home/osboxes/Documents/m4m_mi1
iomtDevThread: Started!
gwBehavThread: Started!
netThread: Started!
Connecting sensor with IP Addr (192.168.1.6) ...
Started receiving reports from iomtDevID: 0123456789
Gw Probability of Intrusion: 0.0
/* ------- Message from  iomtDevID:[0123456789] begins ------ */
1653760850159,109562,738,20524,1456724,877,0,286,0,0,0,7339,1,2030968,141068,637232,619136
1653760910110,109762,738,20570,1462400,879,0,286,0,0,0,7373,4,2030968,134516,630756,621184
1653760911110,109788,738,20578,1462466,879,0,286,0,0,0,7377,2,2030968,130696,626940,624260
1653760912111,109816,738,20584,1462529,879,0,286,0,0,0,7385,1,2030968,120632,616876,633904
1653760913111,109818,738,20586,1462624,879,0,286,0,0,0,7394,1,2030968,120656,616900,633904
1653760914111,109821,738,20588,1462718,879,0,286,0,0,0,7406,1,2030968,120380,616624,633904
1653760915112,109823,738,20588,1462813,879,0,286,0,0,0,7415,1,2030968,120404,616648,633904
1653760916113,109825,738,20589,1462908,880,0,286,0,0,0,7415,2,2030968,122672,618932,633904
1653760917113,109827,738,20590,1463004,880,0,286,0,0,0,7415,1,2030968,137540,633800,619296
1653760918115,109828,738,20591,1463101,880,0,286,0,0,0,7415,1,2030968,137540,633800,619296
1653760919115,109830,738,20591,1463198,880,0,286,0,0,0,7415,1,2030968,137540,633800,619296
/* ------- Message from  iomtDevID:[0123456789] ends ------ */
IoMT Device [0123456789] Probability of Intrusion: 0.0
Gw Probability of Intrusion: 0.0
Net Probability of Intrusion: 0.0
Gw Probability of Intrusion: 0.0
Gw Probability of Intrusion: 0.0
Exiting from netThread (netLoop())!
Exiting from gwBehavThread (gwBehavLoop())!
Exiting from iomtDevThread (hdlSensConnectLoop())!
Program finished!
```

Fig. 4. Execution example of the CD Component.

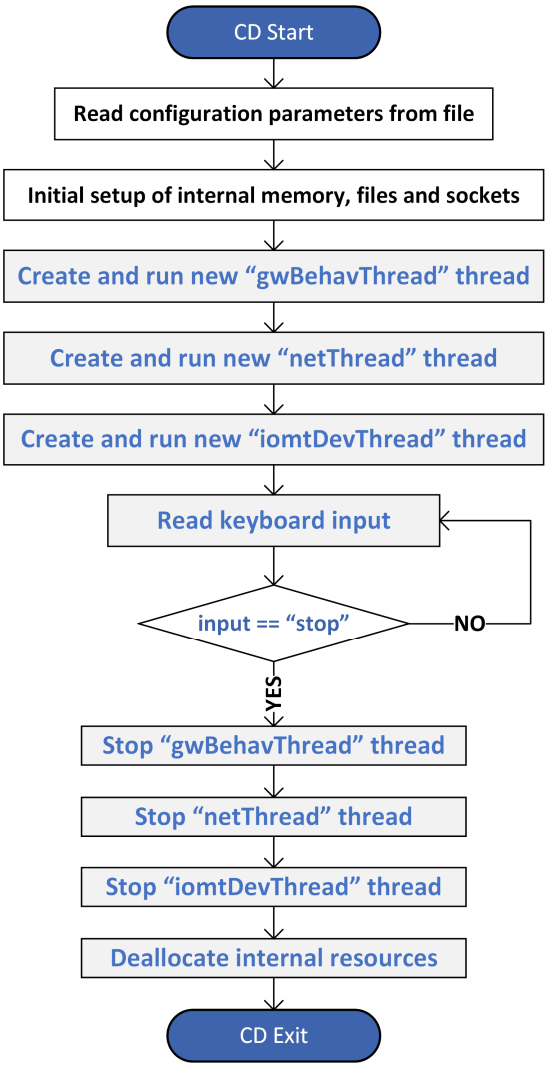

Fig. 5. Run-time Operation of the CD component.

IV. Conclusion

In this paper, we presented details of the implementation process that led to a prototype of the AIDS for IoMT networks that we had proposed in [17]. In particular, we described the run-time operations of the MDA components deployed and running on the IoMT devices connected to the gateway of the IoMT network and the CD component developed and running on the gateway device. Our target is this work to serve as a guidance for other researchers or engineers to develop their own specific implementations of AIDSs for IoMT networks.

As future work, we plan to measure the computation overhead of the presented prototype implementation on raspberry pi 4 Model B devices. Based on the computation overhead results, we will decide if further implementation optimizations are necessary. In addition, our aim is to improve and extend the implementation of the MDA component so that it can be used on devices without requiring a linux-based operating system (OS). Finally, we will focus our research efforts on selecting the optimum ML algorithms to be employed by the CD component for its intrusion detection purposes.

References

[1] J. J. P. C. Rodrigues et al., "Enabling Technologies for the Internet of Health Things," IEEE Access, vol. 6, pp. 13129–13141, 2018, doi: 10.1109/ACCESS.2017.2789329.

[2] M. Papaioannou et al., "A Survey on Security Threats and Countermeasures in Internet of Medical Things (IoMT)," Trans. Emerg. Telecommun. Technol., p. e4049, 2020, doi: 10.1002/ett.4049.

[3] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K. S. Kwak, "The internet of things for health care: A comprehensive survey," IEEE Access, vol. 3, pp. 678–708, 2015, doi: 10.1109/ACCESS.2015.2437951.

[4] E. Karavatselou, M.-A. Fengou, G. Mantas, and D. Lymberopoulos, "Profile Management System in Ubiquitous Healthcare Cloud Computing Environment," in Broadband Communications, Networks, and Systems, 2019, pp. 105–114.

[5] M.-A. Fengou, G. Mantas, D. Lymberopoulos, N. Komninos, S. Fengos, and N. Lazarou, "A New Framework Architecture for Next Generation e-Health Services," IEEE J. Biomed. Heal. Informatics, vol. 17, no. 1, pp. 9–18, 2013, doi: 10.1109/TITB.2012.2224876.

[6] F. Pelekoudas-Oikonomou et al., "Blockchain-Based Security Mechanisms for IoMT Edge Networks in IoMT-Based Healthcare Monitoring Systems," Sensors 2022, Vol. 22, Page 2449, vol. 22, no. 7, p. 2449, Mar. 2022, doi: 10.3390/S22072449.

[7] I. Makhdoom, M. Abolhasan, J. Lipman, R. P. Liu, and W. Ni, "Anatomy of Threats to the Internet of Things," IEEE Commun. Surv. Tutorials, vol. 21, no. 2, pp. 1636–1675, 2019, doi: 10.1109/COMST.2018.2874978.

[8] M. Zhang, A. Raghunathan, and N. K. Jha, "Trustworthiness of medical devices and body area networks," Proc. IEEE, vol. 102, no. 8, pp. 1174–1188, 2014, doi: 10.1109/JPROC.2014.2322103.

[9] M. Karageorgou, G. Mantas, I. Essop, J. Rodriguez, and D. Lymberopoulos, "Cybersecurity attacks on medical IoT devices for smart city healthcare services," in IoT Technologies in Smart Cities: From sensors to big data, security and trust, Institution of Engineering and Technology, 2020, pp. 171–187.

[10] I. Essop, J. C. Ribeiro, M. Papaioannou, G. Zachos, G. Mantas, and J. Rodriguez, "Generating datasets for anomaly-based intrusion detection systems in iot and industrial iot networks," Sensors, vol. 21, no. 4, pp. 1–31, 2021, doi: 10.3390/s21041528.

[11] F. Alsubaei, A. Abuhussein, and S. Shiva, "Security and Privacy in the Internet of Medical Things: Taxonomy and Risk Assessment," in Proceedings - 2017 IEEE 42nd Conference on Local Computer Networks Workshops, LCN Workshops 2017, 2017, pp. 112–120, doi: 10.1109/LCN.Workshops.2017.72.

[12] J. Asharf, N. Moustafa, H. Khurshid, E. Debie, W. Haider, and A. Wahab, "A review of intrusion detection systems using machine and deep learning in internet of things: Challenges, solutions and future directions," Electronics (Switzerland), vol. 9, no. 7. MDPI AG, p. 1177, 2020, doi: 10.3390/electronics9071177.

[13] J. Ribeiro, F. B. Saghezchi, G. Mantas, J. Rodriguez, and R. A. Abd-Alhameed, "HIDROID: Prototyping a Behavioral Host-Based Intrusion Detection and Prevention System for Android," IEEE Access, vol. 8, pp. 23154–23168, 2020, doi: 10.1109/ACCESS.2020.2969626.

[14] J. Ribeiro, F. B. Saghezchi, G. Mantas, J. Rodriguez, S. J. Shepherd, and R. A. Abd-Alhameed, "Towards an Autonomous Host-based Intrusion Detection System for Android Mobile Devices," in 9th EAI International Conference on Broadband Communications, Networks, and Systems (BROADNETS2018), 2018, pp. 139–148.

[15] J. Ribeiro, F. B. Saghezchi, G. Mantas, J. Rodriguez, S. J. Shepherd, and R. A. Abd-Alhameed, "An Autonomous Host-Based Intrusion Detection System for Android Mobile Devices," Mob. Networks Appl., vol. 25, no. 1, pp. 164–172, 2020, doi: 10.1007/s11036-019-01220-y.

[16] G. Thamilarasu, A. Odesile, and A. Hoang, "An intrusion detection system for internet of medical things," IEEE Access, vol. 8, pp. 181560–181576, 2020, doi: 10.1109/ACCESS.2020.3026260.

[17] G. Zachos, I. Essop, G. Mantas, K. Porfyrakis, J. C. Ribeiro, and J. Rodriguez, "An Anomaly-Based Intrusion Detection System for Internet of Medical Things Networks," Electron. 2021, Vol. 10, Page 2562, vol. 10, no. 21, p. 2562, Oct. 2021, doi: 10.3390/ELECTRONICS10212562.

[18] P. Gaur and M. P. Tahiliani, "Operating systems for IoT devices: A critical survey," in Proceedings - 2015 IEEE Region 10 Symposium, TENSYMP 2015, Jul. 2015, pp. 33–36, doi: 10.1109/TENSYMP.2015.17.

[19] M. H. Qutqut, A. Al-Sakran, F. Almasalha, and H. S. Hassanein, "Comprehensive survey of the IoT open-source OSs," IET Wirel. Sens. Syst., vol. 8, no. 6, pp. 323–339, Dec. 2018, doi: 10.1049/IET-WSS.2018.5033.