IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 3, 2021, accepted May 24, 2021, date of publication June 11, 2021, date of current version July 28, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3088395

A Novel Approach for Detecting Cyberattacks in Embedded Systems Based on Anomalous Patterns of Resource Utilization-Part I

ABDULMOHSAN ALOSEEL[1], SABA AL-RUBAYE[1], (Senior Member, IEEE), ARGYRIOS ZOLOTAS[1], (Senior Member, IEEE), HONGMEI HE[2], (Senior Member, IEEE), AND CARL SHAW[3]

[1]School of Aerospace, Transport and Manufacturing (SATM), Cranfield University, Cranfield MK43 0AL, U.K.
[2]School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, U.K.
[3]Cerberus Security Laboratories Ltd., Bristol BS34 8RB, U.K.

Corresponding author: Abdulmohsan Aloseel (abdulmohsan.aloseel@cranfield.ac.uk)

This work was supported by the Centre of Autonomous and Cyber Physical Systems, School of Aerospace, Transport and Manufacturing (SATM), Cranfield University, Bedford, U.K.

ABSTRACT This paper presents a novel security approach called Anomalous Resource Consumption Detection (ARCD), which acts as an additional layer of protection to detect cyberattacks in embedded systems (ESs). The ARCD approach is based on the differentiation between the predefined standard resource consumption pattern and the anomalous consumption pattern of system resource utilization. The effectiveness of the proposed approach is tested in a rigorous manner by simulating four types of cyberattacks: a denial-of-service attack, a brute-force attack, a remote code execution attack, and a man-in-the-middle attack, which are executed on a Smart PiCar (used as the testbed). A septenary tuple model consisting of seven parameters, representing the embedded system's architecture, has been created as the core of the detection mechanism. The approach's efficiency and effectiveness has been validated in terms of range and pattern by analyzing the collected data statistically in terms of mean, median, mode, standard deviation, range, minimum, and maximum values. The results demonstrated the potential for defining a standard pattern of resource utilization and performance of the embedded system due to a significant similarity of the parameters' values at normal states. In contrast, the attacked cases showed a definite, observable, and detectable impact on resource consumption and performance of the embedded system, causing an anomalous pattern. Thus, by merging these two findings, the ARCD approach has been developed. ARCD facilitates building secure operating systems in line with the ES's capabilities. Furthermore, the ARCD approach can work along with existing countermeasures to augment the security of the operating system layer.

INDEX TERMS Anomalous resource consumption, brute-force attack, cyberattacks, denial-of-service attack, embedded systems, password attack, remote code execution, testbed.

I. INTRODUCTION

In light of the enormous expansion and spread of embedded systems (ESs) applications in different application domains such as communication, transportation, education, defense, energy, and medicine, ESs form critical components in cyber-physical systems (CPS) and IoT-enabled systems. Industry 4.0, the industrial internet of things (IIoT), integrates all systems into the Internet, such as supervisory control and data acquisition systems (SCADA), industrial automation and control systems (IACS), industrial control systems (ICS), and distributed control systems (DCS). Hence, cybersecurity by design is a critical challenge for the success of these modern applications. This highlights the urgent need to secure these systems against cyberattacks, which can cause catastrophic consequences to both humans and the economy. There are many mature techniques for protecting IT infrastructure. However, compared to conventional multitasking systems, ESs face significant challenges to adopt existing advanced security solutions due to their capabilities and limiting characteristics, such as small size, small volume of memory, less computational capabilities, low energy consumption,

The associate editor coordinating the review of this manuscript and approving it for publication was Xiali Hei.

mobility, independence, and low cost. It is necessary to have effective security solutions that maintain security goals and match the ES's capabilities and characteristics.

Marwedel [1] discusses many tight constraints on ES resources, particularly in the case of a system on a chip (SoC), and how these limitations hinder the ability to implement advanced security solutions. These constrained capabilities also make it impossible to apply conventional security mechanisms [2]. ESs must be resource-aware, and cannot have excessive pipeline performance due to limited resources [3]. The gaps in processing, battery, and cost are at the top of the security challenges list. The high barrier of constraints on computational capability, storage, power, and real-time determinism makes it difficult to maintain data confidentiality through encryption and key management [4]. The addition of many software-based security mechanisms on top of the OS layer would compete for computational resources, introducing an incompatibility with ES capabilities [4]. Developed security solutions must consider the capabilities of software and hardware during the earliest stages of design, hence mitigating the additional cost for cybersecurity at the later stages of the system lifecycle.

The execution of advanced cryptography and other advanced security solutions and countermeasures, such as an intrusion detection system (IDS), require intensive computing resources and inevitably consume more energy [5]. ESs are unable to handle intensive cryptographic calculation-based encryption and decryption operations due to SoC resource limitations [3]. The existing cryptographic algorithms require sufficient processing, memory, and energy capabilities, which may be unavailable in limited resource SoCs [6]. Therefore, running a robust encryption scheme on an ES with its limited resources is a challenge [7], [8]. Conversely, the nature of an ES poses significant tight constraints and their impact reflects on performance capabilities [9], [10].

Hameed *et al.* [11] addressed the issue of distributed denial-of-service (DDoS) attack detection and stated, ''Detection of DDoS in IoT is a challenging issue as IoT network and traffic characteristics are quite different from the traditional network. Due to the limitation of IoT devices, resource-efficient DDoS detection and countermeasure techniques are required.'' Neglecting security countermeasures at the design stage could provide more chances for attackers to destroy these systems successfully. As many ESs are powered by battery and security enhancement could inevitably sacrifice part of the energy, it is crucial to reduce the amount of power consumption with lightweight and efficient solutions [5]. As Ali *et al.* [12] stated, the conflicting requirements for higher computational capabilities but lower power consumption pose serious challenges to implement effective and efficient solutions for the security of ESs. It is not feasible to implement conventional IT security solutions in ESs in terms of the ES's capabilities.

In general, embedded systems are exposed to the same security threats as conventional computer systems, but with significant limitations to handle the advanced security solutions that have been implemented on these conventional computers and servers. Therefore, there is an urgent need for lightweight security strategies to enhance system security under these constraints [5], [11] based on existing hardware and software systems and without excessive computational requirements or additional cost for hardware components.

Based on the relevant studies in this field, we have published a comprehensive analytical study that covers all aspects of the cybersecurity of embedded systems [13] and addressed some of the existing solutions. In this study, we are presenting an alternative security solution, where the proposed approach utilizes resource consumption patterns to recognize the existence of a cyberattack. In this paper, we aim to deliver a novel security approach to detecting and dealing with cyberattacks without conflicting with the ES's characteristics and without depleting its resources, through monitoring system resource consumption and detecting anomalous performance patterns. The proposed approach is referred to as Anomalous Resource Consumption Detection (ARCD). Anomaly detection as a concept has been previously implemented in applications such as bank fraud and intrusion detection systems (IDS) [14]. It is worth noting, though, that IDS applications concentrated on the network traffic by inspecting every single packet and looking for any malicious activities or unexpected ongoing data traffic using data analysis methods or machine learning algorithms [15], [16]. This is quite different to the concept proposed in our study that focuses particularly on ESs, benefiting from the ES's characteristics (being dedicated to performing specific functions).

The remainder of this paper is organized as follows. In Section II, the hypothesis of this study was presented, and the concept of the septenary tuple model was explained. In the third section, the hardware and software for the experimental environment are illustrated. The adopted methodology has been described in Section IV. The final results and analysis of the typical cases have been summarized in Section V, followed by the attacked cases results in Section VI. The final statistical results for all cases have been explained and discussed in Section VII. Finally, the conclusion and future work are presented in Section VIII.

II. THE HYPOTHESIS AND TESTBED CONFIGURATION

A. THE HYPOTHESIS

The proposed security method is based on the hypothesis that a unique characteristic of ESs distinguishes them from conventional multitasking systems, in which these systems are designed to perform specific functions. Thus, the hypothesis states: In embedded systems designed to perform specific functions, the performance and resource consumption form a narrowly bounded, determinable pattern. Thus, an anomalous pattern, if it exists and is identifiable, can be exploited as an indicator to detect potential cyberattacks.

The anomaly detection concept is a wide concept on its own, and it can be implemented in conjunction with uncounted technologies, and many related studies are

pointing in the same direction as the approach that we seek to present. However, what distinguishes our proposed approach is its comprehensive detection mechanism, which is based on the septenary tuple model, as well as its generality through focusing on the embedded system abstractly rather than its implementations.

The hypothesis comprises two main parts. First, it requires that when an ES is performing its specific functions, we can identify the range and the pattern of the performance and resource consumption under normal operation conditions and in the absence of cyberattacks. The second part implies that an anomalous resource consumption pattern can be inferred as an indicator of a cyberattack occurrence. To validate the hypothesis, we need to prove the feasibility of creating a standard pattern for the system performance and resource consumption under normal operating conditions (with no cyberattacks present) to be used as the control pattern. Also, in an attack case, we need to prove the existence of anomalous performance and resource consumption and their detectability, contrary to the control case. The anomalous pattern should be recognizable and distinguishable from standard resource consumption. To prove the validity of the hypothesis, we conduct an experiment to observe specific parameters' values, which represent resource consumption and the ES's performance under different circumstances.

Based on the collected data, we will be able to confirm the validity of the first part of the hypothesis, which is essential. If the range and pattern of resource consumption under normal operating conditions is highly variable and undeterminable, it would prove difficult to distinguish anomalous patterns from normal patterns. Hence, the second part of the assumption would not be proved without proof of the first part. After proving the first part of the assumption, we will move to the second part of the research hypothesis and execute various scenarios of cyberattacks to see if it is possible to detect the existence of attacks based on the anomalous change on system performance and resource consumption.

B. THE TESTBED

Testbeds "are composite abstractions of systems and are used to study system components and interactions to gain further insight into the essence of the real system" [17]. According to Edgar and Manz [18], a testbed is defined as "a controllable cyber environment that enables experimentation." In our study, a Smart PiCar will be used as a testbed. The ES within this CPS will be monitored to study particular aspects of system performance and its resource consumption under a controlled environment.

The testbed has specific attributes: (1) purpose, (2) resources, (3) resource management, (4) configuration/ initialization, (5) experimental design, (6) experimental control, (7) instrumentation and data collection, and (8) access [18]. During the testbed design stages, these attributes have been taken into account.

Three typical design options are normally adopted for experimental design [19]: (1) simple design, (2) fractional factorial design, and (3) full factorial design. Simple design refers to adopting a fixed configuration and changing one factor at a time to determine the impacts of this factor on system performance. Observing impacts when modifying only a few factors against others reflects the concept of fractional factorial design, whereas altering all factors against each other is an example of the full factorial experimental design concept. The fractional factorial or full factorial design options are the most suitable to study how factors interact with each other.

In our experimental design, we first adopt the simple experimental design approach and alter one factor at a time to determine the impact on system performance. The selected factor is the existence or absence of a cyberattack. This is followed by a parameter selection phase to design a septenary tuple model. In this stage, we adopt a full factorial design to find the most influential parameters to build the detection mechanism's core.

The Smart PiCar provides a realistic hardware-software environment to test the embedded system components under different operational circumstances. As a result of the testbed, we will have quantitative results about the operational behavior of the ES for specific parameters. Thus, our case study's design decision has been taken based on both theoretical and empirical studies.

1) THE PARAMETERS OF A TESTBED

To validate the proposed security approach, it is necessary to monitor, collect and analyze values of specific parameters on the testbed, representing the performance of a real CPS, to be able to make a comparison between unattacked cases and attacked cases. Determining specific parameters to be investigated in the experiments is not arbitrary, but is based on the architecture of an ES. This consists of a central processing unit (CPU) to manage the data and execute the instructions, volatile memory or random-access memory (RAM), read-only memory or non-volatile memory (ROM), as well as the system bus, communication subsystem, power subsystem (battery), and input/output units. Based on these components, seven important parameters have been identified as follows:

1- CPU utilization: the CPU utilization percentage parameter represents the percentage of the total available CPU computing power available that is being used to perform the function of the ES. A Broadcom BCM2711, quad-core ARM Cortex-A72 (SoC - ARMv8) 64-bit is working as the compute unit of the CPS.
2- Memory load: the memory (Mem) usage percentage parameter indicates the amount of memory resource used. The Smart PiCar is equipped with 4GB LPDDR4-3200 SDRAM to support the processes of the ARM processor. Specifically, we monitored the active system virtual memory usage percentage. (see *psutil commands* documentation [20]).
3- Task count: this parameter reflects the total number of current CPU software processes. These processes are the programs currently under execution in the processor or

still alive, and which have not been fully completed by the system or user.

4- Thread count: a single program can contain multiple threads. Threads provide a mechanism for coordinating parallel instruction flows within a process. However, "Linux has a unique implementation of threads" [21]: threads and processes are executed in an equivalent manner by the operating system kernel, so a multi-threaded software application appears in a task view as multiple related processes, and thread count has been extracted based on the system command syntax according to the manual pages 22]. Thus, all active threads have been recorded.

5- CPU temperature: the CPU temperature (in celsius) of the processor is an important factor affecting the performance of the system. If the temperature rises without a justifiable reason, that might be an indicator of unexpected activity. "Thermal throttling" of the BCM2711 has been taken into consideration based on documentation from the official website of the Raspberry Pi Foundation [23].

6- Power consumption: the power consumption rate is the work done per unit of time and is equivalent to the voltage (V) multiplied by the current, which is measured in amperes or "amps" (A). Therefore, the measurements of voltage and current have been monitored.

7- Received (RX) and Transmitted (TX) packets rate: the data traffic whether received or transmitted are meaningful and an important indicator of the system's connection stability, and proves that the system is immune to any suspicious activities. The Raspberry Pi is equipped with 2.4 GHz and 5.0 GHz IEEE 802.11ac Wi-Fi and 2.4 GHz BLE Bluetooth 5.0. Only traffic travelling over the 802.11 Wi-Fi interface was measured.

2) SYMBOLIC MODEL FOR THE TESTBED

The above-mentioned parameters represent the hardware/software components of the ES that interact instantly with performed functions of the system. Thus, a dynamic reflection of the system's health can be observed by monitoring these parameters' changes. The CPU works as the brain of the system and each process running in the CPU will directly reflect on the memory consumption. The temperature of the ARM processer and its changes will reflect the load of the CPU. In addition, to attest to system stability and reliability, and to ensure that the system performance is accurately measured, the number of alive processes and executed threads have been included among the testbed's parameters. Additionally, energy consumption rate has been included as a parameter because it is one of the most important resources of an ES. Its stability or fluctuation is expected to give a meaningful indication of activity during the experiment. A parameter model reflecting the Smart PiCar ES state is therefore represented by a septenary tuple, $\{U, M, N, \vartheta, T, P, R\}$, shaping the ranges and patterns of performance and resource consumption. It is as described as below:

U: CPU utilization, $U = \{u_r, u_p\}$, where u_r is the range of the CPU utilization and u_p is the pattern of CPU utilization.

M: Memory utilization, $M = \{m_r, m_p\}$, where m_r is the range of memory utilization and u_p is the pattern of memory utilization.

N: Task count, $N = \{n_r, n_p\}$, where n_r is the range of active tasks and n_p is the execution pattern of tasks.

ϑ: Thread count, $\vartheta = \{\vartheta_r, \vartheta_p\}$, where ϑ_r is the range of executed threads and ϑ_p is the execution pattern of the threads.

T: CPU temperature, $T = \{t_r, t_p\}$, where t_r is the range of CPU temperature and t_p is the thermal pattern.

P: Power consumption, $P = \{V, I\}$, where V: voltage, $V = \{V_r, V_p\}$, V_r is the range of the voltage, and V_p is its pattern; I: current, $I = \{I_r, I_p\}$, where I_r is the range of current volt and I_p is its pattern.

R: Rate of RX and TX packets, $R = \{RX_r, RX_p\}$, where RX_r is the range of received packets and RX_p is its pattern. Similarly, in TX, TX_r is the range and TX_p the pattern of transmitted packets.

The following table summarizes these parameters in terms of associated resource or performance measurement, along with the symbol used to represent them.

TABLE 1. Parameters of the testbed.

	Parameters	Abbr.	Executed commands / Tools
Consumption Resources	CPU	U	`psutil.cpu_percent()`
	Memory	M	`psutil.virtual_memory()`
	Temp	T	`/sys/class/thermal/thermal_zone0/temp`
	Power	P	`UM25C USB power meter, for (Voltage, Current)`
Performance	Tasks	N	`'sudo ps aux | wc -l'`
	Threads	ϑ	`'sudo ps -eLf | wc -l'`
	Rates of Received & Transmitted Packets	R	`int(open('/sys/class/net/wlan0/statistics/tx_packets').read()); (and for transmitted packets (rx) instead of tx)`

The U, M, T, and P parameters represent resource consumption, while the N, ϑ, and R parameters represent aspects of system performance. All the values of these parameters in both resource consumption and performance will be

TABLE 2. Statistical analysis criteria.

	Criteria	Abbr.	Value
Range	Minimum	X_{minV}	The lowest value.
	Maximum	X_{maxV}	The highest value.
	Range	X_{gV}	The difference between the lowest and highest.
Pattern	Mean	X_{avgV}	The average.
	Mode	X_{modV}	The highest frequency value (density point).
	Median	X_{midV}	The middle number in a sorted list of values.
	Standard deviation	X_{stdV}	A measure of how spread out numbers are.

analyzed based on seven statistical criteria that define their ranges and patterns. For the ranges, the minimal (X_{minV}), maximal (X_{maxV}), and range (X_{gV}) values will be calculated to determine the range, where X refers to a parameter. To determine the pattern, the following values will be calculated: the median (X_{midV}), mean (X_{avgV}), mode (X_{modV}), and standard deviation (X_{stdV}). Table 2 summarizes these criteria.

III. HARDWARE AND SOFTWARE OF THE TESTBED

A Smart PiCar (equipped with a Pi camera) has been designed for reconnaissance purposes and is used as the testbed of the validation experiments. A Raspberry Pi 4 Model B is integrated as a processing unit (the ES) to handle the computational processes and steer the actions of the PiCar. The Smart PiCar consists of the following components: ultrasonic obstacle avoidance module, robot hardware attached on tops (HATs) Raspberry Pi module, and a motor driver module. The bus PWM driver supports independent output power and uses four wires to control the back wheels. The system was configured with DEBIAN RASPBIAN GNU/Linux version 10.2, and to complete the programmatic settings of the Smart PiCar the system was configured according to the documentation provided by SunFounder Inc. In addition, the Secure Shell protocol (SSH) and the Pi camera were enabled. Figure 1 depicts the hardware configuration of the assembled Smart PiCar testbed.

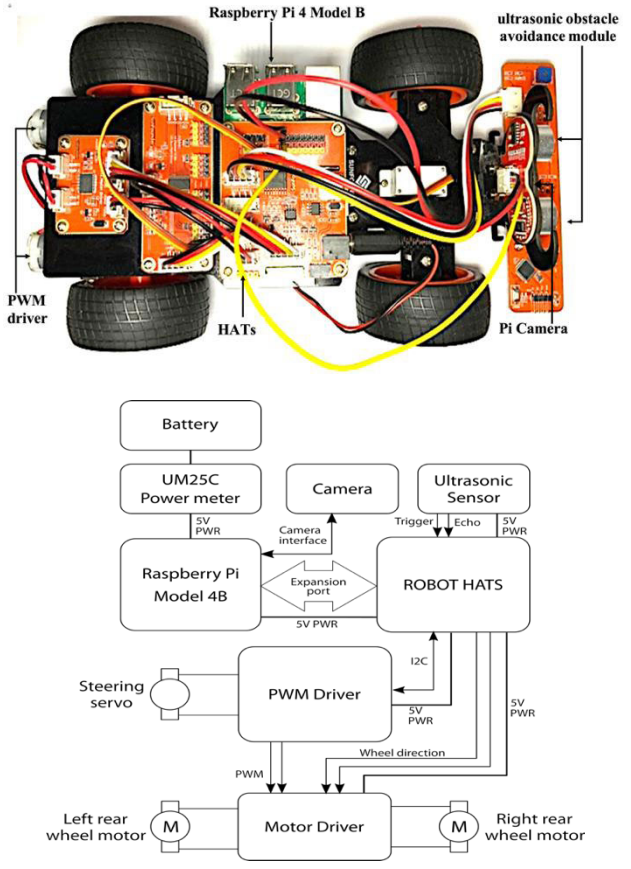


FIGURE 1. The testbed: A smart PiCar with its schematic diagram.

IV. METHODOLOGY

To prove the effectiveness of the proposed approach, we conducted a series of scientific experiments using the testbed. This series of experiments was performed on a healthy system and under attack conditions by exposing the same system to four different types of cyberattacks to validate the accuracy of the hypothesis, which states there exist discernable patterns in system resource consumption.

The methodology of this study consists of three main phases: (1) initializing the experiment environment, (2) conducting the experiments, and (3) using an analysis method for validation (Figure 2).

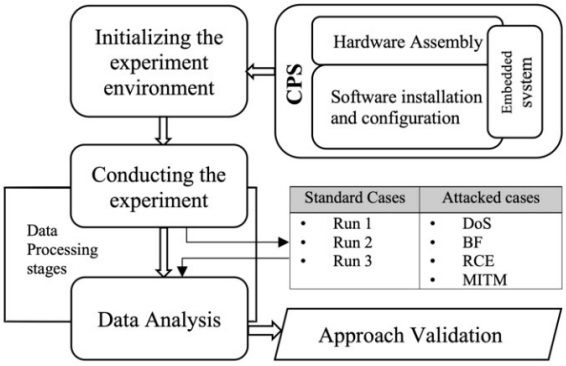


FIGURE 2. The experiment landscape on the testbed.

A. CONFIGURATION OF THE EXPERIMENT ENVIRONMENT

Phase I is to create a suitable testbed environment and its procedure should be applicable across different platforms. To extract the values of specific parameters from Raspberry Pi, specific Linux commands are inserted in compatible Python syntax. Based on the official Linux and Python references [22], [24], Table 1 shows the executed commands to extract the required parameters' values and the tool that has been used to measure the power consumption. A UM25C USB power meter tester was attached to the Smart PiCar to take voltage and current measurements and was connected to a portable power source to supply the Smart PiCar with the needed energy.

In Phase II, two cases are studied:

Normal case: The Smart PiCar is run repeatedly without any cyberattacks to generate three rounds of data reference sets (Rs): R1 first round, R2 second round, and R3 third round. The values of the selected parameters will be collected and analyzed according to the statistical criteria. Based on the values analysis, a standard model (*Std)* will be generated, where the range is Std_r and pattern is Std_p.

Attacked case: The Smart PiCar is run repeatedly four times but this time under different types of cyberattacks:

(1) Denial-of-Service (DoS), (2) password attack (Brute-Force (BF)), (3) Remote Code Execution (RCE), and (4) Man-in-the-Middle attack (MITM). The landscape of the experiments is shown in Figure 2.

To ensure the data collection's accuracy and integrity, the system was rebooted before each experiment and the following steps were taken:

- All the physical components and connecting cables were checked visually.
- The USB meter tester was activated to monitor the energy consumption via Bluetooth.
- The electrical power source was connected.
- The Raspberry Pi and Smart PiCar were switched on.
- The IP address of the test device was determined by using the following command: $ arp -a
- After determining the IP address, the Smart PiCar was remotely accessed.
- The accuracy of the system time and time zone was checked in order to ensure an precise time stamp during the data collection period.
- The recording function for the built-in camera was activated in the Smart PiCar by executing the following command: `pi@raspberrypi: ~ $ raspivid -o video.h264 -t 360000`
- The ultrasonic obstacle avoidance module was activated according to the instructions provided by the manufacturer.
- The Python program that had been written for the purpose of extracting and saving the values of the testbed's parameters was started.
- Finally, the ambient and initial temperatures of the system were recorded.

After completing all these steps and after the system had completed an entire cycle of performing its functions, parameter values representing (336 seconds) of operation were extracted and stored as records. These steps have been repeated for each round. At the end of the video recording period, the system stops its recording function automatically. After that, the following steps were followed to collect and save the data in preparation for the analysis phase: the log file and the recorded video were renamed with unique names to avoid being overwritten by the next run; the produced files for R1, R2, and R3 were relabeled; and the system was rebooted in preparation for the next round. The steps were then repeated.

B. EXPERIMENT CONDUCTION AND ANALYSIS METHOD

After completing the data collection of the recurring experiments, the third phase of data analysis was started, aiming to generate a reliable standard model for measurement and comparison. This should allow the validity of the first part of the hypothesis to be tested in terms of the determinability and definability of the standard range and pattern of regular consumption. Then, moving to the second part of the hypothesis, the same steps will be repeated but under different cyberattack scenarios that are expected to have an anomalous impact. The execution of attack scenarios will be explained in the upcoming sections. The third phase consists of six stages, starting with data generation and ending with approach validation. Figure 3 summarizes these six stages.

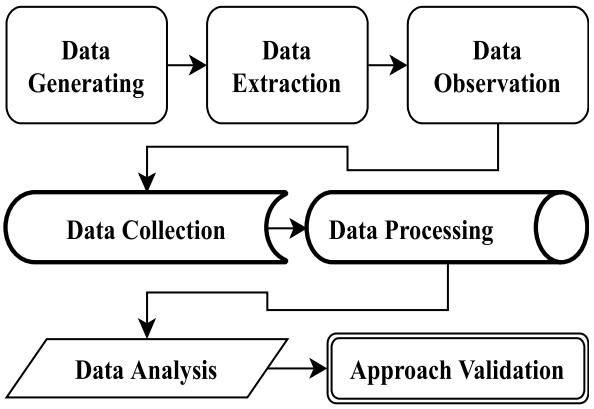


FIGURE 3. Data processing stages.

To generate the data, we will operate the Smart PiCar remotely through the terminal using the SSH protocol. The targeted parameters were linked to an IoT analytics platform (ThingSpeak) to visualise and analyse the data when needed as an option. To collect and store the data in a log file, the following command was included in the Python code: `'sys.stdout = open("log.csv","a+")'`. Log entries were written to this file following a comma-separated values (CSV) file convention within the Python code. During the data processing stage, the timestamp of each entry was checked and any corrupted or duplicated data was excluded.

By comparing the values of the selected parameters in terms of the statistical criteria, the range and pattern of the system performance and resource consumption, whether under normal operating conditions or under attack impacts, could be defined. For the first part of the experiment, if the results of these criteria show similarity, consistency, identicality, or conformity in terms of the parameters' values, the validity of the first part of the hypothesis will be proven. A standard model, based on typical operating cases, will be generated by filling the defined statistical criteria according to the septenary tuple model after the analysis stage. For the second part of the experiment, if the parameters' values of these criteria under the cyberattack circumstances diverge, deviate, or differ from standard cases in terms of the range and patterns, then the mechanism of the approach will be an efficient security solution as long as this anomalous consumption can be monitored and is detectable and distinguishable.

V. RESULTS AND ANALYSIS OF NORMAL CASES

A. CPU UTILIZATION

In this section, the acquired results from the testbed were analyzed and the analytical results of the collected data of the embedded system's functions under normal operating conditions showed great similarity in terms of U, where the

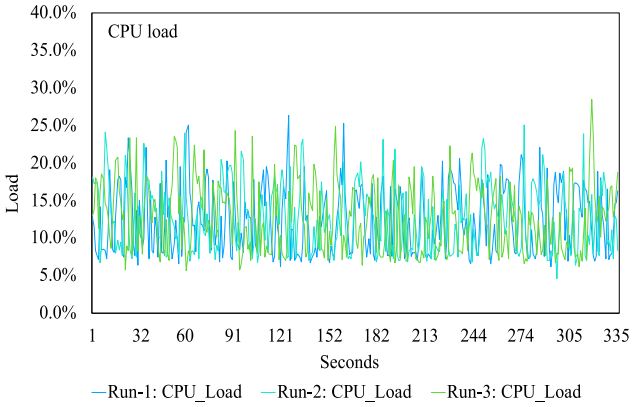


FIGURE 4. CPU load patterns for R1, R2, and R3.

TABLE 3. CPU utilization under normal operating conditions.

	CPU (U)	Abb.	R1	R2	R3
Pattern	Mean	U_{avgV}	12.39	12.68	12.74
	Mode	U_{modV}	7.90	7.70	7.50
	Median	U_{midV}	11.80	12.40	12.50
	Std. Deviation	U_{stdV}	4.40	4.35	4.53
Range	Range	U_{gV}	20.20	20.50	22.80
	Minimum	U_{minV}	6.20	4.60	5.70
	Maximum	U_{maxV}	26.40	25.10	28.50

U_{avgV} ranged between 12.39%, 12.68%, and 12.74% for R1, R2, and R3, respectively. Also, the U_{modV} was concentrated between 7.90%, 7.70%, and 7.50%. The range (U_{gV}) of the distribution of the values was between 20.20%, 20.50%, and 22.80%. Table 3 summarizes these criteria values, and Figure 4 demonstrates the CPU utilization patterns, showing their great similarity among R1, R2, and R3.

B. MEMORY LOAD

In terms of memory usage, not only were similarities observed, but an identical memory performance was recorded in terms of memory mode (M_{modV}), where the memory usage percentage (Mode) was concentrated at 9.50% in R1, R2, and R3. Likewise, the M_{midV} value was 9.50 in all cases and the minimum value was at 9.30% in R1, R2, and R3. The range values ranged between 1.40 and 1.50, which showed a stable pattern in memory load with only a 0.10 difference. Also, the standard deviation values were close in R1, R2, and R3. Table 4 summarizes these criteria values, and Figures 5 and 6 show the transcendent conformity in terms of memory usage rate each time the system's functions have been activated under regular operation conditions.

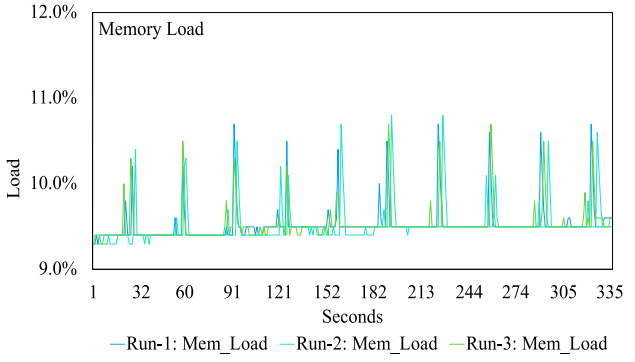


FIGURE 5. Memory load for R1, R2, and R3.

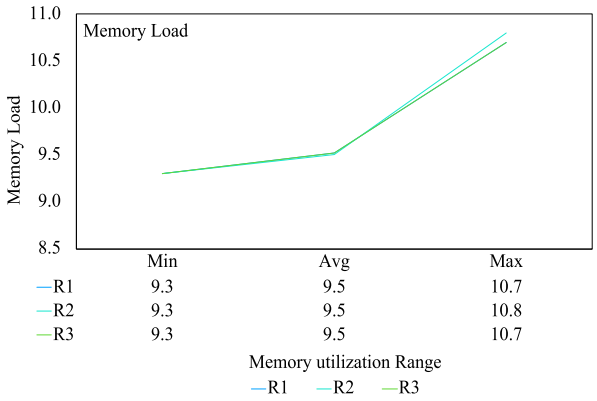


FIGURE 6. Memory load for R1, R2, and R3.

TABLE 4. Memory performance under standard operating conditions.

	Memory (M)	Abb.	R1	R2	R3
Pattern	Mean	M_{avgV}	9.52	9.51	9.52
	Mode	M_{modV}	9.50	9.50	9.50
	Median	M_{midV}	9.50	9.50	9.50
	Std. Deviation	M_{stdV}	0.21	0.23	0.20
Range	Range	M_{gV}	1.40	1.50	1.40
	Minimum	M_{minV}	9.30	9.30	9.30
	Maximum	M_{maxV}	10.70	10.80	10.70

C. TASKS AND THREADS

In terms of the number of processes and threads, the statistical criteria results showed a remarkable convergence, as can be seen in Tables 5 and 6. The average values (N_{avgV}) of the three rounds in terms of counting alive tasks were 167.9, 168.8, and 168.1 for R1, R2, and R3, respectively. Also, the average values of executed threads were 228, 228.9, and 228.3, respectively, for R1, R2, and R3. Also, one of the most important aspects to determine the pattern is the mode (N_{modV} and ϑ_{modV}) value, which has been calculated to determine the performance pattern for R1, R2, and R3. In terms of tasks count mode, the N_{modV} values were identical in all Rs cases (172 alive tasks). Also in terms of threads, the ϑ values were identical (232 threads). Likewise, in terms of the N_{maxV} for R1, R2, and R3, the values were identical (173 tasks). In terms of threads number, the range (ϑ_{gV}) between the highest and lowest value also was identical between R1 and R3. Therefore, the ability to define the range and pattern of the typical consumption for the system designed to perform specific functions can be seen clearly in these testbed results.

TABLE 5. Values of tasks number under standard operating conditions.

	Tasks No.	Abb.	R1	R2	R3
Pattern	Mean	N_{avgV}	167.9	168.8	168.1
	Mode	N_{modV}	172.0	172.0	172.0
	Median	N_{midV}	172.0	171.0	172.0
	Std. Deviation	N_{stdV}	4.5	3.6	4.2
Range	Range	N_{gV}	14.0	11.0	13.0
	Minimum	N_{minV}	159.0	162.0	160.0
	Maximum	N_{maxV}	173.0	173.0	173.0

TABLE 6. Values of threads number under standard operating conditions.

	Threads No.	Abb.	R1	R2	R3
Pattern	Mean	ϑ_{avgV}	228.0	228.9	228.3
	Mode	ϑ_{modV}	232.0	232.0	232.0
	Median	ϑ_{midV}	232.0	231.0	232.0
	Std. Deviation	ϑ_{stdV}	4.5	3.6	4.2
Range	Range	ϑ_{gV}	14.0	12.0	14.0
	Minimum	ϑ_{minV}	219.0	222.0	220.0
	Maximum	ϑ_{maxV}	233.0	234.0	234.0

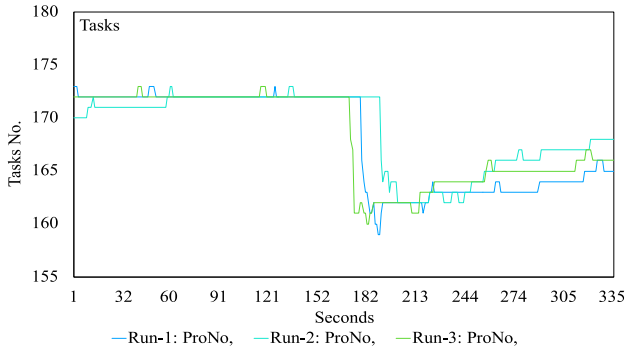


FIGURE 7. Tasks number for R1, R2, and R3.

FIGURE 8. Threads number for R1, R2, and R3.

Figures 7 and 8 demonstrate a significant similarity in the pattern of performance in terms of tasks and threads.

D. CPU TEMPERATURE

When dealing with the ARM processor's temperature, it was necessary to take into account the ambient temperature and consider the initial temperature before experimenting.

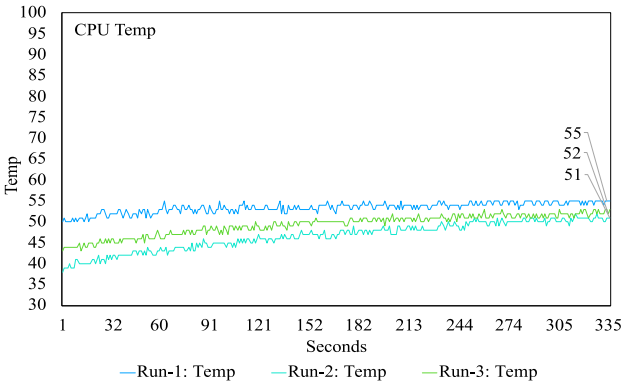


FIGURE 9. CPU temp (°C) for R1, R2, and R3.

TABLE 7. CPU temp under standard operating conditions.

	CPU Temp	Abb.	R1	R2	R3
Pattern	Mean	T_{avgV}	53.45	46.60	49.34
	Mode	T_{modV}	54.00	50.00	51.00
	Median	T_{midV}	54.00	47.00	50.00
	Std. Deviation	T_{stdV}	1.24	3.32	2.45
Range	Range	T_{gV}	5.00	14.00	10.00
	Minimum	T_{minV}	50.00	38.00	43.00
	Maximum	T_{maxV}	55.00	52.00	53.00
	Initial	-	50.00	38.00	43.00
	Ambient	-	23.00	21.00	23.00

By taking into account the ambient and initial temperatures, we will avoid any misleading of these external factors, especially when conducting the attacked scenarios. After conducting the experiment and collecting the data, Table 7 shows the final results for R1, R2, and R3 in terms of CPU temperature. Figure 9 depicts the range and pattern of the processor temperature under normal operating conditions.

Based on Table 7, the CPU temperatures were concentrated (T_{modV}) at 54°C, 50°C, and 51°C for R1, R2, and R3, even when the initial temperature in R2 was at 38°C, which is considered low compared to the other initial temperatures (R1 and R3), where it tended to rise and stabilize between 51°C and 54°C. Regarding the effect of the surrounding environment's temperature, the difference between the T_{modV} values and the outer temperature (ambient) in all ordinary cases was very close as the difference reached 31°C, 29°C, and 28°C for R1, R2, and R3, respectively.

E. POWER CONSUMPTION MEASURING

Among the testbed's parameters, the rate of power consumption was observed, where Tables 8 and 9 show a remarkable similarity and significant convergence in the rate of P in terms of V and I, with the current (I) being more important as the voltage (V) drops slightly while the current changes rapidly and the regulator struggles to maintain the 5V level. As shown in the voltage table results, the V_{modV} was 4.99v in R1 and 5v in R2 and R3, and these values represent the minimum and maximum recorded volts, which are identical in R1, R2, and

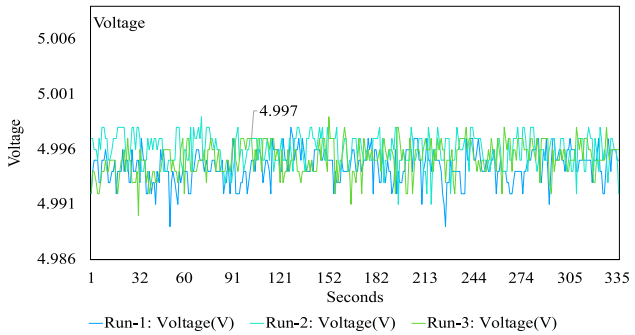


FIGURE 10. Voltage measurements for R1, R2, and R3.

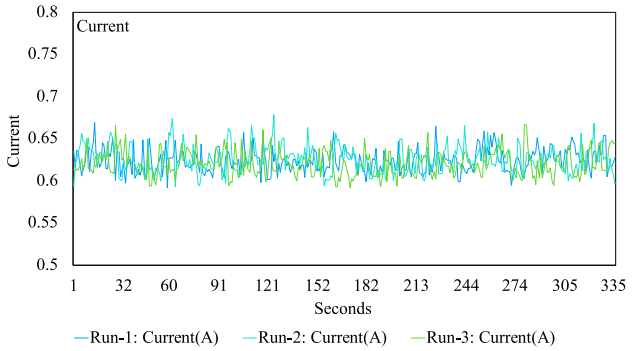


FIGURE 11. Current measurements for R1, R2, and R3.

TABLE 8. Values of voltage (*V*) under typical operating conditions.

	Volt (V)	Abb.	R1	R2	R3
Pattern	Mean	V_{avgV}	4.995	4.996	4.995
	Mode	V_{modV}	4.990	5.000	5.000
	Median	V_{midV}	4.995	4.996	4.995
	Std. Deviation	V_{stdV}	0.001	0.001	0.001
Range	Range	V_{gV}	0.010	0.010	0.010
	Minimum	V_{minV}	4.990	4.990	4.990
	Maximum	V_{maxV}	5.000	5.000	5.000

TABLE 9. Current values (Ampere: *A*) Under typical operating conditions.

	Current *(A)*	Abb.	R1	R2	R3
Pattern	Mean	I_{avgV}	0.624	0.625	0.620
	Mode	I_{modV}	0.610	0.620	0.610
	Median	I_{midV}	0.623	0.625	0.618
	Std. Deviation	I_{stdV}	0.014	0.016	0.015
Range	Range	I_{gV}	0.080	0.090	0.070
	Minimum	I_{minV}	0.590	0.590	0.590
	Maximum	I_{maxV}	0.670	0.680	0.670

R3. Also, the median (V_{midV}) values were almost identical (4.99) in R1, R2 and R3. In terms of the current, the I_r or current patterns were noticeably intertwined, as demonstrated in Figure 11. This pattern formed the normal range of power consumption in terms of current, ranging between 0.59A at its lowest level and 0.68A as the maximum value. Also, the I_{avgV} for R1, R2, and R3 was at $\simeq$ 0.62A, where similarly the I_{midV}, and I_{std} values were converging. These results describe the power consumption rate's stability under normal conditions. Tables 8 and 9 summarize these results, and Figures 10 and 11 show the power consumption patterns and their ranges.

F. DATA TRAFFIC RATE

The data traffic rate is one of the testbed parameters, and as long as the ESs are designed to perform specific functions, we assume that the data transmission rate (transferring and receiving) will be within a determinable range (RX_{gV} and TX_{gV}). Therefore, the number of received and transmitted packets per second were monitored and stored in the log files. The final results of R1, R2, and R3 in terms of the received and transmitted packets count are described in Tables 10 and 11.

TABLE 10. Received packets rate under standard operating conditions.

	RX Packets	Abb.	R1	R2	R3
Pattern	Mean	RX_{avgV}	4.04	4.29	5.00
	Mode	RX_{modV}	0.00	0.00	0.00
	Median	RX_{midV}	1.00	1.00	2.00
	Std. Deviation	RX_{stdV}	5.28	5.72	6.02
Range	Range	RX_{gV}	24.00	27.00	27.00
	Minimum	RX_{minV}	0.00	0.00	0.00
	Maximum	RX_{maxV}	24.00	27.00	27.00

TABLE 11. Transmitted packets rate under standard operating conditions.

	TX Packets	Abb.	R1	R2	R3
Pattern	Mean	TX_{avgV}	4.03	4.29	5.05
	Mode	TX_{modV}	0.00	0.00	0.00
	Median	TX_{midV}	1.00	1.00	2.00
	Std. Deviation	TX_{stdV}	5.27	5.72	6.03
Range	Range	TX_{gV}	24.00	29.00	27.00
	Minimum	TX_{minV}	0.00	0.00	0.00
	Maximum	TX_{maxV}	24.00	29.00	27.00

Based on the results of the analysis criteria of received packets shown in Table 10, it can be concluded that the RX_{avgV} was between 4.04 and 5 packets. The normal distribution range (RX_{gV}) of the received packets rate ranged between 0 to 27 packets based on R1, R2, and R3. In general, these results show remarkable similarity, especially if we consider that the most frequent value (RX_{modV}) is 0 in R1, R2, and R3, indicating that the volume of the received data is low and subject to the purposes for which the system was designed. Also, the RX_{midV} and RX_{stdV} values reflect significant convergence between R1, R2, and R3 in terms of received packets volume. Figure 12 shows the similarities in patterns of the received packets for R1, R2, and R3.

In terms of the transmitted packets, the TX_{modV} and TX_{minV} values were identical for R1, R2, and R3, which is 0, as shown in Table 11. As for the TX_{avgV}, TX_{maxV}, and TX_{rV} of the transmitted packets, the values were not 100 % identical, but they are close enough to shape a determinable range and pattern for the standard model. Table 11 summarizes the final results of the statistical criteria. Figure 13 depicts the similarities in patterns for R1, R2, and R3 in terms of transmitted packets.

Based on the septenary tuple model of the parameters' values $\{U, M, N, \vartheta, T, P, R\}$, and according to the statistical

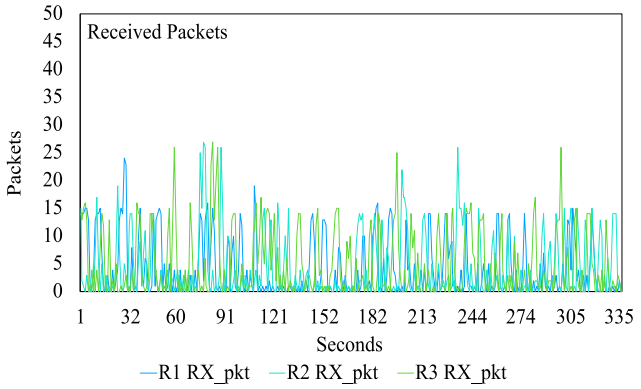


FIGURE 12. Received packets rates for R1, R2, and R3.

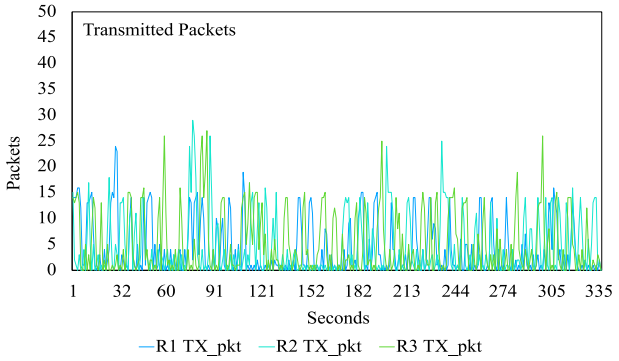


FIGURE 13. Transmitted packets rates for R1, R2, and R3.

TABLE 12. Values of the generated standard model.

	Std_r	Std_P		
	Std_{rV}	Std_{avgV}	Std_{midV}	Std_{modV}
U	$3 > V > 31.9$	$12.0381 > V > 13.0863$	$11.1 > V > 13.2$	$7.5 > V > 8.1$
M	$9.3 > V > 10.9$	$9.4922 > V > 9.5396$	$V = 9.5$	$V = 9.5$
N	$156 > V > 173$	$166.9851 > V > 169.7172$	$170 > V > 173$	$V = 172$
J	$216 > V > 235$	$227.0923 > V > 229.8511$	$230 > V > 233$	$V = 232$
R_{rx}	$0 > V > 30$	$3.0744 > V > 5.9583$	$0 > V > 3$	$V = 0$
R_{tx}	$0 > V > 34$	$3 > V > 6.0804$	$0 > V > 3$	$V = 0$
P_v	$4.99 > V > 5$	$4.993 > V > 4.9975$	$4.994 > V > 4.997$	$4.98 > V > 5.01$
P_I	$0.59 > V > 0.69$	$0.6154 > V > 0.6304$	$0.6112 > V > 0.6313$	$V = 0.61$
T	$26 > V > 58$	$39.747 > V > 60.3006$	$40 > V > 61$	$46 > V > 58$

criteria results, we can say that the range and pattern of the ES's performance and its resource consumption under normal operating conditions fall within a definable range and determinable pattern.

Table 12 shows the values of the standard model in terms of the Std_r and Std_p of the Smart PiCar implemented based on the septenary tuple model. Also, the probability of unlisted, convergent values within the normal range has been calculated by extending the maximal and minimal marginal by adding the highest subtracted value between Rs cases in each parameter for every statistical criterion to the standard range and pattern. The following table shows the determined range and pattern values of standard performance and resource consumption.

Proving the ability to define a standard model for the ES designed to perform specific functions paves the way for the software engineer to apply the necessary restrictions to maintain an optimal resource consumption level for the prevention of any heterogeneous consumption due to illegal activities. However, we will not stop at this conclusion alone, as these results prove the correctness of the first part of the hypothesis. It must be ensured that cyberattacks are causing anomalous consumption patterns that can be distinguished from typical consumption. This approach will be validated computationally to ensure that the ES with its hardware and software layers can distinguish between regular consumption and anomalous consumption. This will be covered later by testing a random dataset of the collected data by using the Support Vector Machine (SVM) algorithm.

VI. ATTACKED CASES RESULTS

A. DENIAL-OF-SERVICE ATTACK (DoS)

According to the Department of Homeland Security official website [25], the denial-of-service (DoS) cyberattack is "a denial-of-service condition accomplished by flooding the targeted host or network with traffic until the target cannot respond or simply crashes, preventing access for legitimate users." DoS is the general description of different types of cyberattacks that tend to breach one of the security goals, which is availability. Some examples of this type of attack are distributed denial-of-service (DDoS), application-layer attacks, advanced persistent DoS, and denial-of-service as a service. Carrying out one of these types of DoS attacks can be done by using many techniques. Also, repelling these types of attacks depends on different defence techniques.

During the testbed, the Low Orbit Ion Cannon (LOIC) tool was implemented to perform a DDoS attack. From Windows virtual machine, the LOIC was used to flood the targeted ES (the victim device) with random data using TCP as the connection protocol, and port 22 was selected as the targeted port. The LOIC is an open-source tool written by C#. The victim device (the Smart PiCar) was flooded with arbitrary messages from another Windows virtual machine by using the User Datagram Protocol (UDP). After the DDoS attack was completed, we started collecting the parameter's values from the log file, and these results reflect the DDoS attack impact on system performance. Table 13 summarizes the final results.

TABLE 13. Parameters values under DoS attack.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
CPU	31.90	31.40	30.90	6.26	47.30	6.80	54.10
Mem	10.14	10.20	10.20	0.27	1.80	9.70	11.50
Tasks	181.64	184.00	171.00	9.39	33.00	166.00	199.00
Threads	241.76	244.00	249.00	9.39	33.00	226.00	259.00
Voltage	4.98	4.98	4.98	0.00	0.01	4.98	4.99
Current	0.61	0.61	0.61	0.02	0.08	0.56	0.64
Temp	61.57	62.00	64.00	2.72	10.00	55.00	65.00

Based on the results that are shown in the above table, the CPU load reached 54.1% as the maximum percentage

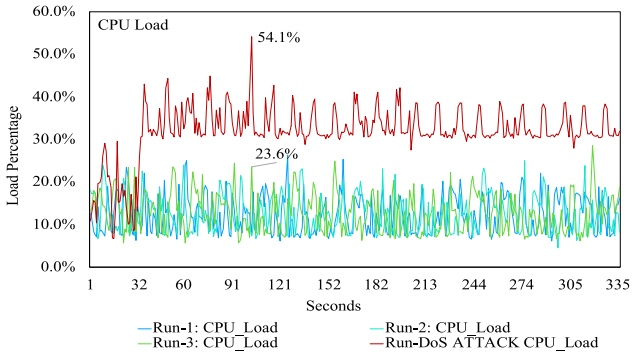


FIGURE 14. CPU load pattern DoS attack vs R1, R2, and R3.

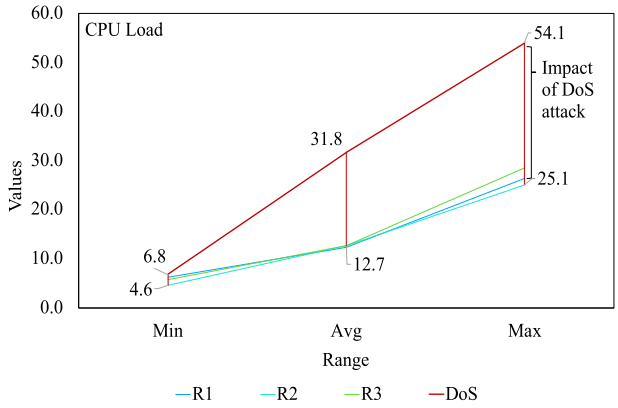


FIGURE 15. CPU range under DoS attack vs R1, R2, and R3.

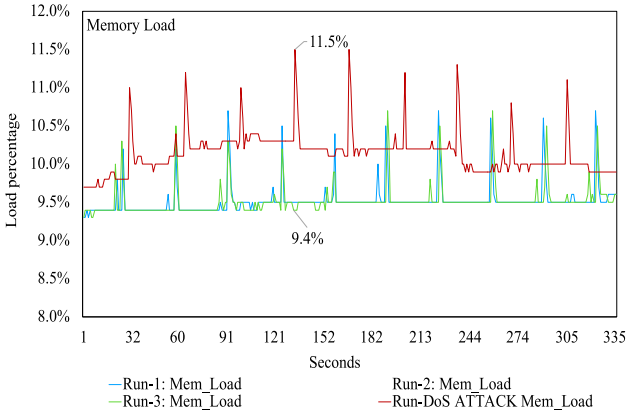


FIGURE 16. Memory load under DoS attack vs R1, R2, and R3.

and the most frequent value is concentrated at 30.90%. The CPU utilization values were also widely distributed from 6.80 to 54.10%. Thus, it reached 47.30 as the range of values distribution, reflecting the magnitude of the large fluctuation in performance and instability in the pattern. In terms of CPU performance, the following chart reflects the impact of the DDoS attack on the CPU utilization pattern compared to the typical performance.

The difference between typical performance and anomalous utilization while the system is exposed to the DoS attack can be seen clearly based on Figure 14 (time in seconds). Also, Figure 15 shows the DoS attack's impact on CPU resources compared to typical utilization in terms of minimum, average, and maximum values.

In terms of memory usage, the most repeated value was 10.20% (see Table 13) and the maximum value was 11.50%, while the minimal was 9.70%. The range of propagation or distribution of the consumption values was between 9.70% and 11.50%. These values reflect a clear difference compared to regular consumption in terms of range (M_r) and pattern (M_p). Figure 16 shows the impact of a DoS attack on memory resources.

Tasks and threads are among the parameters whose values were collected during the testbed. These two parameters showed high accuracy in monitoring the performance of the Smart PiCar, and even minor changes have been reflected in these parameters. Based on the values in Table 13, the most common value of alive tasks during the exposure of the ES to the DoS attack was 171 tasks, but the average value of alive tasks was 181.6. This was despite the concentration value or density point, which was 171 tasks per second, but due to the high fluctuation between the lowest value 166 and the highest value 199, and due to the wide distribution range of tasks number per second, which reached 30 processes as the difference between the lowest and highest value, an unbalanced and anomalous performance pattern has been formed and this is attributed to the abnormal activity. Also, it is worth mentioning that the most common value occurred and was recorded at the end of the DoS attack. While at the peak of the DoS attack, the impact caused a significant number of tasks until it reached 199 processes per second. Figure 17 demonstrates the anomalous performance of the Smart PiCar in terms of alive processes as a result of the DoS attack. Moreover, Figure 18 shows how the number of the tasks under the DoS attack diverge significantly from the N_r and N_p of *Rs* and *Std* cases, which is represented by the blue area in Figure 18.

Regarding the pattern of executed threads, the results did not differ from task patterns due to their relationship. The results showed that the number of executed threads at the peak of the DoS attack was 259 threads and the most frequent value was 249 threads. As with the value of the distribution range, the difference between the highest and lowest value (33 threads) caused an abnormal and unstable performance pattern. Thus, this significant difference is considered an anomalous performance. Figures 19 and 20 show the effect that the DoS attack has had on the embedded system's performance in terms of executed threads per second. Based on Figure 20, the typical performance range can be seen within the blue area. However, we can see how the DoS attack's effect took the number of executed threads to unprecedented levels, which caused unfamiliar consumption and pattern.

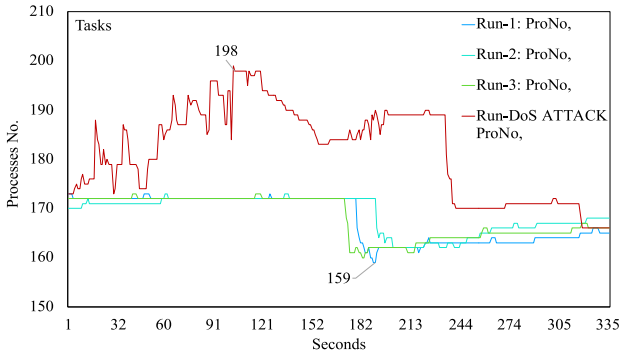


FIGURE 17. Tasks number under DoS attack vs R1, R2, and R3.

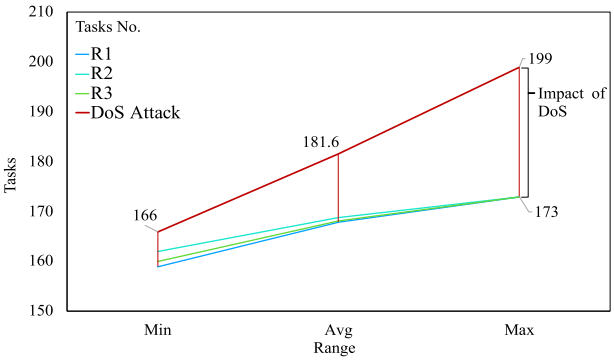


FIGURE 18. Tasks range (XgV) under DoS attack vs R1, R2, and R3.

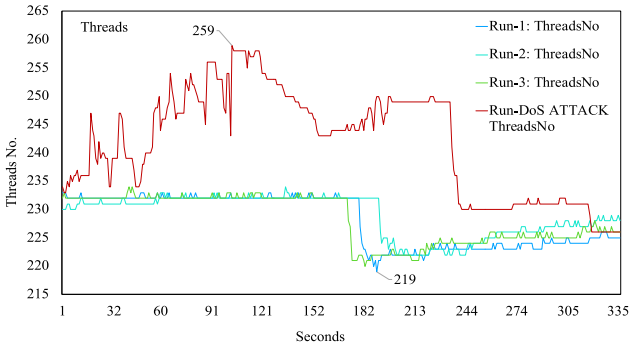


FIGURE 19. Threads pattern under DoS attack vs R1, R2, and R3.

Thus, the threads' values have deviated significantly out of the blue area.

Concerning voltage performance, an apparent deviation from the normal range is observed as shown in Figure 21. The reason for this anomalous power consumption is due to the effect of the DoS attack on the system's functions, as it was noted that the ultrasonic avoidance module was no longer functioning normally, and the connection with the administrator's device had become unstable as a result of the flooded traffic generated by the attacker.

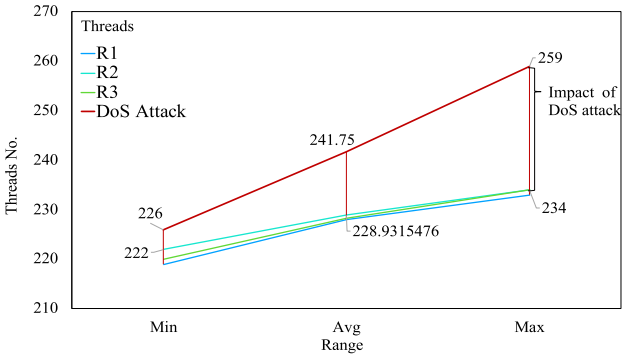


FIGURE 20. Threads range (XgV) under DoS attack vs R1, R2, and R3.

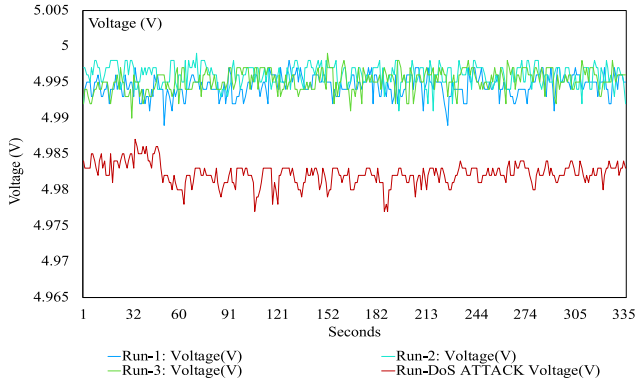


FIGURE 21. Voltage measurements under DoS attack vs R1, R2, and R3.

Thus, this apparent decrease indicates the presence of an attack that disrupts the system's functions and that a significant voltage drop caused it. Accordingly, if we exclude any hardware deficiencies, we can infer the existence of an attack based on the unfamiliar consumed voltage pattern compared to the V_p in Rs cases. It was also noticed that the effect of the DoS attack was not evident in terms of the flow rate of electricity (current) except during the first few seconds of the attack, as shown in Figure 22. Thus, the significant interference in the current rate under conditions of attack or its absence makes this sub-parameter independently insufficient to predict and detect the attack because it did not achieve the anomalies and distinguishability principles. However, supporting this sub- parameter with other parameters' impacts will help clarify the anomalous pattern caused by the DoS attack. Figure 23 depicts the impact of the DoS attack in terms of current power consumption.

The DoS attack's impact not only diminished on the direct parameters of system resources, but it impacted the temperature of the Smart PiCar processor (ARMv8). The CPU temperatures recorded abnormal values that were not observed previously under normal operating conditions. The initial temperature was 55°C, which is within the normal range, and the ambient temperature was 24°C, which was calculated to

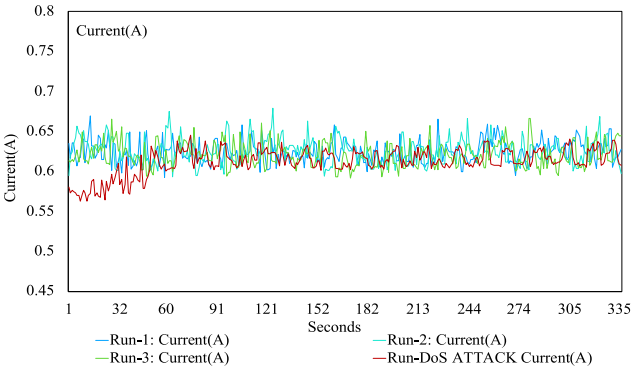


FIGURE 22. Current measurements under DoS attack vs R1, R2, and R3.

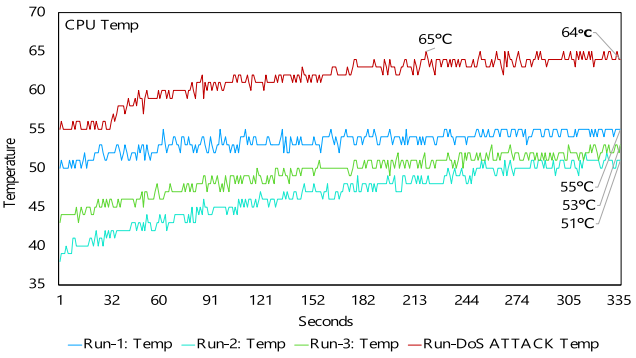


FIGURE 23. CPU Temp under DoS attack vs R1, R2, and R3.

avoid any contribution by an external factor. The following figure shows the effect of the DoS attack on the processor temperature.

During the attack, the CPU temperature continued to increase significantly. The difference between the maximum recorded temperature and the ambient temperature reached 41°C. Thus, this is an anomalous value of CPU temperature compared to 31°C, 29°C, and 28°C for R1, R2, and R3.

B. PASSWORD ATTACK (BRUTE-FORCE ATTACK)

According to the *Encyclopaedia of Cryptography and Security* [26], a dictionary attack describes a password-guessing technique in which the attacker attempts to determine a user's name or password by successively trying a compiled list of likely words, numbers, and symbols. In this test, ''Brute-Dum'' has been used as a brute-forcing tool. Within this tool, Network Mapper (Nmap) has been used for port scanning. After selecting a specific port, the SSH port has been selected for the attack purpose. The Hydra tool has been activated as a parallelized network logon cracker, supported with a random text password list. The initial expectation was that this type of attack would not have a monitorable effect or an indistinguishable impact. However, the obtained results after the attack was carried out are promising and auspicious for the effectiveness of the approach we are seeking to deliver. Table 14 summarizes the final results obtained

TABLE 14. Parameters values under brute-force attack.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
CPU	31.16	31.50	30.50	9.13	54.70	6.50	61.20
Mem	9.78	9.60	9.60	0.33	1.90	9.40	11.30
Tasks	189.10	180.00	180.00	17.47	50.00	163.00	213.00
Threads	249.23	240.00	240.00	17.52	50.00	223.00	273.00
Voltage	4.99	4.99	4.99	0.00	0.01	4.99	5.00
Current	0.63	0.63	0.63	0.02	0.15	0.54	0.69
Temp	58.11	59.00	60.00	3.29	13.00	50.00	63.00

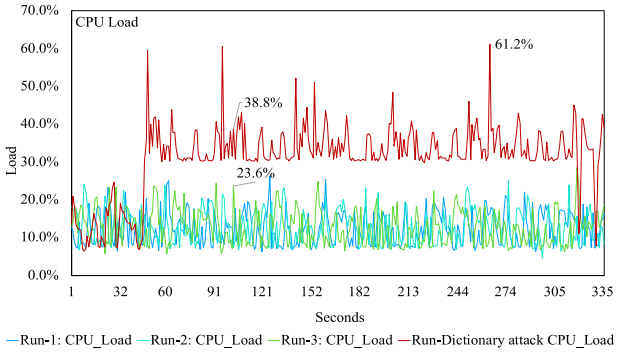


FIGURE 24. CPU load pattern under BF attack vs R1, R2, and R3.

after calculating the values of the statistical criteria for the septenary tuple model.

The results were as follows. In terms of CPU utilization rate, the maximum usage was 61.20% and the most common value was 30.50. The minimum level was 6.50%. Thus, the range of distribution of values is vast, reaching 54.70. Also, the mean value is 31.16, which is close to the most repeated value. Figure 24 gives an impression of the effect that the password attack had on the Smart PiCar processor's performance compared to the typical patterns.

Moreover, the standard deviation values of tasks and threads were significantly higher than the typical values in R1, R2, and R3. Therefore, this performance pattern of CPU utilization diverges significantly from the standard range and pattern.

The memory utilization rate reached its highest limit at 11.30. This value had not been previously recorded under normal conditions. For the mean and mode values, the results were 9.78 and 9.60, respectively. Although these values overlapped within the normal range Std_r of memory usage, the performance pattern appeared unfamiliar and anomalous compared to the standard pattern (Std_p), as shown in Figure 25. Furthermore, adding the anomalous patterns of other parameter impacts to this parameter will increase the attack recognition accuracy of detecting the existence of a password attack. Figure 25 depicts the impact of the password attack on memory usage.

In terms of the executed number of processes and threads resulting from the password attack, the highest number of

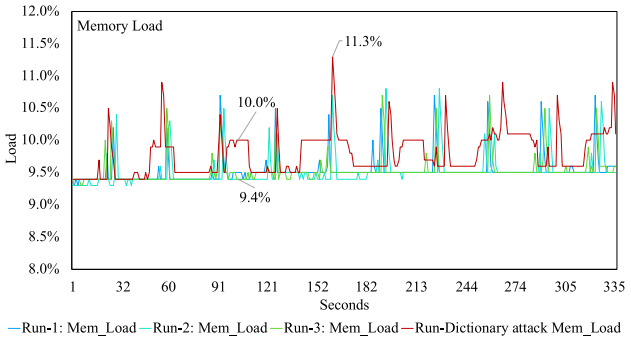


FIGURE 25. Memory load pattern under BF attack vs R1, R2, and R3.

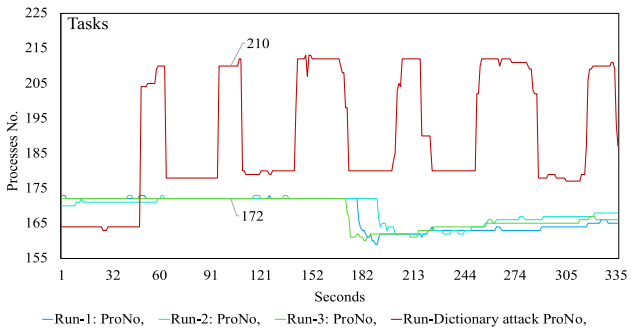


FIGURE 26. Tasks pattern under BF attack vs R1, R2, and R3.

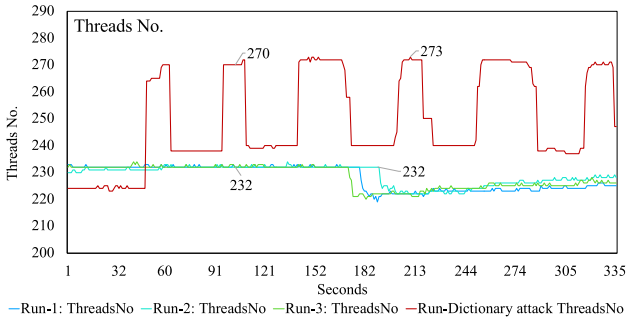


FIGURE 27. Threads pattern under password attack vs R1, R2, and R3.

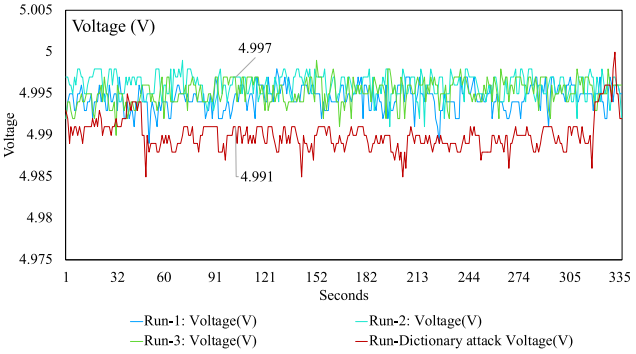


FIGURE 28. Voltage pattern under BF attack vs R1, R2, and R3.

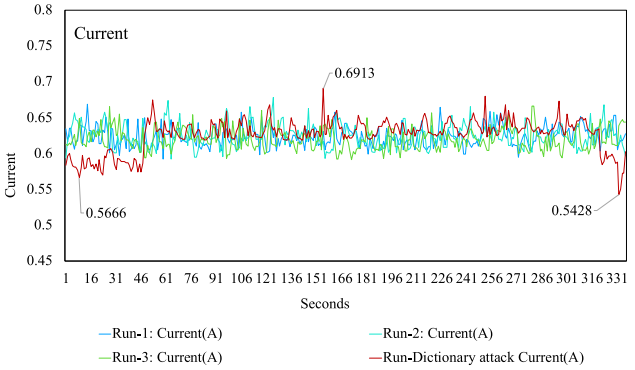


FIGURE 29. Current pattern under BF attack vs R1, R2, and R3.

tasks reached 213 alive processes and the lowest number was 163 processes. Thereby, the attack distributed the number of the tasks into a large scale and caused an oscillating pattern. Likewise, the mode value was 180 and the mean was 189.1, which sharply deviated from the standard patterns and formed an anomalous pattern that was distinguishable. Figure 26 illustrates the executed task pattern under the password attack.

Regarding the pattern of executed threads, the similarity with the pattern of executed processes is clear, as shown in Figures 26 and 27. This similarity is due to their association with each other. Therefore, the tasks and threads patterns are heading in the same direction of anomalies, and these two parameters deviated from the standard pattern. Figure 27 shows how the threads pattern under the password attack differed from the normal range.

In terms of power consumption under the password attack, Figures 28 and 29 depict the password attack effect on the power consumption rate in terms of voltage or current. This pattern differs from the regular pattern and is similar to the impact pattern of the DDoS attack, especially in terms of voltage, which appeared decreased compared to typical patterns in R1, R2, and R3. Thus, the rate of anomalous consumption of power is a significant indicator that deserves attention. However, the decrease in voltage is not due to the cyberattack's direct impact, but instead to the attack's indirect effect on the system's functions. Thereby, this deficiency in the system's functions has been reflected in the voltage and current patterns, indicating the existence of suspicious activities.

Regarding the effect of the password attack on CPU temperature, an increase in CPU temperature was observed. The CPU temperature under the attack differed from the typical patterns (T_p) in R1, R2, and R3. Thus, this abnormal pattern in temperature proved the existence of illegal activity that caused an increase in computing operations, which thus produced an increase in CPU temperature. Therefore, defining the pattern and range of CPU temperature under normal operating conditions, taking into account the ambient temperature (25°C, in this scenario) and initial temperature (51°C, in this scenario) of the system, and monitoring any anomalous temperatures (63°C, as was recorded due to the attack impact) will contribute to predicting the presence of cyberattacks on

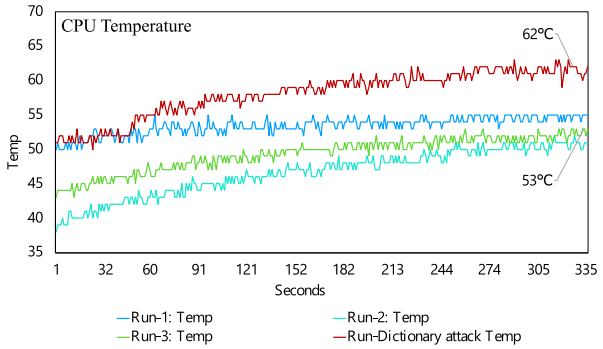


FIGURE 30. CPU temp under BF attack vs R1, R2, and R3.

the system. Figure 30 depicts the CPU temperature pattern under the password attack.

C. REMOTE CODE EXECUTION ATTACK (RCE)

The third attack scenario that was executed is the remote code execution attack (RCE). The RCE is an attacker's ability to gain unauthorized access to a remote device and execute malicious software. In other words, the attacker will run a local code to be executed remotely in the victim's device [27]. The following command was executed to generate this attack scenario: '`sudo ssh pi@192.168.8.117 python -u - < cameraRemotly.py`'. As shown in this command's syntax, it consists of malicious software (*cameraRemotly.py*), written in Python language to intentionally cause damage to the Smart PiCar through its targeted IP address and disrupt one of its functions by disabling its Pi camera remotely. Table 15 shows the final results of the statistical criteria of the testbed's parameters under the RCE attack.

The most common value was 6.20% in terms of CPU utilization, whereas the typical utilization's mode values were between 7.5% and 7.9%. The mean value was 12.33% and the minimum and maximum values were 4.9 and 37.1, respectively, which created a wide distribution range and fluctuating performance, especially in the first few seconds of the RCE attack. Despite interference with the typical performance, a different CPU utilization pattern had been shaped due to the RCE attack. For example, the U_{modV} values of CPU utilization in R1, R2, and R3 was between 7.5% and 7.9%, while under RCE attack the mode value was 6.20%. Figure 31 illustrates the impact of the RCE attack on CPU utilization compared to the typical patterns.

TABLE 15. Parameters values under remote code execution attack.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
CPU	12.33	10.40	6.20	7.51	32.20	4.90	37.10
Mem	9.90	9.90	9.90	0.00	0.00	9.90	9.90
Tasks	167.45	168.00	169.00	2.44	11.00	162.00	173.00
Threads	220.56	221.00	221.00	2.49	11.00	215.00	226.00
Voltage	5.00	5.00	5.00	0.00	0.01	4.99	5.01
Current	0.56	0.55	0.55	0.03	0.14	0.51	0.65
RX pkt	3.99	1.00	0.00	5.26	27.00	0.00	27.00
TX_pkt	4.06	1.00	0.00	5.29	26.00	0.00	26.00
Temp	54.63	54.00	54.00	1.95	8.00	52.00	60.00

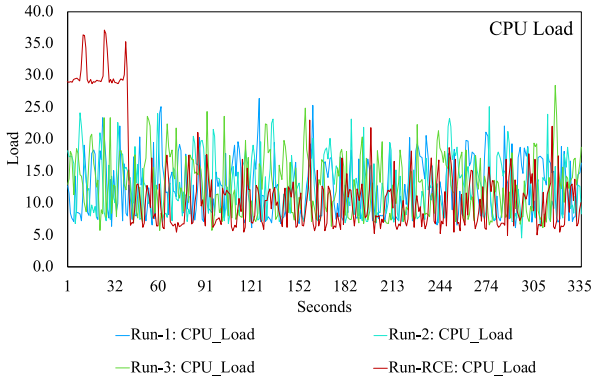


FIGURE 31. CPU load pattern under RCE attack vs R1, R2, and R3.

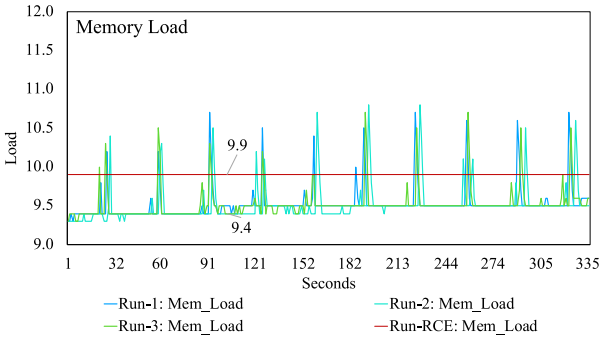


FIGURE 32. Memory load pattern under RCE attack vs R1, R2, and R3.

As for memory usage percentage, the results that we obtained for this parameter are essential. Note that the range value is 0 and the lowest, highest, most frequent, and average values are identical (9.9%). This indicates the absence of any change in memory usage, which demands attention. The absence of change in memory usage indicates the absence of changeable activities, which indicates the presence of a defect resulting from a failure in one of the system functions. This failure can be inferred as an indicator of the presence of illegitimate activity, especially if we exclude any known manufacturer's defect. Disrupting the Pi camera function due to the RCE attack was evidently reflected in the values of this parameter. The irregular pattern of memory usage under the RCE attack and its impact on the memory usage rate compared to the typical patterns (M_p) are demonstrated in Figure 32.

In terms of the number of executed processes under the RCE attack conditions, the most frequent value was 169 and the average was 167.45. The distribution range (X_{gV}) values reached 11 tasks as the difference between the lowest value (162) and the highest value (173). Figure 33 shows the anomalous pattern of this parameter compared to the typical performance.

Regarding the number of executed threads, the formed pattern resulting from the RCE attack is shown in Figure 34. This parameter's pattern was unfamiliar and deviates entirely outside the normal range, as at the beginning of the attack period the minimum value was 215 threads, the maximum value was 226 threads, and the most frequent value (mode) was 221 threads. The following chart shows how the threads

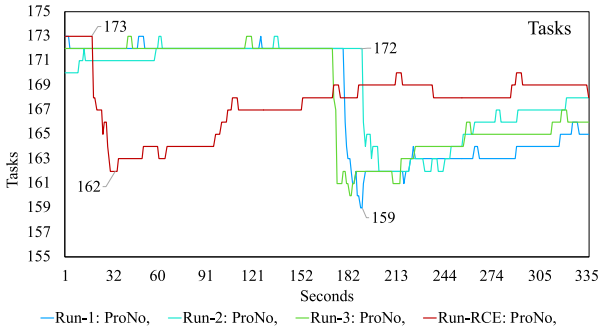


FIGURE 33. Alive tasks pattern under RCE attack vs R1, R2, and R3.

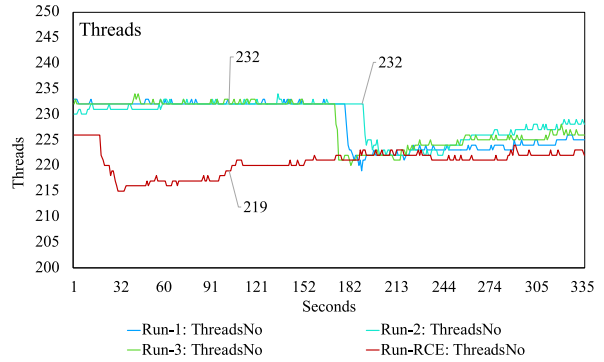


FIGURE 34. Threads pattern under RCE attack vs R1, R2, and R3.

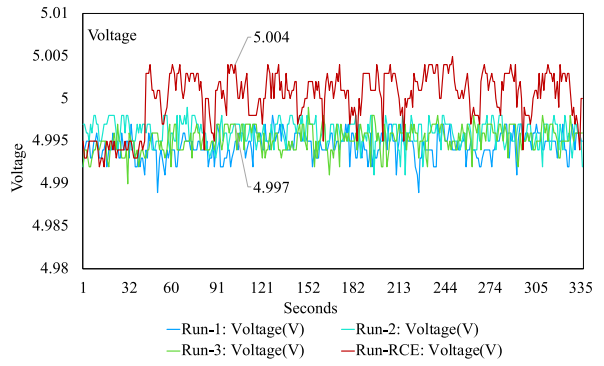


FIGURE 35. Voltage pattern under RCE attack vs R1, R2, and R3.

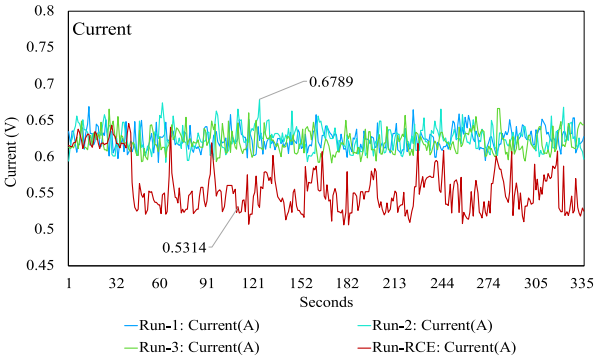


FIGURE 36. Current pattern under RCE attack vs R1, R2, and R3.

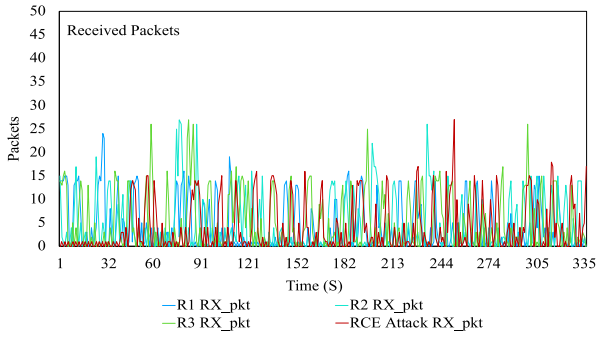


FIGURE 37. Received packets rates under RCE attack vs R1, R2, and R3.

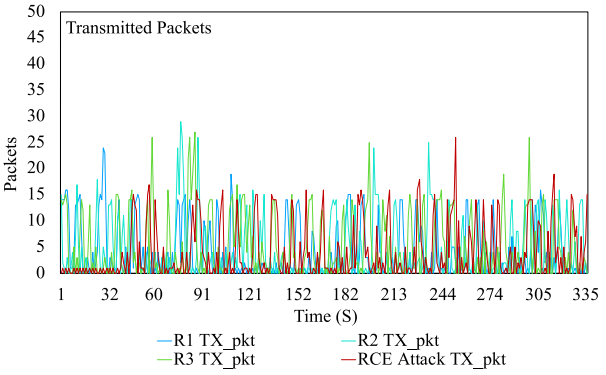


FIGURE 38. Transmitted packets rates under RCE attack vs R1, R2, and R3.

number under the RCE attack deviates from the typical patterns.

For the first time, change has appeared in both the voltage and current in terms of energy consumption. In the voltage, change has appeared in the form of an increase, while in the current it appeared in the form of a decrease. This is a crucial indication of the malfunction of one of the system's functions, hereby detecting an attack depending on the anomalous energy consumption rate is an effective method, especially if this is done in addition to monitoring other parameters. Figures 35 and 36 show the abnormality of the power consumption and how it differed from the typical patterns.

In this attack scenario, transmitted and received packets have been monitored. Figures 37 and 38 show the effect of the RCE attack on the data traffic rate. Based on the analytical criteria, there was interference in the range but with a pattern difference. The slight abnormality or anomalous pattern in transmitted and received packets will support the distinguishability between typical and anomalous traffic, especially if the other parameters' impacts have been considered. In this attack case, the impact might not be noticeable enough in this parameter as it will be in the next scenario. Furthermore, this kind of parameter might help in determining the type of attack as a network attack. For example, in a packet drop attack or blackhole attack [28], anomalous change in the expected predetermined packets volume can lead to the detection of this kind of attack based on this parameter.

Concerning CPU temperature under the conditions of this type of attack and after considering the initial (59°C) and ambient (25°C) temperatures, it is noticeable that disabling one of the embedded system's functions due to the attack was reflected by the CPU temperature. This reflection has complicated the possibility of predicting the existence of a cyberattack based on this parameter alone. Therefore, the decrease

in computing operations caused a decrease in CPU load, allowed the CPU temperature to decrease. It is worth noting that this decrease placed the temperature within the range of familiar patterns (T_p). Thus, it is impossible to rely on this parameter to infer the anomalous pattern that indicates a cyberattack's existence. However, this observation is based on a relatively short period. Thus, a long operational period might show a different pattern of CPU temperature if the attack is held for a while, causing a significantly noticeable decrease due to function failure. The following figure depicts the impact of the RCE attack on CPU temperature.

D. MAN-IN-THE-MIDDLE ATTACK (MITM) REMOTE COMMAND EXECUTION

In this attack scenario, the attacker not only disabled the recording function of the equipped Pi camera, but also redirected the camera broadcast to their device by executing a remote command, which is known as a remote command execution attack. This attack type differs from the remote code execution attack, which had been executed in the previous scenario. By executing the following command in terminal: `'raspivid -o - -t 0 -w 960 -h 540 | cvlc -vvvstream:///dev/stdin –sout '#rtp{sdp = rtsp://:8554/}':demux = h264'`, the adversarial will gain unauthorized streamed video. Furthermore, the passive repetition of this illegitimate activity, without harming the system (only eavesdropping), will make the attack an advanced persistent threat (APT). During the attack period, the testbed's parameters were monitored and the final results of the analysis criteria are shown in Table 16.

TABLE 16. Parameters values under MITM attack.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
CPU	15.87	12.60	7.40	9.05	33.60	5.30	38.90
Mem	10.50	10.50	10.50	0.01	0.10	10.50	10.60
Tasks	171.79	173.00	173.00	1.70	10.00	167.00	177.00
Threads	239.89	241.00	241.00	1.78	10.00	235.00	245.00
Voltage	4.99	5.00	5.00	0.00	0.02	4.98	5.00
Current	0.62	0.61	0.61	0.03	0.17	0.53	0.69
RX pkt	7.79	3.00	0.00	10.37	55.00	0.00	55.00
TX pkt	284.29	307.00	1.00	148.68	645.00	1.00	646.00
Temp	56.15	56.00	55.00	1.34	6.00	54.00	60.00

In terms of CPU utilization, the most common value under these conditions was 7.40%, while the mean value is 15.87%. The lowest value was 5.30% and the highest value was 38.90%. The general pattern of CPU utilization was fluctuating and anomalous in comparison to the Std_p pattern. Figure 40 gives a clear vision of the CPU utilization pattern under this attack, and it shows how the CPU performance pattern of the attacked scenario differs from the typical patterns (U_p) in R1, R2, and R3.

In terms of memory usage, a steady and high pattern of memory usage has been observed, and this steadiness is

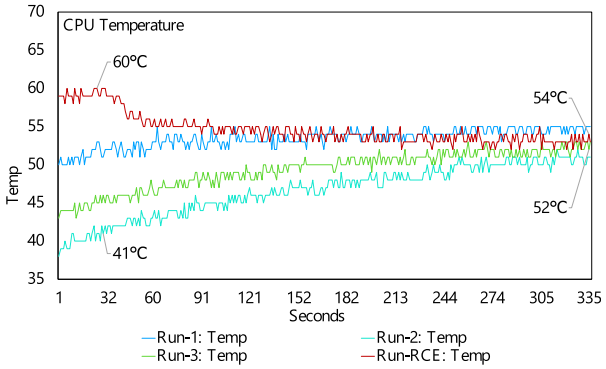


FIGURE 39. CPU temp under RCE attack vs R1, R2, and R3.

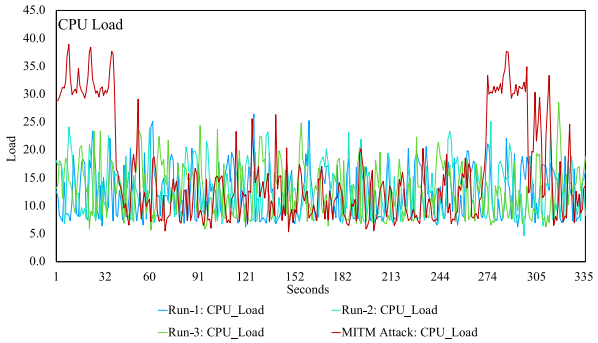


FIGURE 40. CPU load pattern under MITM attack vs R1, R2, and R3.

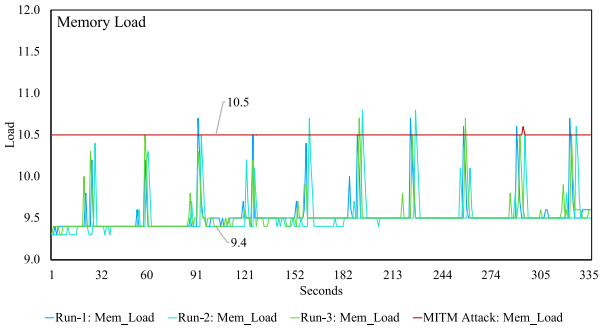


FIGURE 41. Memory load pattern under MITM attack vs R1, R2, and R3.

abnormal in this case. Also, the standard deviation value was 0.01 compared to 0.2 in Rs cases. The mode and mean values were stable at 10.50%. This performance pattern's steadiness appeared utterly different from the typical patterns (M_p) in Rs cases due to the MITM attack. Figure 41 illustrates the memory performance patterns under the attack and typical operational circumstances.

In terms of the number of tasks executed per second, Figure 42 shows the pattern of the executed tasks/s, which ranged from 167 to 177 tasks/s, with173 the most common value (mode). Although there was interference within the typical ranges (N_p), the anomalies in the executed tasks pattern under the MITM attack were sufficient to characterize the tasks pattern as an anomalous performance pattern, thus shaping an abnormal pattern with the system performance indicating the presence of an attack. Therefore, this situation facilitates the ability to detect the attack based on the attack's impact on

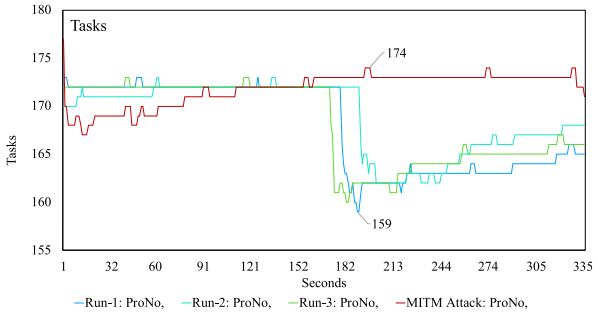


FIGURE 42. Tasks pattern under MITM attack vs R1, R2, and R3.

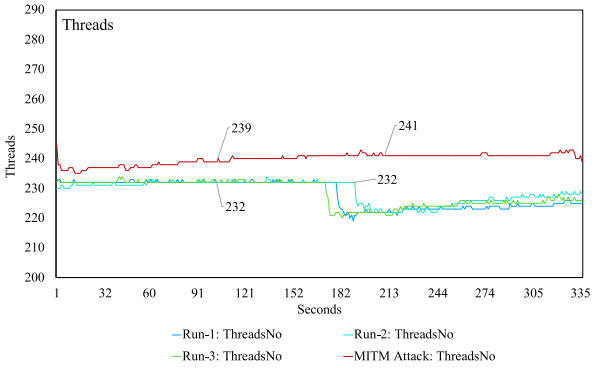


FIGURE 43. Threads pattern under MITM attack vs R1, R2, and R3.

system performance. The following figure depicts the pattern of this parameter under the MITM attack.

Although there was interference within the Std_r, the abnormality in this pattern was sufficient to characterize the performance as an anomaly performance due to the presence of the MITM attack. Hence, this situation facilitates the ability of detection when making a comparison to the supposed performance.

Regarding the number of executed threads, the pattern formed from the attack is shown in Figure 43. It was unfamiliar and deviated entirely outside the normal range (ϑ_r) as its minimum value was 235 threads, which is higher than the maximum value in the normal operation conditions (234 threads). The mode value was 221 threads and the maximum number of executed threads was 245. Overall, this parameter's pattern is categorized within the anomaly patterns compared to R1, R2, and R3, as illustrated in the following figure.

Regarding the change in power consumption under the MITM attack conditions, it was noted that the change was more pronounced in the pattern rather than range. The consumption rate of V and I during the attack was intertwined with the normal P_p but differently. At the end of the period, there was a sharp fluctuation due to the MITM attack's impact. In terms of voltage mode, minimum and maximum values were 5_{V}, 4.98_{V}, and 5_{V}, respectively. In terms of the current rate, the mode was 0.61_{A} and the minimum and maximum values were 0.53_{A} and 0.69_{A}, sequentially. Figures 44 and 45 demonstrate the impact of the MITM attack on this parameter $P(V$ and $I)$.

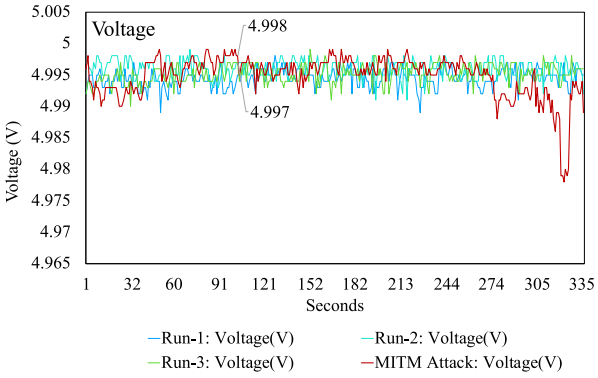


FIGURE 44. Voltage pattern under MITM attack vs R1, R2, and R3.

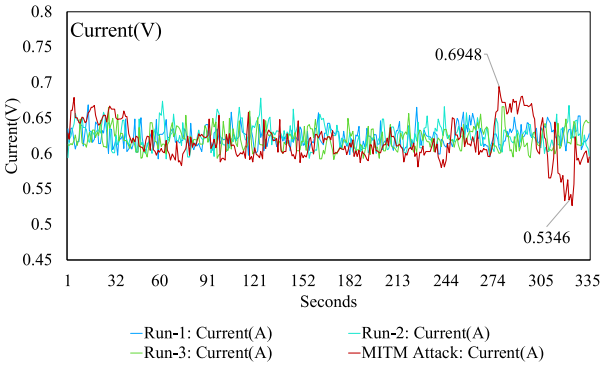


FIGURE 45. Current pattern under MITM attack vs R1, R2, and R3.

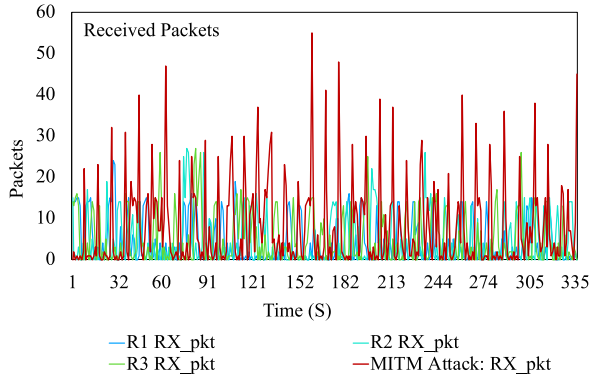


FIGURE 46. Received packets rates under MITM attack vs R1, R2, and R3.

In terms of the data traffic rate under the circumstances of the MITM attack can be observed through Figures 46 and 47 that the patterns have changed dramatically, which confirms the presence of illegitimate activity. Also, the analysis criteria (*mean, mode, median, std.d, range, min,* and *max*) for TX/RX packets showed a significant difference and the results deviated significantly outside the normal standard range (Std_r), as well as in the pattern (Std_p), especially in terms of transmitted packets number. The mean value of the transmitted packets under the conditions of this type of attack was 284.29 packets, while the TX_{avgV} in Rs cases was between 4 to 5 packets/s, which is considered an enormous amount of data compared to the typical transmission rates (TX_r and TX_p) in Rs cases, proving undoubtedly the existence of unauthorized activities.

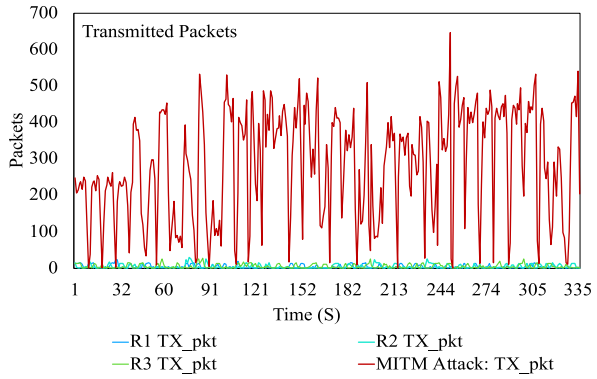


FIGURE 47. Transmitted packets under MITM attack vs R1, R2, and R3.

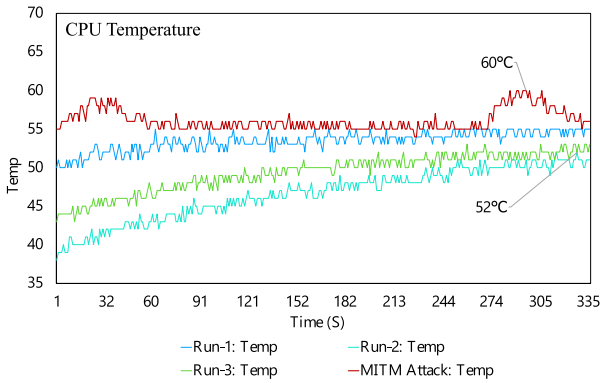


FIGURE 48. CPU temp under MITM attack vs R1, R2, and R3.

Regarding the CPU temperature pattern under the attacker's influence (as the attacker redirects the camera back to his device), the temperatures were close to the typical ranges but relatively anomalous in the pattern. The most common temperature was 55°C, the maximum was 60°C, the minimum was 54°C, and the average value was 56.15°C. In the first and last seconds of the attack, the CPU temperature increased for a short period, forming a different pattern from the usual pattern. Altogether, adding this parameter pattern to other parameters' impacts will enable the detection of the cyberattack's presence. Figure 48 summarizes the CPU temperature pattern in this attack scenario compared to other typical patterns. The initial temperature was 55°C and the ambient temperature was 25°C.

VII. ANALYSIS AND DISCUSSION

A. CPU UTILIZATION PARAMETER

Based on the results shown in the CPU utilization parameter (Table 17), it is clear to us that the CPU utilization rates under normal operational conditions (R1, R2, and R3) are very similar, whether in terms of ranges or patterns.

The typical range of CPU utilization extends from 4.6% as the lowest value among Rs cases to 28.50% as the highest value, and the typical pattern of CPU utilization (mode value) was concentrated at 7.90%, 7.70%, and 7.50% for R1, R2, and R3, respectively. Based on the significant similarity of these results in terms of range and pattern, we defined the standard pattern (Std_p) and range (Std_r) of CPU utilization. Thereby,

TABLE 17. Overall CPU performance for all cases.

	Pattern				Range (XgV)		
	Mean	Median	Mode	Std.D	Range	Min	Max
R1	12.39	11.80	7.90	4.40	20.20	6.20	26.40
R2	12.68	12.40	7.70	4.35	20.50	4.60	25.10
R3	12.74	12.50	7.50	4.53	22.80	5.70	28.50
DoS	**31.90**	**31.40**	**30.90**	**6.26**	**47.30**	**6.80**	**54.10**
BF	**31.16**	**31.50**	**30.50**	**9.13**	**54.70**	**6.50**	**61.20**
RCE	**12.33**	**10.40**	**6.20**	**7.51**	**32.20**	**4.90**	**37.10**
MITM	**15.87**	**12.60**	**7.40**	**9.05**	**33.60**	**5.30**	**38.90**

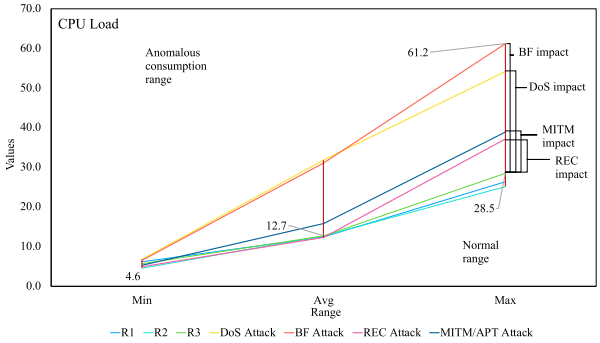


FIGURE 49. Overall CPU ranges for attacked cases vs standard cases.

the validity of the first part of the hypothesis in terms of this parameter has been proven.

In contrast, the CPU utilization under all cyberattack conditions without exception had shown anomalous patterns. Consequently, based on the mathematical results of the collected and analyzed data, it is possible to distinguish the regular consumption from anomalous consumption in terms of range and pattern. Therefore, as long as we were able to define a standard utilization of the CPU based on the embedded system's functions, the cyberattacks had a distinguishable anomalous impact on the CPU performance. Thus, it has been proven to us that cyberattacks can be detected based on this technique. When looking closely at the values in Table 17, we notice that the mean values in all cyberattack cases are sufficiently different except for the RCE attack, which is somewhat close to the average of R1, R2, and R3. However, it differed in its range, pattern, lowest values, and highest values, which created a discernible anomalous pattern that enabled its distinction, especially if we consider the other parameters' anomalous patterns impacts. In Figure 49, we see that the most deviated impacts of an attack from the normal range and pattern in terms of CPU utilization compared to typical performance were the password attack (brute-force attack), followed by the denial-of-service attack, man-in-the-middle attack, and finally the remote code execution attack. Table 17 and Figure 49 show the effect that cyberattacks have had on the performance of the CPU (represented in the red area). The abnormality of CPU utilization under the attack scenarios shows how these scenarios' CPU patterns deviated outside the normal range (represented in the blue area).

B. MEMORY USAGE PARAMETER

Table 18 summarizes the final results that we obtained, and it clearly demonstrates that memory consumption in optimal operating conditions was not only similar but identical in terms of the most common value (9.50%), and the average is almost 9.5% in R1, R2, and R3.

TABLE 18. Overall memory performance for all cases.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
R1	9.52	9.50	9.50	0.21	1.40	9.30	10.70
R2	9.51	9.50	9.50	0.23	1.50	9.30	10.80
R3	9.52	9.50	9.50	0.20	1.40	9.30	10.70
DoS	**10.14**	**10.20**	**10.20**	**0.27**	**1.80**	**9.70**	**11.50**
BF	**9.78**	**9.60**	**9.60**	**0.33**	**1.90**	**9.40**	**11.30**
RCE	**9.90**	**9.90**	**9.90**	**0.00**	**0.00**	**9.90**	**9.90**
MITM	**10.50**	**10.50**	**10.50**	**0.01**	**0.10**	**10.50**	**10.60**

The lowest recorded value in R1, R2, and R3 was 9.30%. The highest values were 10.7%, 10.8%, and 10.7% for R1, R2, and R3, respectively. Thus, the possibility to define a standard pattern and range of the typical memory consumption due to its similarity pattern, which has reached a congruence level, has been proven.

Conversely, after analyzing the pattern of memory consumption in the context of cyberattack cases, we notice that the general characteristic of the consumption pattern is anomalous compared to the standard patterns in R1, R2, and R3. Furthermore, defining the regular optimal pattern implicitly necessitates that any other patterns will be anomalous.

Also, it is worth mentioning here that the impact of the cyberattacks does not necessitate the appearance of an abnormal increase, but rather, it may appear in the form of a decrease that calls attention and indicates the existence of function failure, such as that which occurred in the RCE attack scenario. In addition, the values distribution ranges in RCE and MITM attacks were 0 and 0.1, respectively, indicating the absence of alive activities and processes. Thus, it can be concluded that there was a malfunction in the system functions, and this is what exactly happened as the attacker in the RCE scenario disrupted the recording camera. In the MITM attack, the camera was directed to an attacker's device to monitor streaming broadcasts instead of the recording function that was supposed to take place if it had not been hacked. Figure 50 demonstrates how the DoS and password attacks differ from the normal patterns of R1, R2, and R3. Also, how the RCE and MITM attacks shape an anomalous pattern in terms of memory usage compared to the normal performance can be seen.

In general, based on Table 18 and Figure 50, it becomes apparent that the memory parameter (M) fulfills the requirements to be a parameter that effectively detects cyberattacks in terms of anomalies and deviations from the normal range. Departing from the standard range requires an anomaly in the range of consumption. However, if consumption does not deviate from the defined standard range (Std_r), the anomalies in the patterns will be the vital factor in determining the feasibility of the parameter to be used to detect cyberattacks,

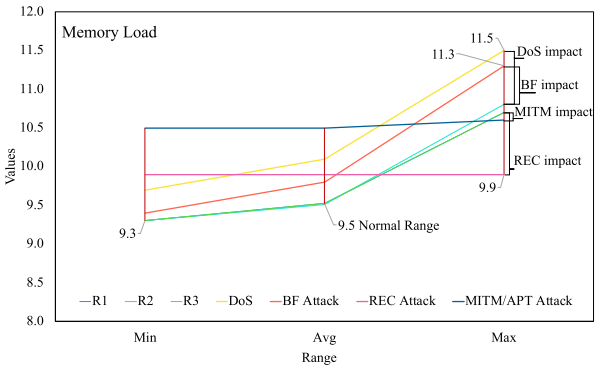


FIGURE 50. Overall memory ranges for attacked cases vs standard cases.

and this is what happened in the RCE and MITM attacked cases.

Also, it is worth noting that the more coherent the typical consumption value is, the more precise a pattern is formed, from which the anomalous patterns are easily distinguished and vice versa. The more widespread the consumption values are, the more the pattern of attacks needs to be clearly anomalous to distinguish it. Therefore, when the mode values of Rs cases were identical precisely at 9.5%, any minor deviation would be enough to distinguish the anomalous pattern.

C. TASKS AND THREADS PARAMETERS

The numbers of alive tasks and threads are among the parameters of the testbed. Based on the analysis results in terms of executed tasks number, it has been precisely proven that the range and pattern of the regular performance of a Smart PiCar can be determined based on the collected values during repeated runs of the system's functions. As shown in Table 19, the normal ranges range between 173 tasks as the maximal value and 159 tasks as the minimal value. Also, the mean value (Std_{avgV}) is 168.2 tasks as the average of R1, R2, and R3. In addition, the median values (T_{midV}) were between congruence and similarity. Moreover, the standard deviation values were closely converging compared to the attacked cases. The following table summarizes these results.

TABLE 19. Overall tasks number for all cases.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
R1	167.90	172.00	172.00	4.52	14.00	159.00	173.00
R2	168.81	171.00	172.00	3.61	11.00	162.00	173.00
R3	168.15	172.00	172.00	4.17	13.00	160.00	173.00
DoS	**181.64**	**184.00**	**171.00**	**9.39**	**33.00**	**166.00**	**199.00**
BF	**189.10**	**180.00**	**180.00**	**17.47**	**50.00**	**163.00**	**213.00**
RCE	**167.45**	**168.00**	**169.00**	**2.44**	**11.00**	**162.00**	**173.00**
MITM	**171.79**	**173.00**	**173.00**	**1.70**	**10.00**	**167.00**	**177.00**

In terms of pattern, the most frequent value or the density value was identical in R1, R2, and R3, which is 172 processes. This significant similarity in general and identicality, in particular in terms of the most common values and maximum values, demonstrate to us the possibility of defining the normal range and pattern of tasks count that is supposed to be executed while the system performs its function under normal conditions. However, the tasks number values under the circumstances of cyberattacks with their anomalous patterns

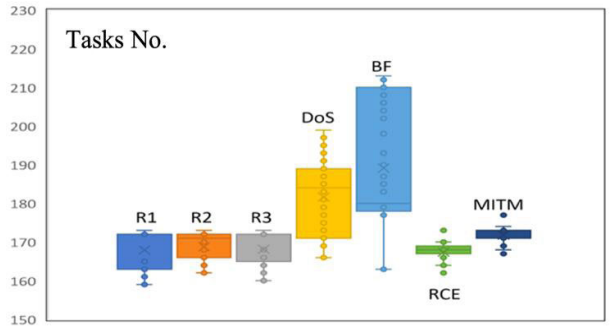


FIGURE 51. Overall tasks ranges for attacked cases vs standard cases.

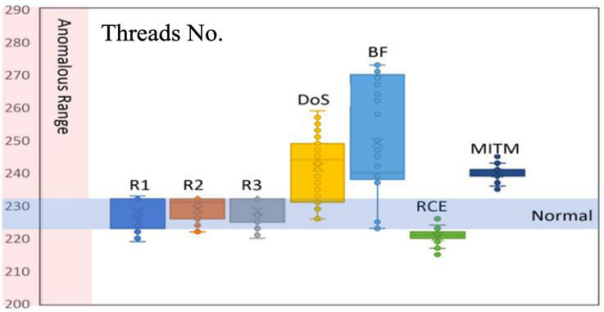


FIGURE 52. Overall threads ranges for attacked cases vs standard cases.

TABLE 20. Overall threads number for all cases.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
R1	228.01	232	232	4.48	14	219	233
R2	228.93	231	232	3.61	12	222	234
R3	228.28	232	232	4.18	14	220	234
DoS	**241.76**	**244**	**249**	**9.39**	**33**	**226**	**259**
BF	**249.23**	**240**	**240**	**17.52**	**50**	**223**	**273**
RCE	**220.56**	**221**	**221**	**2.49**	**11**	**215**	**226**
MITM	**239.89**	**241**	**241**	**1.78**	**10**	**235**	**245**

proves to us the effectiveness of this approach in detecting cyberattacks. Figure 51 illustrates the significant convergence in the pattern of tasks number in R1, R2, and R3 compared to attacked cases.

In the DoS attack scenario, the number of executed tasks reached 199 tasks. However, in the dictionary attack case, the number of executed tasks reached 213, which is an unprecedented value that caused an abnormality in the performance pattern compared to Std_p and significantly deviated from the standard range (Std_r), as was also the case with the DoS attack.

In the RCE attack case, it has been noticed that the number of executed tasks did not significantly deviate out of the normal range, but the pattern was odd compared to the normal pattern in R1, R2, and R3. The most frequently occurring value was 169 tasks, the average was 167.4 tasks, and these two values differ from the normal pattern (Section V, Table 12). Thus, the pattern has become anomalous in comparison to the normal pattern. Furthermore, taking into account the impacts of other parameters will clarify the distinguishability in this situation. The same applies to the MITM attack, where the highest value was slightly outside the normal range but the anomaly in terms of mean, minimum and maximum values supported the possibility of detecting the cyberattack.

In general, the results of the analysis criteria proved the accuracy of the defined standard model in terms of tasks number under normal operating conditions. Also, the obtained results under the circumstances of various cyberattack scenarios proved the ability to detect the attacks due to their abnormalities and deviations from the typical performance.

Regarding the number of threads, the general trait is similar to the tasks number parameter because it is linked to it.

As shown in Figure 52, the normal range is demonstrated within the blue area, starting from 219 threads to 234 threads as the maximum value (see Table 20). Also, the normal pattern concentrated identically in 232 threads for R1, R2, and R3, shaping the Std_p in terms of ϑ_r.

Thus, these results have proved the ability to determine the normal range, which has been used to generate the *Std* model and pattern for the threads number. In contrast, the attacked cases deviated from the normal range, and with its most common value (mode) at 249, 240, 221, and 241 for DoS, BF, RCE, and MITM, respectively. The anomalous behaviors have been monitored in comparison to the normal pattern, with the anomalous values appearing along the red zone, having more than 234 threads or less than 219 threads as shown in Figure 52. The following table summarizes the final results of the analysis criteria under all various conditions.

D. CPU TEMPERATURE

The reliance on CPU temperature to predict the existence of a cyberattack was effective in some cases and not sufficient in other cases. However, by looking at the maximal values shown in Table 21, the impact of cyberattacks that took the CPU temperatures to unprecedented levels compared to typical patterns can be observed. In general, attacks that consume more resources caused an increase in temperatures, while attacks that disrupted the system's functions interfered with the typical CPU temperature patterns. Therefore, the effectiveness of this parameter remains effective when combined with the effects of other parameters. Figure 53 depicts the temperature difference in the attacked cases from those in the conventional cases. Also, the overlap of the lower limits of the attacked cases' CPU temperatures with the conventional cases' upper limits is presented in this figure.

TABLE 21. Overall CPU temp number for all cases.

	Pattern				Range (XgV)		
	Mean	Median	Mode	Std.D	Range	Min	Max
R1	53.45	54.00	54.00	1.24	5.00	50.00	55.00
R2	46.60	47.00	50.00	3.32	14.00	38.00	52.00
R3	49.34	50.00	51.00	2.45	10.00	43.00	53.00
DoS	**61.57**	**62.00**	**64.00**	**2.72**	**10.00**	**55.00**	**65.00**
BF	**58.11**	**59.00**	**60.00**	**3.29**	**13.00**	**50.00**	**63.00**
RCE	**54.63**	**54.00**	**54.00**	**1.95**	**8.00**	**52.00**	**60.00**
MITM	**56.15**	**56.00**	**55.00**	**1.34**	**6.00**	**54.00**	**60.00**

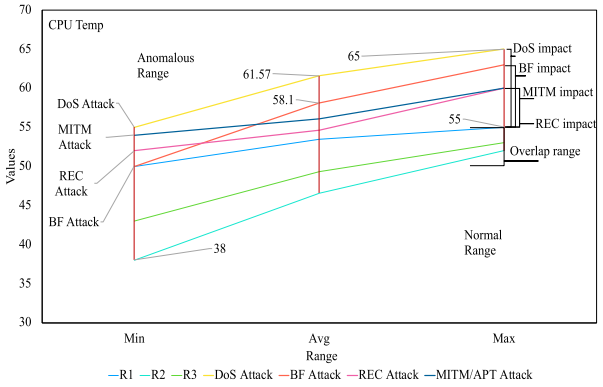


FIGURE 53. Overall temp ranges for attacked cases vs standard cases.

E. THE VOLTAGE AND CURRENT PARAMETER

The values of voltage and current have been monitored during the testbed and while the system performs its functions. In terms of range and pattern, R1, R2, and R3 are very similar and several values are often identical in terms of range or pattern as shown in Tables 22 and 23. For example, the standard deviation values in terms of voltage were identical in Rs cases. Therefore, defining the normal range and power consumption pattern is possible due to the identicality and significant similarity of the consumption.

For the attacked cases, in terms of the voltage parameter, the DoS attacks, BF, and RCE clearly deviate out of the normal range (XgV), as shown in Figure 54.

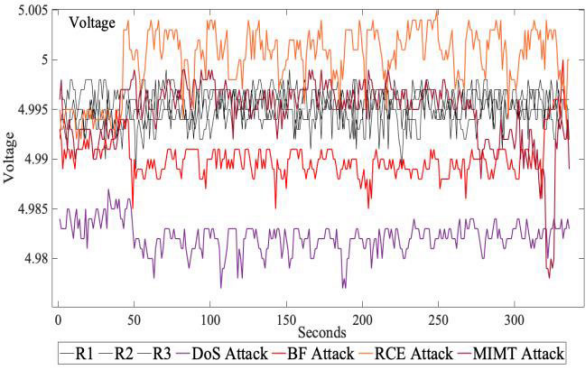


FIGURE 54. Overall voltage patterns for attacked cases vs standard cases.

In the MITM attack case, the power consumption in terms of voltage differs slightly from the normal range but is clearer in terms of pattern. Thus, relying on this parameter separately to detect an attack is not enough. However, in this situation the impacts of other parameters will support the determination of a cyberattack's existence. The following table summarizes the results of power consumption in terms of voltage for all cases.

In terms of the current-voltage sub-parameter, the general trait is the overlap and similarities in the range or pattern except in the RCE attack case. In the RCE scenario, the rate of power consumption significantly differed from the normal range and shaped an anomalous pattern due to the failure of its function because of the RCE attack. Figure 55 demonstrates the P rate in terms of I_r and I_p in all cases as summarized in the following table.

TABLE 22. Overall voltage consumption for all cases.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
R1	4.995	4.995	4.99	0.001	0.01	4.99	5
R2	4.996	4.996	5	0.001	0.01	4.99	5
R3	4.995	4.995	5	0.001	0.01	4.99	5
DoS	**4.982**	**4.982**	**4.98**	**0.002**	**0.01**	**4.98**	**4.99**
BF	**4.99**	**4.99**	**4.99**	**0.002**	**0.01**	**4.99**	**5**
RCE	**5**	**5.001**	**5**	**0.003**	**0.01**	**4.99**	**5.01**
MITM	**4.995**	**4.996**	**5**	**0.003**	**0.02**	**4.98**	**5**

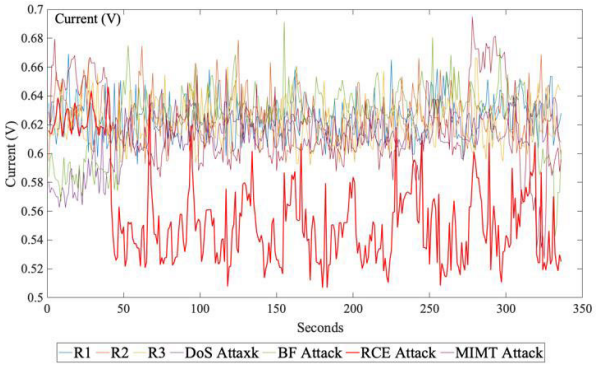


FIGURE 55. Overall current patterns for attacked cases vs standard cases.

TABLE 23. Overall current consumption for all cases.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
R1	0.624	0.623	0.61	0.01364	0.08	0.59	0.67
R2	0.625	0.625	0.62	0.01569	0.09	0.59	0.68
R3	0.62	0.618	0.61	0.01491	0.07	0.59	0.67
DoS	**0.612**	**0.613**	**0.61**	**0.01638**	**0.08**	**0.56**	**0.64**
BF	**0.626**	**0.631**	**0.63**	**0.02249**	**0.15**	**0.54**	**0.69**
RCE	**0.557**	**0.551**	**0.55**	**0.03347**	**0.14**	**0.51**	**0.65**
MITM	**0.617**	**0.611**	**0.61**	**0.02729**	**0.17**	**0.53**	**0.69**

F. RECEIVED AND TRANSMITTED PACKETS PARAMETERS

The average of transmitted and received packets has been monitored during the RCE and MITM attacks as one of the testbed parameters. Based on the results in Table 24 and Figure 56, it is possible to define the normal range and patterns of the typical rate of received packets as we did in Section V because of the huge convergence in terms of packets count and due to its similarity in the performance pattern, which was sufficient to generate the *Std*. The range and pattern of R2 and R3 are almost identical in many aspects and very close to R1's range and pattern. Based on the that, the RX_r ranges from 0 to 27 packets, with 0 the most frequent value and 4.36 to 4.997 the average of received packets.

In the attacked cases, for the RCE attack case there is no significant difference in terms of received packets. However, in the MITM attack case, the rate of received packets per second has differed from the normal range. Thus, this sort of parameter might help in detecting specific kinds of attacks such as a network-based attack, where the attacks continuously receive data from the victim's device. The following

TABLE 24. Overall received packets for all cases.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Mini	Max
R1	4.04	1.00	0.00	5.28	24.00	0.00	24.00
R2	4.29	1.00	0.00	5.72	27.00	0.00	27.00
R3	5.00	2.00	0.00	6.02	27.00	0.00	27.00
RCE	**3.99**	**1.00**	**0.00**	**5.26**	**27.00**	**0.00**	**27.00**
MITM	**7.79**	**3.00**	**0.00**	**10.37**	**55.00**	**0.00**	**55.00**

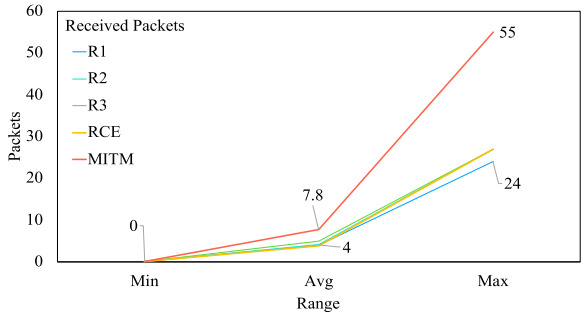


FIGURE 56. Overall received packets rates patterns for MITM attack vs standard cases.

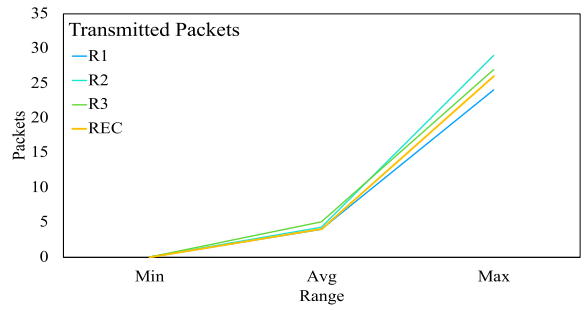


FIGURE 57. Overall transmitted packets rates patterns for RCE attack vs standard cases.

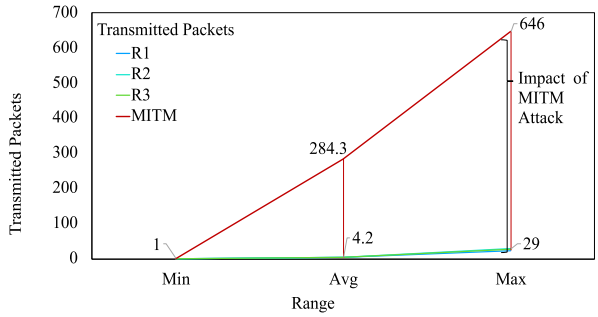


FIGURE 58. Overall transmitted packets rates patterns for MITM attack vs standard cases.

figure shows the impact of RCE attacks and MITM on the volume of received packets/s.

In terms of transmitted packets, the R1, R2, and R3 showed a sufficient convergence to determine the normal range and pattern. The normal range of the transmitted packets rate extended from 0 to 29 packets per second and the most common value was 0. Also, the average was between 4 to 5 packets per second. In the attacked cases, the general trait was similar to the received packets rate parameter. As shown in Figures 57 and 58, the RCE attack (yellow line) does

TABLE 25. Overall transmitted packets for all cases.

	Pattern				Range		
	Mean	Median	Mode	Std.D	Range	Min	Max
R1	4.03	1.00	0.00	5.27	24.00	0.00	24.00
R2	4.29	1.00	0.00	5.72	29.00	0.00	29.00
R3	5.05	2.00	0.00	6.03	27.00	0.00	27.00
RCE	**4.06**	**1.00**	**0.00**	**5.29**	**26.00**	**0.00**	**26.00**
MITM	**284.29**	**307.00**	**1.00**	**148.68**	**645.00**	**1.00**	**646.00**

not differ significantly from the normal range as the MITM attack shown in Figure 58, where even the pattern was somehow similar to R1, R2, and R3. Thus, the impacts of other parameters must be added in order to facilitate the separation ability in this case. In contrast, the TX parameter in the MITM case deviated significantly from the normal range, causing an anomalous pattern as shown in Figure 58. Therefore, having this parameter within the septenary tuple model will help detect specific kinds of attacks. The following table summarizes the final analytics results of the transmitted packets/s.

The work in this paper is mainly concentrated on the feasibility of detecting anomaly patterns using the proposed approach, which is illustrated from the presented results, but detection latency is also a critical factor in demonstrating its effectiveness. In our suggested approach, the detection latency is closely related to the determination of the anomaly threshold, and the anomaly threshold depends on the formation of the abnormal pattern in performance. Therefore, according to our proposed approach, which is based on monitoring the values of specific parameters representing the system performance and resource consumption in real-time, the time taken to detect the attack is the total time of: monitoring, analyzing and anomalous pattern formation. Thus, another vital factor related to the detection latency, and which plays an important role, is the time taken for the abnormal pattern to form and for its effect on system performance and resource consumption to appear.

In other words, there is no problem in detecting the attack; as soon as there is an anomalous pattern, the attack will be detected, but how long does it take for the abnormal pattern to form?. To find out, the time-series of two attacked scenarios have been revised: DoS and password attack, and relevant insights have been presented. Based on the results shown in Table 26, and Table 27, the DoS attack was executed at 00:26:25, and the anomalous pattern started to form at 00:26:26.

Also, at 00:16:16s, the password attack was executed, and at 00:16:18s, the anomalous pattern began to appear (Appendix I: snapshots of real-time observation). Therefore, detecting attacks based on this approach is an efficient method. However, the formation of the anomalous pattern is also related to other factors that might have an influence, i.e., the type of attack, the architecture of the operating system, and the computing capabilities of the ES.

Although all executed attack scenarios caused anomalous performance patterns, this approach is active in terms of detecting the attack existence and not proactive in terms

TABLE 26. Timestamps during DoS attack.

T. Stamp	Temp	CPU	Mem	Tasks	Threads	RX	TX
00:26:16	60	19	10.3	167	228	11	12
00:26:17	60	17.1	10.3	167	227	12	10
00:26:18	61	22	10.3	167	228	11	10
00:26:19	60	23.3	10.3	167	227	11	11
00:26:20	60	21.6	10.3	167	227	12	10
00:26:21	60	17.3	10.3	167	228	11	10
00:26:22	59	17.7	10.3	167	227	11	19
00:26:23	61	17.9	10.3	167	227	11	13
00:26:24	60	16.9	10.3	167	227	11	10
00:26:25	**60**	**16.3**	**10.3**	**167**	**227**	**13**	**11**
00:26:26	62	65.4	11.3	167	227	1449	700
00:26:27	60	47.4	11.9	261	326	4737	2974
00:26:28	60	23.9	11.9	267	327	2308	1575
00:26:29	60	21.3	11.9	267	327	1675	930
00:26:30	60	17.6	11.9	267	328	28	30

Time-series of DoS attack

00:26:24	Normal Pattern.
00:26:25	The attack was launched.
00:26:26	The threshold of anomalies, (first outlier values).
00:26:27	Anomalous pattern formation

TABLE 27. Timestamps during password attack.

T. Stamp	Temp	CPU	Mem	Tasks	Threads	RX	TX
00:16:08	55	7.1	9.9	172	232	2	2
00:16:09	55	6.8	9.9	172	232	1	0
00:16:10	56	7.9	9.9	172	233	2	2
00:16:11	56	8.5	10.3	172	232	1	1
00:16:12	55	10.6	10.7	173	233	2	2
00:16:13	56	8.1	10.3	173	233	1	0
00:16:14	56	7.5	9.9	173	233	2	1
00:16:15	57	13	9.9	173	234	1	0
00:16:16	56	13	9.9	173	233	1	1
00:16:17	57	14.6	9.9	173	233	15	12
00:16:18	57	47	10.3	173	233	216	211
00:16:18	57	5.6	10.3	205	265	2	2
00:16:19	57	7.9	10.3	203	263	1	0
00:16:20	55	10.4	10.3	203	263	2	1
00:16:21	57	7.2	10.3	203	263	1	0

Time-series of DoS attack

00:16:15	Normal Pattern.
00:16:16	The attack was launched.
00:16:18	The threshold of anomalies, (first outlier values).
00:16:18	Anomalous pattern formation

of preventing the attack occurrence. Thus, it is better not to rely on the anomaly detection technique independently but also to initially preserve the predefined standard performance by applying the necessary restrictions on resource consumption. Usually, cyberattacks need an entry point, thereby living on system resources to pass on their harmful instructions, and this will be reflected negatively on the system performance. Therefore, the cyberattacks will be efficiently detected if a reference pattern has been defined. Accordingly, it is expected that other untested types of attacks will cause irregular performance patterns. It will be possible to detect them based on this approach, especially since the septenary tuple model, which represents the core of the detection mechanism, includes seven parameters considering the ES's architecture. Thus, this detection mechanism is capable of monitoring the most important changes in system resources comprehensively. However, it is beneficial to examine more attack types to find out if there is an attack that does not have a noticeable impact on the system's performance; as this will not be detected according to this technique.

Returning to the starting point from which we started (the hypothesis of the study) and based on our final results, we can confirm the validity of the hypothesis where cyberattacks against embedded systems can be detected according to our proposed approach. By delivering this approach, we have paved the way for software engineers to augment OS robustness by adding a protective layer to prevent cyberattacks based on a new perspective, which is the statistical analysis of the septenary tuple model.

VIII. CONCLUSION AND FUTURE WORK

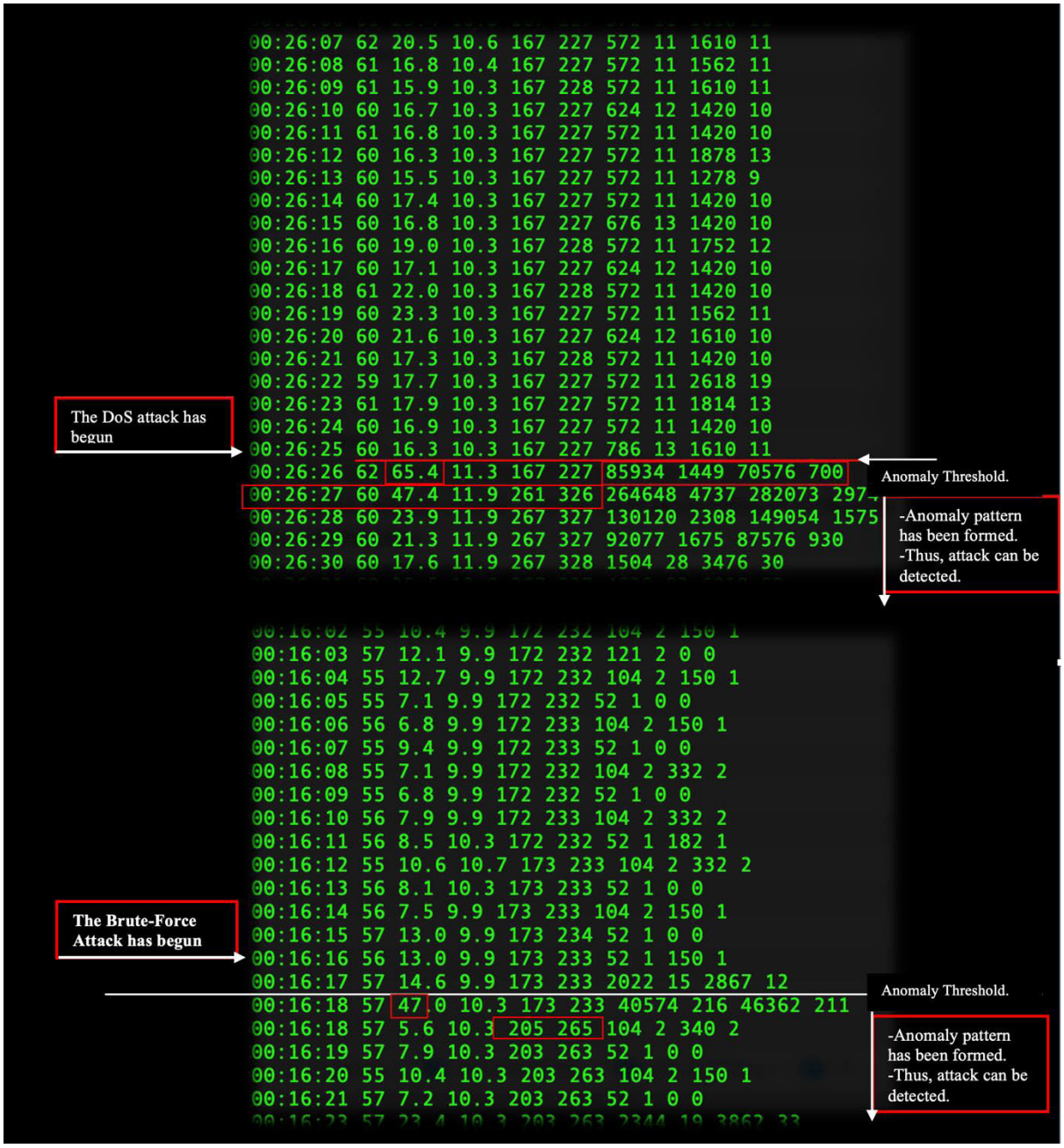
Embedded systems differ from typical computer systems given that they are designed to perform specific functions. In this context, we proposed a new security approach for detecting cyberattacks based on the recognition of anomalous resource consumption and performance patterns of an embedded system. In particular, a septenary tuple of a parameter model, consisting of CPU utilization, memory load, tasks, threads, data traffic rate, temperature, and power consumption, is presented as the detection mechanism. The data has been collected through the repetitive operation of a test model, i.e., Smart PiCar, where the collected data has been analyzed in terms of seven statistical criteria, i.e., mean, median, mode, standard deviation, range, minimum, and maximum. The final result illustrated there exists a pattern of resource consumption and performance of embedded systems that can be defined as a standard pattern due to a distinctive feature (ESs are dedicated to performing specific functions), providing a baseline on which the pattern of normal states of an embedded system can be determined and utilized as a reference profile to detect anomalous patterns.

In contrast, the anomalous patterns of the Smart PiCar performance under different cyberattacks (denial-of-service, password, remote code execution, and man-in-the-middle) have clearly appeared and been observed. These anomalous patterns differ from the typical normal pattern.

The experimental results prove the efficacy of the ARCD approach. Thus, this approach enables software engineers to improve the OS layer of the ESs. By creating a recognition mechanism against cyberattacks based on resource consumption and performance patterns, the OS layer of an ES could secure regular consumption and respond to cyberattacks once an anomalous pattern occurs. Moreover, for the adoption of ARCD by practicing software engineers, an additional layer of protection can be added to the embedded system's OS.

Furthermore, the raw data enabled us to develop a machine learning model by using the support vector machine algorithm to determine the prediction accuracy percentage in terms of separating anomalous patterns from typical patterns, forming our future task. The outcomes of computational validation based on machine learning algorithms will be presented in a forthcoming paper, including an architecture framework facilitating this approach adoption for software engineers.

APPENDIX



*Snapshots of real-time observation

REFERENCES

[1] P. Marwedel, *Embedded System Design*. Cham, Switzerland: Springer, 2018.

[2] P. I. R. Grammatikis, P. G. Sarigiannidis, and I. D. Moscholiosb, "Securing the Internet of Things: Challenges, threats and solutions," *Internet Things*, vol. 5, pp. 41–70, Mar. 2019, doi: 10.1016/j.iot.2018.11.003.

[3] W. Wang, X. Zhang, Q. Hao, Z. Zhang, B. Xu, H. Dong, T. Xia, and X. Wang, "Hardware-enhanced protection for the runtime data security in embedded systems," *Electronics*, vol. 8, no. 1, p. 52, Jan. 2019, doi: 10.3390/electronics8010052.

[4] D. Papp, Z. Ma, and L. Buttyan, "Embedded systems security: Threats, vulnerabilities, and attack taxonomy," in *Proc. 13th Annu. Conf. Privacy, Secur. Trust (PST)*, Jul. 2015, pp. 145–152, doi: 10.1109/PST.2015.7232966.

[5] H. Chai, G. Zhang, J. Zhou, J. Sun, L. Huang, and T. Wang, "A short review of security-aware techniques in real-time embedded systems," *J. Circuits, Syst. Comput.*, vol. 28, no. 2, Feb. 2019, Art. no. 1930002, doi: 10.1142/S0218126619300022.

[6] A. R. Sfar, E. Natalizio, Y. Challal, and Z. Chtourou, "A roadmap for security challenges in the Internet of Things," *Digit. Commun. Netw.*, vol. 4, no. 2, pp. 118–137, 2018, doi: 10.1016/j.dcan.2017.04.003.

[7] H. Habibzadeh, B. H. Nussbaum, F. Anjomshoa, B. Kantarci, and T. Soyata, "A survey on cybersecurity, data privacy, and policy issues in cyber-physical system deployments in smart cities," *Sustain. Cities Soc.*, vol. 50, Oct. 2019, Art. no. 101660, doi: 10.1016/j.scs.2019.101660.

[8] A. Humayed, J. Lin, F. Li, and B. Luo, "Cyber-physical systems security—A survey," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1802–1831, Dec. 2017, doi: 10.1109/JIOT.2017.2703172.

[9] S. Ravi, P. Kocher, R. Lee, G. McGraw, and A. Raghunathan, "Security as a new dimension in embedded system design," in *Proc. 41st Annu. Conf. Design Autom. (DAC)*, 2004, p. 753, doi: 10.1145/996566.996771.
[10] C. Bodei, S. Chessa, and L. Galletta, "Measuring security in IoT communications," *Theor. Comput. Sci.*, vol. 764, pp. 100–124, Apr. 2019, doi: 10.1016/j.tcs.2018.12.002.
[11] S. Hameed, F. I. Khan, and B. Hameed, "Understanding security requirements and challenges in Internet of Things (IoT): A review," *J. Comput. Netw. Commun.*, vol. 2019, pp. 1–14, Jan. 2019, doi: 10.1155/2019/9629381.
[12] S. Ali, T. Al Balushi, Z. Nadir, and O. K. Hussain, *Cyber Security for Cyber Physical Systems*, vol. 768. Cham, Switzerland: Springer, 2018, pp. 11–33, doi: 10.1007/978-3-319-75880-0_2.
[13] A. Aloseel, H. He, C. Shaw, and M. A. Khan, "Analytical review of cybersecurity for embedded systems," *IEEE Access*, vol. 9, pp. 961–982, 2021, doi: 10.1109/ACCESS.2020.3045972.
[14] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, "Network intrusion detection," *IEEE Netw.*, vol. 8, no. 3, pp. 26–41, May 1994, doi: 10.1109/65.283931.
[15] A. Vikram, "Anomaly detection in network traffic using unsupervised machine learning approach," in *Proc. 5th Int. Conf. Commun. Electron. Syst. (ICCES)*, Jun. 2020, pp. 476–479, doi: 10.1109/ICCES48766.2020.9137987.
[16] S. Singh and S. Banerjee, "Machine learning mechanisms for network anomaly detection system: A review," in *Proc. Int. Conf. Commun. Signal Process. (ICCSP)*, Jul. 2020, pp. 0976–0980, doi: 10.1109/ICCSP48568.2020.9182197.
[17] P. J. Fortier and H. E. Michel, "Introduction," in *Computer Systems Performance Evaluation and Prediction*. Amsterdam, The Netherlands: Elsevier, 2003, pp. 1–38.
[18] T. W. Edgar and D. O. Manz, "Instrumentation," in *Research Methods for Cyber Security*. Amsterdam, The Netherlands: Elsevier, 2017, pp. 321–344.
[19] P. J. Fortier and H. E. Michel, "Hardware testbeds, instrumentation, measurement, data extraction, and analysis," in *Computer Systems Performance Evaluation and Prediction*. Amsterdam, The Netherlands: Elsevier, 2003, pp. 305–330.
[20] *Psutil Documentation, Psutil, 2009–2021*. Accessed: Apr. 25, 2021. [Online]. Available: https://psutil.readthedocs.io/en/latest/
[21] R. Love, *Linux Kernel Development*, 3rd ed. Reading, MA, USA: Addison-Wesley, 2010, p. 33.
[22] *The Linux Man-Pages Project, Hkernel.Org Doc Man-Pages 2021*. Accessed: Apr. 25, 2021. [Online]. Available: https://www.kernel.org/doc/man-pages/
[23] *Frequency Management and Thermal Control, Raspberry Pi Foundation*. Accessed: Apr. 25, 2021. [Online]. Available: https://www.raspberrypi.org/documentation/hardware/raspberrypi/frequency-management.md
[24] (2021). *Python 3.9.4 Documentation,Python.Org, 2021*. Accessed: Apr. 25, 2021. [Online]. Available: https://docs.python.org/3/
[25] (2020). *Understanding Denial-of-Service Attacks, Us-Cert.Cisa.Gov, November 04, 2009*. Accessed: Apr. 25, 2021. [Online]. Available: https://us-cert.cisa.gov/ncas/tips/ST04-015
[26] H. C. A. van Tilborg and S. Jajodia, Eds., *Encyclopedia of Cryptography and Security*. Boston, MA, USA: Springer, 2011.
[27] S. Biswas, M. Sohel, M. M. Sajal, T. Afrin, T. Bhuiyan, and M. M. Hassan, "A study on remote code execution vulnerability in web applications," in *Proc. Int. Conf. Cyber Secur. Comput. Sci.*, Oct. 2018, pp. 1–8.
[28] E. Fazeldehkordi, O. A. Akanbi, A. Study, and H. Attack, *A Study of Black Hole Attack Solutions*. Amsterdam, The Netherlands: Elsevier, 2016.

ABDULMOHSAN ALOSEEL received the B.S. degree in management information systems (MIS) from King Faisal University, in 2006, and the M.S. degree in computer and network security from Middlesex University, London, in 2015. He is currently pursuing the Ph.D. degree in cybersecurity of embedded systems with the School of Aerospace, Transport and Manufacturing (SATM), Cranfield University.

SABA AL-RUBAYE (Senior Member, IEEE) received the Ph.D. degree in electrical and electronic engineering from Brunel University London, U.K. She is currently a Senior Lecturer, a DRATeC Fellow, and leading connected system research group with the School of Aerospace, Transport and Manufacturing, Cranfield University, U.K. She is participating in developing industry standards by being an Active Voting Member of IEEE P1920.2, standard for Vehicle-to-Vehicle Communications for Unmanned Aircraft Systems and IEEE P1932.1 standard of License/Unlicensed Interoperability. She has published many papers in IEEE journals and conferences and a recipient of the Best Technical Paper Award twice published in IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, in 2011 and 2015, respectively. Her main research interests include, but not limited to UAV connectivity, communication networks, artificial intelligence, and safety and security of autonomous vehicle. She is a Chartered Engineer (C.Eng.), a member of IET, and a Certified Unmanned Aircraft System (UAS) Pilot. She has been the general co-chair, the TPC co-chair, and has held other leading roles for many international conferences. She has organised and chaired the 6G Network Workshop in IEEEICC2020.

ARGYRIOS ZOLOTAS (Senior Member, IEEE) received the B.Eng. degree (Hons.) from the University of Leeds, the M.Sc. degree (Hons.) from the University of Leicester, and the Ph.D. degree from Loughborough University. He was with the University of Lincoln, the University of Sussex, Loughborough University, and Imperial College London. He was a Visiting Professor with Grenoble INP, from May 2018 to June 2018. He is currently a Reader with Cranfield University. His research interests include advanced control, systems autonomy, and applied AI. He is a fellow of HEA.

HONGMEI HE (Senior Member, IEEE) received the Ph.D. degree in computer science from Loughborough University, in 2006. She had rich postdoctoral experience at different universities, such as the University of Bristol, from January 2007 to March 2011, the Ulster University of Ulster, from April 2011 to December 2011, and the University of Kent, from January 2012 to October 2013. She was a Lecturer in AI and cybersecurity with Cranfield University, from October 2013 to November 2020. Before coming to the U.K., she was a Senior Embedded System Engineer at the Motorola Design House, Shenzhen, China. She is currently a Senior Lecturer with the School of Computer Science and Informatics, De Montfort University. Her current research interests include AI, covering AI for cognitive cybersecurity, and safety and security of autonomous systems. She is a Working Group Member of IEEE Technical Ethics P7000 Standard. She actively serves as the Chapter Secretary for the IEEE U.K. and Ireland RAS Chapter. She is the Chair of the task force, "Artificial Intelligence and Edge Computing for Trustworthy Robots and Autonomous Systems," at the Adaptive Dynamic Programming and Reinforcement Learning Technic Committee (ADPRLTC) of IEEE Computational Intelligence Society.

CARL SHAW received the Ph.D. degree in physics from Queen's University Belfast, in 1997. He then worked in U.K. defense before leaving for the private sector, where he worked in the semiconductor industry at STMicroelectronics. Throughout this period, he worked on the electronic design, system architecture, and software of embedded systems. For the last 16 years, he has been active in software and hardware security. He is currently a Co-Founder of Cerberus Security Laboratories Ltd., a U.K. security consultancy, where he advises global multinationals on electronic product cybersecurity and works closely with academic institutions researching secure hardware and embedded systems.