IEEE Open Journal of the Computer Society

Received 22 October 2024; revised 12 December 2024; accepted 14 December 2024. Date of publication 26 December 2024; date of current version 17 January 2025. The review of this article was arranged by Associate Editor Milos Stojmenovic.

Digital Object Identifier 10.1109/OJCS.2024.3522566

Energy Efficiency of Kernel and User Space Level VPN Solutions in AIoT Networks

ALEKSANDAR JEVREMOVIC [1] (Member, IEEE), ZONA KOSTIC [2] (Member, IEEE), IVAN CHORBEV [3] (Member, IEEE), DRAGAN PERAKOVIC [4] (Member, IEEE), ANDRII SHALAGINOV [5] (Member, IEEE), AND IVAN CVITIC [4]

[1]Faculty of Informatics and Computing, Singidunum University, 160622 Beograd, Serbia
[2]Harvard University, Cambridge, MA 02138 USA
[3]Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University, 1000 Skopje, North Macedonia
[4]Faculty of Transport and Traffic Sciences, University of Zagreb, 10000 Zagreb, Croatia
[5]Kristiania University College, 0107 Oslo, Norway

CORRESPONDING AUTHOR: ALEKSANDAR JEVREMOVIC (email: ajevremovic@ieee.org).

This work was supported in part by the ENViSEC project and in part by European Union's Horizon 2020 Research and Innovation programme within the framework of the NGI-POINTER Project under Grant 871528.

ABSTRACT The ability to process data locally using complex algorithms is becoming increasingly important in Internet of Things (IoT) contexts. Numerous factors contribute to this trend, including the requirement for immediate response, the need to protect data privacy/security, a lack of adequate infrastructure, and the desire to reduce costs. Due to the extensive hardware requirements (in terms of required computing power, memory, and other resources) for handling various scenarios, edge devices are typically configured to utilize general-purpose operating systems, primarily GNU/Linux. However, energy efficiency remains a critical requirement for this devices, especially in battery-powered scenarios (where energy inefficiency could make the device completely inoperable). Local data processing usually minimizes, but not entirely eliminates, data exchange with the environment. Along with energy costs of data processing, it is critical to also consider the energy efficiency of data protection when communicating with the environment. In this article, we evaluate the energy efficiency of kernel-level and user-space-level communication protection solutions: WireGuard and OpenSSL. These systems are evaluated on a range of hardware platforms, including Raspberry Pi 3, Nvidia Jetson NANO, Nvidia Jetson TX2, and Nvidia Jetson AGX Xavier. The energy efficiency of these systems was determined by examining long transfer streams with maximum channel/CPU utilization. We discovered that determining the energy efficiency of a device or protocol is difficult due to the high reliance on factors such as communication speed and direction.

INDEX TERMS Internet of Things (IoT), AIoT, WireGuard, OpenSSL, energy efficiency.

I. INTRODUCTION

AIoT, also known as Artificial Intelligence on the IoT infrastructure, is comprised of edge node nodes linked to a distributed intelligent environment. By combining distributed computational power with artificial intelligence, this new technology can be applied in a variety of environments. However, communication between devices (edge, gateways, and/or cloud) is prone to a range of cyberattacks [1]. As a result, the research community is constantly on the lookout for more secure methods of connecting network nodes, particularly when sensitive data transfer is involved. To fully exploit the AIoT environment, edge devices must operate efficiently and consume the least amount of energy possible [2].

By connecting edge devices to IoT networks, massive amounts of data can be exchanged between nodes or with other services. While the large volume of data transmitted across the network is necessary for an AI model to perform accurately, adding an additional layer of secure data transmission mechanisms may cause the communication channel to become overburdened. Transferring highly sensitive data

between devices (for example, streams of medical records from multiple patients), which typically involve large amounts of unstructured data such as images or videos, is one of the transmission scenarios that is prone to a range of security attacks. While there are numerous algorithms for protecting sensitive data and ensuring its secure transmission [3], not all of them support efficient data transfer while minimizing power consumption between edge devices.

As a result, the primary concerns for edge devices that communicate with one another continue to be privacy and energy consumption [4]. While various encryption algorithms enable secure data processing to be offloaded, this requires a substantial amount of bandwidth between devices. Despite their stated goal of disabling encryption release on low-resource devices, client-assisted encrypted systems on the edge are unable to manage node efficiency adequately [5]. Historically, the energy capabilities of edge devices that generate large amounts of data, particularly those that are battery-powered, have been limited [6], elevating privacy concerns.

Correspondingly, ensuring the stability and integrity of communication becomes critical for minimizing node energy consumption while defending against potential security threats. Edge Intelligence [7] which traditionally relied on cloud resources, may not be suitable for scaling in heterogeneous environments. For instance, in environments where machine learning is used to adapt dynamically to changing conditions and behaviors [8], or in other scenarios involving a variety of devices, connection failures, limited battery life, and varying processing capacities. Thus, it is critical to understand the trade-offs between AI efficiency, energy consumption, and the required level of security for various edge devices as well as the communication between them. This trade-off may serve as a starting point for developing a mechanism for balancing energy consumption and security [1].

Energy efficiency is becoming an increasingly important consideration for battery-powered edge devices, and enhancing data security incurs significant communication costs. It is critical to support robust AI models while also considering how power consumption is affected by security level. Understanding trade-offs could enable all devices on the network to make an informed decision about the optimal combination of AI model, data, and security measures for low-power edge devices [1]. Numerous significant IoT vulnerabilities exist [9], the majority of which are related to insecure network services, such as denial-of-service attacks and a lack of transport encryption/integrity verification. In this article, we evaluate the energy efficiency of kernel-level (WireGuard) and user space-level (OpenSSL) VPN solutions. These were evaluated on a variety of AIoT devices, including Raspberry Pi 3, Nvidia Jetson NANO, Nvidia Jetson TX2, and Nvidia Jetson AGX Xavier. The energy efficiency of these systems was determined using long transfer streams with maximum channel/CPU utilization. Measured energy efficiency is presented as the number of consumed milliwatts per transferred megabyte of data.

II. RELATED WORK

A. AIOT AND SECURITY

AIoT devices, and the ecosystem they can create, are a real game-changer in the era of boosted digitalization, green shift and automation. We can see a growing number of various platforms that are suitable for building versatile ad-hoc IoT solutions ranging from single-board *microcontrollers* (or *IoT nodes*) such as Arduino and to a single-board portable *microcomputer* (or *IoT gateways*) such as Raspberry Pi, Nvidia Jetson and others. Moreover, IoT devices may gather information from a wide range of *sensors* that further need to be fused and used for decision making. *Actuators* are available to operate in a physical world and respond to sensors' measurements. The *microcontrollers* usually include KBytes of SRAM, tens of MHz of CPU and consume sub-Watt energy levels. There is no possibility to utilise conventional cybersecurity protection mechanisms such as Intrusion Detection Systems or Anti-Virus for attacks detection.

However, there exist proof-of-concept demonstrations of cryptographic functionality that can be used for hardening data-in-transfer, and data-at-rest [10], [11]. On the contrary, *microcomputers* are more powerful devices and are capable of computations comparable to modern full scale personal computers. In this article, the authors analyse the energy consumption consideration for the microcomputer-level devices when it comes to the application of cryptographic algorithms for data-in-transit and data-at-rest.

IoT-based infrastructure has become a dominant way of organising smart applications, especially with Artificial Intelligence (AI) models for everyday life. Until recently, there have been very few studies related to the actual cost of cybersecurity (in Watts) and energy consumption assessment when applying cryptography for data exchange protection. As a matter of fact, cryptography is a mandatory requirement for building any modern software, for either transmission of the data or storing the data on the device. Early versions of the IoT and smart devices did not include any viable data protection mechanisms. In 2017 Norwegian Consumer Council studied smartwatches in the report #WatchOut and found that only one out of four watches manufacturers promised the implementation of necessary security mechanisms. It is evident that the energy consumption considerations often come before security-by-design and privacy-by-design as an inseparable part of defence-in-depth [12]. Undoubtedly, there are more initiatives targeting standardisation of the IoT security in recent years, such that the implementation of the ETSI standard ETSI EN 303 645 [13] and the development of the Arduino Crypto Library [10]. However, the main challenge is cybersecurity's impact on the IoT microcomputer's performance and energy consumption. As mentioned, this paper's goal includes a range of AI capable IoT microcomputers. Therefore, there is a need to establish a reproducible experiment and energy consumption assessment to understand the cost of the power (in Watts) per MByte of encrypted data.

B. ENERGY AND PERFORMANCE EVALUATION IN COMPUTER AND IOT ECOSYSTEMS

To understand the actual consumption of the IoT, we need to go through the current state of the art used in the community. One of the measures is so-called computer architecture efficiency and is measured in Performance per Watt [14]. However, performance is a generic term used to describe various aspects of computer systems. Therefore, one commonly used measure of CPU performance on Linux OS is the number of internal busy-loops in so-called bogomips ("bogus" millions of instructions per second) [15]. Another way to measure it, applicable in this article, will be to evaluate the MBytes of process data per Watt of energy. There are tremendous differences between desktop machines and IoT platforms regarding this measure. A modern laptop with 11th Gen Intel Core i7-1165G7 (2.80 GHz clock speed) has eight cores (5606.40 bogomips each), while the latest Raspberry Pi 4 with Cortex-A72 CPU (1.5Ghz clock speed) has four cores (108 bogomips each). The first CPU consumes 28 Watts according to Intel specifications,[1] while the second - up to 10 Watts [16]. Therefore, an average IoT microcomputer is tens of times less energy efficient than a desktop or server station when it comes to computational capabilities. Nevertheless, modern AI-optimized microcomputers have integrated many optimisation techniques that will be explored during the experimental part of this article.

C. POWER CONSUMPTION OF CRYPTOGRAPHY ALGORITHMS

With the growing demand for IoT devices integrated into the conventional computer ecosystem, there is a definite need for energy efficiency in many aspects of the IoT platforms development. One of the game changes is Intel's work with more efficient crypto acceleration that can achieve three times faster performance while using Advanced Encryption Standard (AES) on the newer architecture [17]. In general, it demonstrates that computers can now perform computations faster by optimising the software and using parallel computing paradigms. However, the idea was there last decade, and the same Intel released recommendations regarding energy-efficient software design [18].

When it comes to IoT and cartographic algorithm, authors [19] raised concerns about privacy and untrusted cloud and edge ecosystem. To mitigate these issues, a novel algorithm was developed called Client-aided Homomorphic Encryption for Opaque Compute Offloading (CHOCO) that reduced the actual energy cost of the encryption in IoT by 648 times in comparison to existing modern encryption solutions. Additionally, it was suggested a novel scheme called Energy Efficient Distributed Lightweight Authentication and Encryption (EEDLAE) to efficiently optimised the energy footprint of the authentication on IoT [20]. Kane, Chen, Thomas, Liu, and McKague [21] have performed a comparison of the energy consumption for microcontrollers such that Atmega328 and EPS8266-based boards by using de-facto standard encryption algorithms AES, ChaCha and Acorn. It was discovered that the lower clock speed results in better energy efficiency, while a higher speed clock results in faster processing speed. Even though all of the evaluated devices had tens of MHz CPU clock speed, it was also discovered that there are considerable differences in the performance and energy consumption depending on the flash and RAM usage in the encryption process. Furthermore, Atmega-based Libelium Waspomote was used to study the optimal trade-off in AES configurations and demonstrated how the energy consumption and transmission distance change when turning on or off the encryption and using various key lengths [22].

Fotovvat, Rahman, Vedaei, and Wahid [2] focused on the evaluation of the lightweight cryptography (LWC) on more powerful IoT microcomputers, such as Raspberry Pi, and found that it is a good way of reducing the energy and computing cost of the encryption in communication between the sensor node and receiver in the IoT applications. It was also noted that the consumption of the power depends on the configuration of the platform, e.g. RPI-ZW and RPI-3B consume 420 mW and 2200 mW. With ADED algorithm execution, the power consumption jumps to nearly 700 mW and 2,700 mW, respectively. Therefore, the cost of the actual encryption will depend on the platform the methods are running on. Undoubted, there is a strong push towards development of the lightweight encryption and communication protocols and some of the authors [23] performed expensive evaluation on both IoT nodes and gateways, while others [24] looked at the differences between using MQTT (Message Queuing Telemetry Transport) and CoAP (Constrained Application Protocol) with corresponding encryption mechanisms available for the protocols. Despite the fact that MQTT has known applications in IoT with low energy overhead [25] and minimal packet size being only 2 Bytes, the main issue with the main problem with the protocol is that it can be vulnerable to eavesdropping. Therefore, Gupta, Garg, Gupta, Alnumay, Ghosh, and Sharma [26] proposed to utilise the homomorphic encryption and selective data transfer for preserving the battery on the IoT nodes. We also can see other examples where the greener IoT solutions are suggested to applications like e-health [27] to reduce the power footprint.

III. METHODOLOGY

To obtain the data for classification we use four IoT/SoC (System on a chip) devices: Raspberry Pi 3B, Nvidia Jetson Nano, Nvidia Jetson TX2, and Nvidia Jetson AGX Xavier. These devices demonstrate the full range of performance available in this field, with the Raspberry Pi demonstrating the bare minimum for local processing and the Nvidia Jetson AGX Xavier demonstrating the most powerful processing at the time of writing this article. The hardware configuration of the devices used is detailed in Table 1.

Additionally, Jetson devices support multiple power modes in addition to the standard CPU frequency scaling, making them more suitable in terms of energy efficiency and thus

[1] https://ark.intel.com/content/www/us/en/ark/products/208921/intel-core-i71165g7-processor-12m-cache-up-to-4-70-ghz-with-ipu.html

TABLE 1. Devices' Hardware Characteristics and Configurations

Device	CPU	BogoMIPS	Memory
Raspberry Pi 3B	Quad Core 1.2GHz Broadcom BCM2837 64bit	38.40	1GB DDR3
Jetson Nano	Quad-core ARM Cortex-A57 MPCore	38.40	4 GB 64-bit LPDDR4, 1600MHz 25.6 GB/s
Jetson TX2	Dual-core NVIDIA Denver2 + quad-core ARM Cortex-A57	62.50	8GB LPDDR4, 128-bit interface
Jetson AGX Xavier	Octal-core NVIDIA Carmel ARMv8.2 CPU @ 2.26GHz	62.50	32GB 256-bit LPDDR4x, 2133MHz (137GB/s)

TABLE 2. Devices' Power Modes and Characteristics

Platform	Acronym	Mode	CPU Cores	Max CPU Freq. (MHz)	Max Watts (declared)
Raspberry Pi 3B	RPI	N/A	4	1200	5
Jetson Nano	NANO0	0	4	1479	10
	NANO1	1	2	921.6	5
Jetson TX2	TX2	0	6	2000	n/a
Jetson AGX Xavier	AGX0	0	8	2265.6	30+
	AGX1	1	2	1200	10

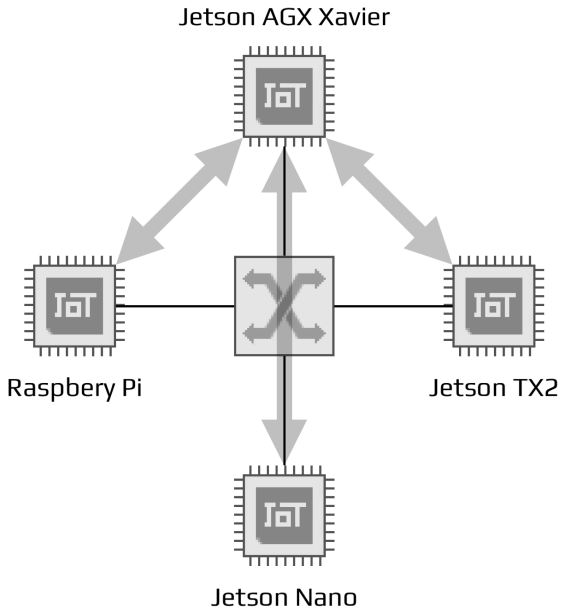

FIGURE 1. Physical network topology with VPN channels.

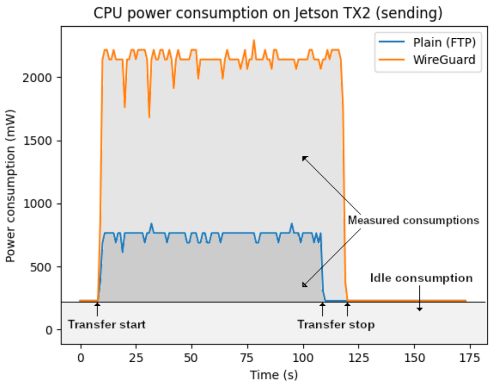

FIGURE 2. Only the difference between idle CPU power consumption and consumption during the transfer was measured, for each protocol independently.

applicable to a broader range of scenarios. We also evaluate the Nano and AGX Xavier devices in their most powerful (mode 0) and most energy-efficient mode (mode 1). The Table 2 lists the device modes and configurations used.

We connected the devices via a wired, 1 Gb/s Ethernet network (Fig. 1). Selecting this technology is not a realistic assumption, since these devices, especially when used in battery-powered scenarios, will most likely use some wireless technology for communication with their environment. However, the reason why we used an Ethernet network in these experiments is to maximize throughput while minimizing the energy consumption directly related to it. This way, we can focus on encryption-related power consumption. Energy consumption related to wireless data transfer depends on many factors, is well-covered in many research, and is out of the scope of this research. On the other hand, wireless communications are usually already encrypted at the technology level, but the OS/application-level encryption can increase the security level in some scenarios. Eventually, selecting 1 Gb/s communication turned out to be a good decision also because the results show that using faster data transfers results in better mW/MB efficiency.

All devices used in the experiment have integrated 1 Gb/s Ethernet interfaces, except the Raspberry Pi 3B, where only a FastEthernet (100 Mb/s) interface was available. Also, the lack of USB 3.0 support on this device makes it impossible to add a gigabit network interface. Therefore, we can say that the device is dimensioned so that the supported network communication speeds do not exceed the capabilities of the CPU in this aspect. On the other hand, we were not able to determine at which data transfer speeds the CPU becomes a bottleneck of the system by using the methodology.

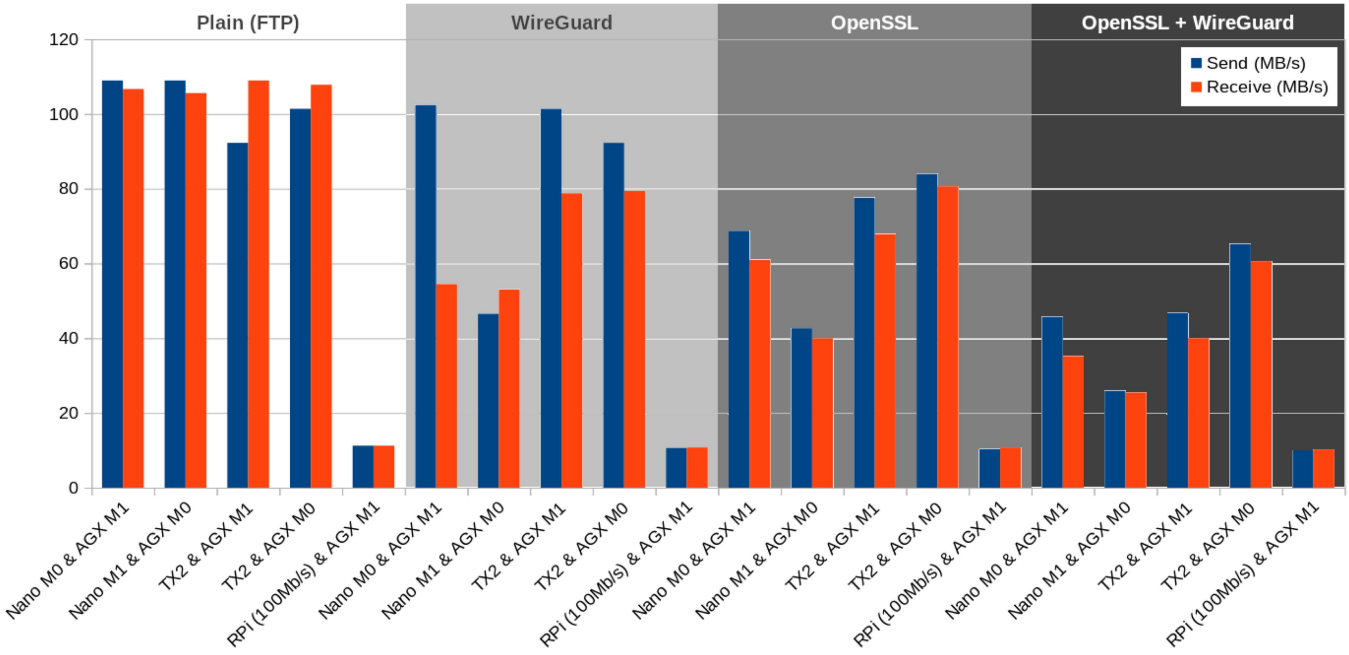

FIGURE 3. Average transfer speed for different devices and transfer protocols.

We measured power consumption during the transfer of 10 GB of data via different protocols and encryption solutions. The data was generated before the experiment, by using the (pseudo)random generator, and was stored as a 1 GB large file (the file was transferred 10 times within each measurement). The random data was used to prevent accidental compression effects, while the transfer size was selected to last long enough for the CPU to scale its frequency. The file was stored on a ram drive, to prevent external memory-related delays and power consumption. The only exception was the Raspberry Pi device, where we used a 100 MB large file, and 1 GB transfers (since the device doesn't have enough memory and has only a 100 Mb/s network interface).

The data was transferred using different transfer/encryption solutions: plain FTP, OpenSSL, and WireGuard. Jetson AGX Xavier, as the most powerful device used in the experiment, played the server role, while other devices were acting as clients. All data transfers were initiated on the client-side. For FTP protocol, VSFTP 3.0.3-9 was installed on the Jetson AGX Xavier, and other devices were using FTP client software (version 0.17-34). Also, the same protocol and tools were used for transfers protected by the WireGuard protocol, where VPN channels (secured by the 256-bit long keys) were created between each 'client' device and the AGX Xavier. Linux kernel implementation of WireGuard was used on all systems. For OpenSSL encryption, we used 2048-bit RSA keys, and the SCP tool (from the OpenSSH package) was used. All devices were running the Linux operating system, where Nvidia devices were running GNU/Linux 4.9.253-tegra aarch64, and Raspberry Pi was running Linux raspberrypi 5.10.63-v7+ armv7l. The firewall and all not necessary services were disabled on all systems.

The power consumption was measured by using the integrated I2C current and bus voltage monitor (INA3221) and via the platform-specific tegrastats utility. We measured power consumption on SoC and CPU power rails, although only the CPU is relevant for our research. We already confirmed the accuracy of the built-in power consumption measurements in our paper [Micro]. In parallel, we measured CPU frequency and utilization. Raspberry Pi device doesn't have support for reliable and detailed power consumption, so we didn't include it in the results (we can assume maximum power consumption of 15 W, but that would be too imprecise and unfair).

First, before each transfer, we determined the idle CPU power consumption. This was done by measuring a 10 seconds average consumption after one minute of the device being completely idle. Then, the client-side initiated the transfer and the power consumption was measured during the transfer, and 10 seconds after it was finished. All transfers lasted between 89 and 395 seconds.

Since there were no other processes (with significant CPU utilization) on the systems running, we measured power consumption by calculating and summing the difference between the consumption at the idle state and the consumption during the data transfer (Fig. 2). Since we analyzed only the consumption on the CPU power rail, the power needed to send/receive the data on the hardware level (e.g. for the network interface card) was not included (since it is irrelevant for this study). However, measured power consumption included both kernel level and user-space level activities. Plain FTP and OpenSSL-based transfers are expected to cause mostly user-space level consumption, while WireGuard related processing is expected to happen at the kernel level. Eventually, we measured

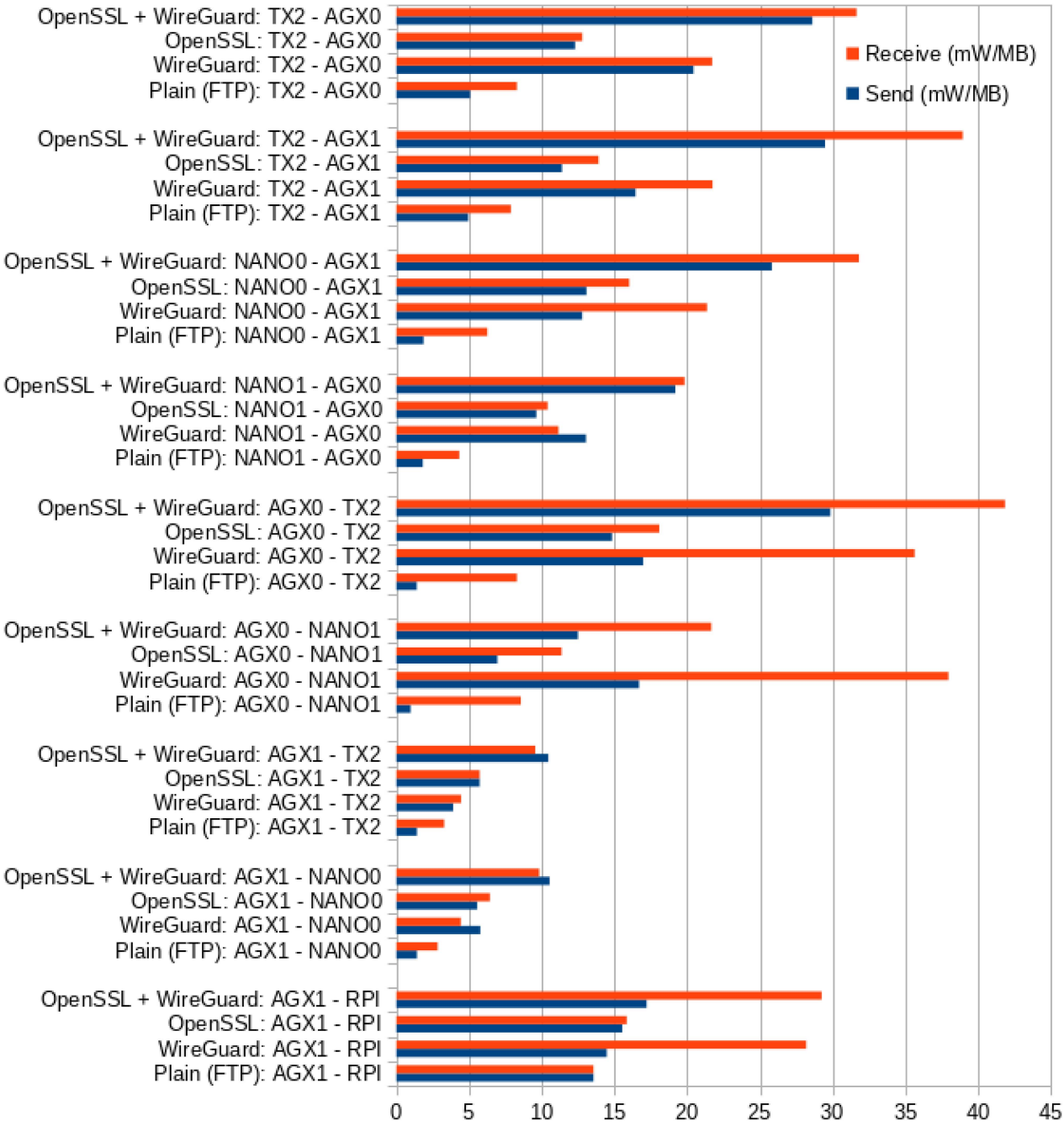

FIGURE 4. **Power consumption, expressed in consumed milliwatts per megabyte (mW/MB), for the different devices and encryptions solutions, both with sending and receiving data roles. The legend format is the following: [Protocol]: [Device where power consumption is measured] - [Peer device].**

power consumption during the transfers with both OpenSSL and WireGuard protection enabled, to maximize power load.

IV. RESULTS AND ANALYSIS

The graph in Fig. 3 shows the average transfer speeds achieved during the experiment Table 4. This graph is not significant in terms of energy efficiency, but it is important to illustrate what maximum speeds different communication protocols, in combination with different devices, can provide. Also, any value that is significantly lower than the theoretical maximum (118 MB/s) implies the existence of a bottleneck at some level. Since all values related to the plain transfer with FTP protocol, are close to that value (except with the Raspberry Pi device, which is using 100 Mb/s Ethernet connection), we can infer that the tested devices and network infrastructure are capable of transferring data at 1 Gb/s rate. On the other hand, the rest of the diagram shows that the use of encryption in most cases creates a bottleneck, a CPU-related one as we'll show later. It is, also, important to notice the difference, in

TABLE 3. Numerical Data Presented on Chart in Fig. 4 Power Consumption, Expressed in Consumed Milliwatts Per Megabyte (mW/MB), for the Different Devices and Encryptions Solutions, Both With Sending and Receiving Data Roles

Transfer	Send	Receive	Transfer	Send	Receive
Plain (FTP): AGX1 - RPI	13.56	13.56	Plain (FTP): AGX1 - NANO0	1.42	2.83
WireGuard: AGX1 - RPI	14.47	28.18	WireGuard: AGX1 - NANO0	5.77	4.43
OpenSSL: AGX1 - RPI	15.54	15.84	OpenSSL: AGX1 - NANO0	5.56	6.43
OpenSSL + WireGuard: AGX1 - RPI	17.21	29.25	OpenSSL + WireGuard: AGX1 - NANO0	10.54	9.83
Plain (FTP): AGX1 - TX2	1.42	3.29	Plain (FTP): AGX0 - NANO1	0.98	8.56
WireGuard: AGX1 - TX2	3.90	4.45	WireGuard: AGX0 - NANO1	16.70	37.97
OpenSSL: AGX1 - TX2	5.73	5.73	OpenSSL: AGX0 - NANO1	6.95	11.36
OpenSSL + WireGuard: AGX1 - TX2	10.45	9.55	OpenSSL + WireGuard: AGX0 - NANO1	12.49	21.66
Plain (FTP): AGX0 - TX2	1.40	8.29	Plain (FTP): NANO1 - AGX0	1.82	4.32
WireGuard: AGX0 - TX2	16.98	35.65	WireGuard: NANO1 - AGX0	13.04	11.15
OpenSSL: AGX0 - TX2	14.83	18.08	OpenSSL: NANO1 - AGX0	9.63	10.40
OpenSSL + WireGuard: AGX0 - TX2	29.85	41.86	OpenSSL + WireGuard: NANO1 - AGX0	19.18	19.82
Plain (FTP): NANO0 - AGX1	1.87	6.26	Plain (FTP): TX2 - AGX1	4.95	7.88
WireGuard: NANO0 - AGX1	12.78	21.37	WireGuard: TX2 - AGX1	16.45	21.75
OpenSSL: NANO0 - AGX1	13.08	16.01	OpenSSL: TX2 - AGX1	11.39	13.90
OpenSSL + WireGuard: NANO0 - AGX1	25.82	31.81	OpenSSL + WireGuard: TX2 - AGX1	29.49	38.95
Plain (FTP): TX2 - AGX0	5.08	8.28			
WireGuard: TX2 - AGX0	20.44	21.72			
OpenSSL: TX2 - AGX0	12.30	12.79			
OpenSSL + WireGuard: TX2 - AGX0	28.61	31.65			

The legend format is the following: [protocol]: [device where power consumption is measured] **-** [peer device].

some cases very significant, between sending and receiving transfer speeds, caused by the different needs of CPU time of sending/receiving functions.

The summarized results are presented in Fig. 4 (Table 3). The figure displays power consumed (in milliwatts) per transferred megabyte of data, for the devices and protocols analyzed in this study. As it is explained in the methodology part, this excludes idle CPU power consumption and consumption not related to software functions of the transfers (e.g. network interface power consumption). The first column in the legend presents the data protection protocol used (Plain FTP, WireGuard, OpenSSL, and OpenSSL and WireGuard combined. The second column represents the name of the device where measurement is performed. The third column represents the peer device name. Both communication directions (sending and receiving data) are included.

The main information we can get from the results is that, as expected and obvious, the use of encryption increases the use of CPU cycles, thus increasing the power consumption. This increase ranges from 6.74% (0.91 mW/MB on Jetson AGX M1 and RPI) to 2029.76% (28.45 mW/MB on Jetson AGX M0 and TX2) when sending data, and from 16.85% (1.16 mW/MB on Jetson AGX M1 and RPI) to 408.61% (33.57 mW/MB on Jetson Nano M0 and AGX M1) when receiving data. This non-linearity implies inefficient CPU use and will be analyzed further in the rest of the document. In most cases, as expected, power consumption when receiving is higher than when sending data. However, this difference is not proportional and depends on the protocol and devices used in the transfer. Power consumption on Jetson AGX in power mode 1, is (slightly) lower when using WireGuard than when using OpenSSL. We expected this result since WireGuard functions are implemented at the kernel level, while OpenSSL functions execute in userspace. However, this is not the case with any other combination of devices and power modes, where the power consumption of WireGuard is significantly higher than that of OpenSSL - on average 150% or 4.7 mW/MB when sending and 183% or 11.18 mW/MB, and up to 240% or 9.75 mW/MB when sending and 334% or 26.61 mW/MB when receiving data (case AGX0 - Nano 1). This difference is not directly correlated with any individual systems' characteristics and needs to be investigated further.

As previously mentioned, the expected reason for lower transfer speed when using encryption is a CPU-related bottleneck. Time diagrams in Fig. 5. show CPU frequency and utilization on the Jetson Nano device (mode 1) while receiving data from Jetson AGX (mode0) by using Plain FTP and WireGuard protocols. In both cases, CPU utilization during the transfer is high enough for the CPU to scale to maximum frequency. However, in the first case, with plain FTP transfer, the average CPU utilization is 63% (45% for core 0 and 80% for core 1). On the other hand, the average CPU utilization

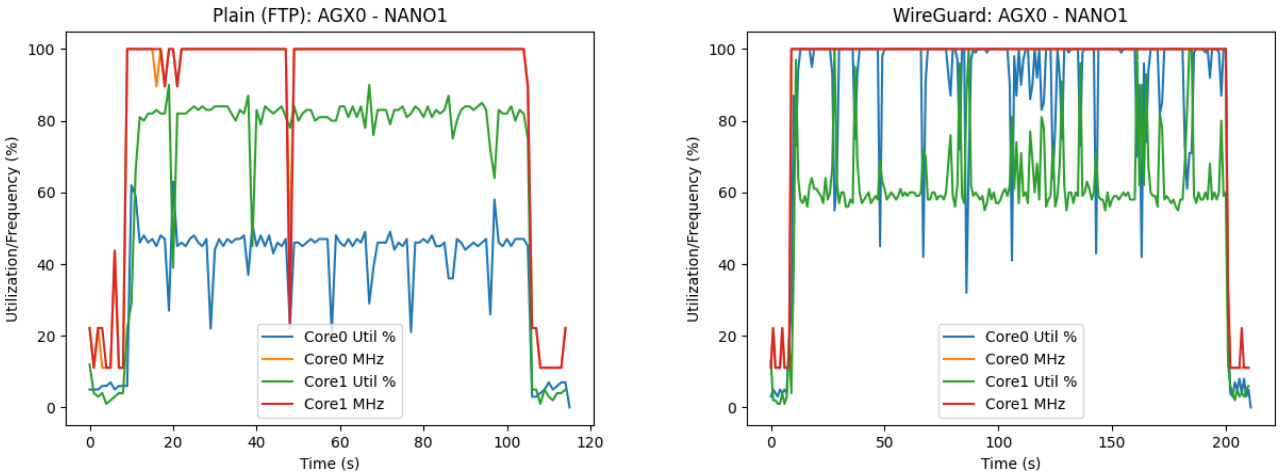

FIGURE 5. Time diagram of Jetson Nano's (mode 1) CPU frequency and utilization during the transfer using plain FTP and WireGuard protocols.

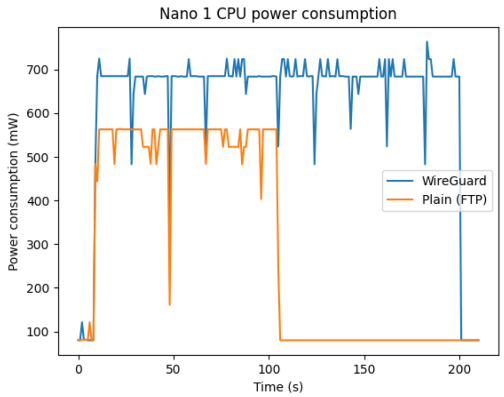

FIGURE 6. CPU power consumption on Jetson Nano (mode 1) while receiving data from Jetson AGX (mode 0) via plain FTP and WireGuard.

during the WireGuard-protected transfer is 78% (94% for core 0 and 63% for core 1). Since WireGuard doesn't support the utilization of multiple CPU cores, the close-to-maximum utilization for core 0 indicates that the CPU is a bottleneck in this transfer. (Actually, the average utilization of core 0 during the transfer is 100%, but there are nine short delays while reinitiating the transfer of a 1 GB file.)

Fig. 6 shows the CPU power consumption on Jetson Nano (mode 1) while receiving data from Jetson AGX (mode 0) via plain FTP and WireGuard protocols. As we can see, the power consumption is not dramatically different - about 24% higher when using WireGuard (680 mW) than when using plain FTP (545 mW). However, since the transfer with WireGuard lasts exactly two times longer (192 seconds, compared to 96 seconds with plain FTP), the majority of the power consumption difference originates from the second half of the transfer. So, the difference in power consumption can be seen as the difference between areas bordered by WireGuard and plain FTP lines on the chart.

Another interesting result to investigate is the different power consumption for Jetson AGX in power mode 1 when sending data to low-end devices, Raspberry Pie (Fig. 8) and Jetson Nano (mode 0) by using the plain FTP protocol. The diagram in Fig. 3 shows that maximum communication channel capacity (1 Gb/s in case of communication with Jetson Nano, and 100 Mb/s in case of Raspberry Pi device) was reached, which implies that receiving devices' CPUs were not bottlenecks. This is confirmed by the diagrams in Fig. 7, where we can see that CPU usage is not at its maximum. The CPU utilization was about two times higher when sending data to the Nano device. Both transfers lasted almost identically, 90 and 94 seconds. So, during the whole transfer CPU power consumption was identical - 468 mW. However, due to the network interface limitation on the Raspberry Pi device (100 Mb/s), the total amount of transferred data was 1 GB, while the total amount of data transferred to the Jetson Nano device was 10 GB. This results in the lower energy efficiency of AGX1 when sending data to RPI (13.56 mW/MB) than when using a faster network channel to NANO0 (1.42 mW/MB). A conclusion we can draw from this is that the devices have the highest energy efficiency when the communication channel is not a bottleneck.

Fig. 9 shows the difference between OpenSSL and WireGuard based protection in terms of where the encryption/decryption process is happening - in kernel space or userspace. The left diagram shows the CPU time distribution when using OpenSSL, while the right diagram shows the same information but for WireGuard. As expected, OpenSSL-based transfer consumes more CPU time in userspace, since encryption/decryption functions are executing in the userspace. On the other hand, encryption/decryption functions at WireGuard are executing in the kernel space, so more CPU time is consumed by system processes.

Another anomaly worth further investigation is the big difference in energy efficiency between sending (16.70 mW/MB) and receiving (37.97 mW/MB) data from Jetson AGX in power mode 0 to/from Jetson Nano in power mode 1. The power consumption chart in Fig. 10. shows that the average CPU power consumption during the sending was 1374 mW, while during the receiving it was 2308 mW. CPU frequency

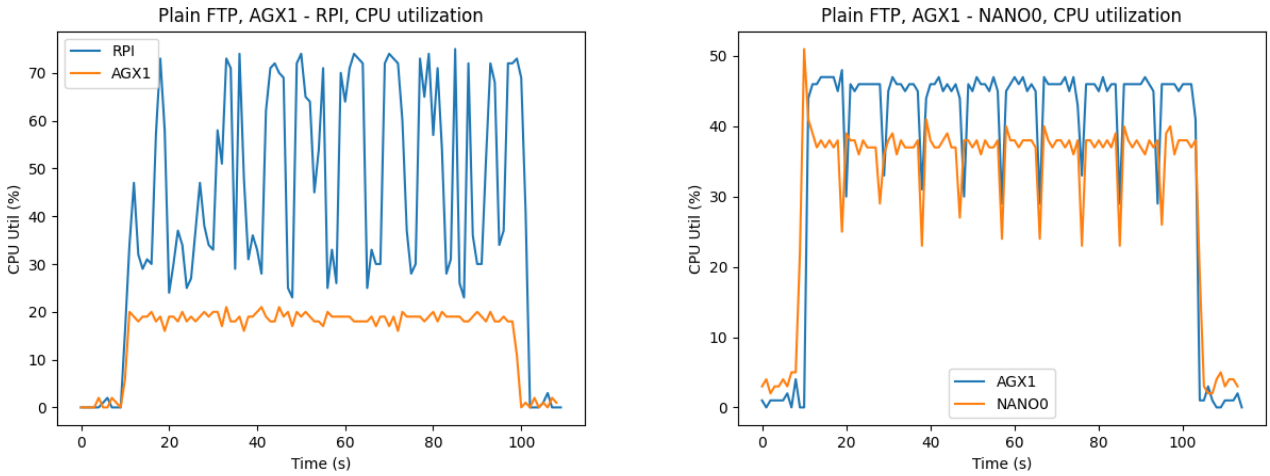

FIGURE 7. CPU utilization on AGX1, RPI, and NANO0 devices during transfers from AGX1 using plain FTP protocol.

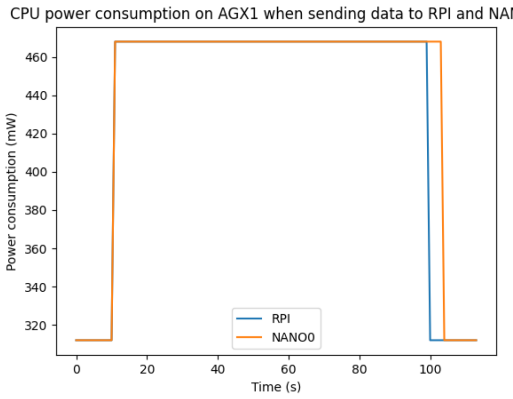

FIGURE 8. CPU power consumption on Jetson AGX (power mode 1) while sending data to Raspberry Pi and Jetson Nano (power mode 0).

chart on the same figure shows that the average CPU frequency during sending was 2089 MHz and 2202 MHz during the receiving. Also, CPU frequency change was much more frequent during the sending process. The CPU transition latency parameter in the kernel was set to the default value of 300 microseconds in both cases.

In order to further investigate this power consumption difference, we checked CPU utilization. Charts in Figs. 11 and 12 present the utilization of each CPU core during the sending and receiving.

The average utilization is summarized in the chart in Fig. 11. Since the WireGuard doesn't support multithreading, encryption/decryption was performed on Core0 both during the sending and receiving. The average utilization is slightly lower during the sending (52% vs 55%). Also, all other cores were utilized less during the sending process (6% vs 14%). Additionally, cores 1,3-7 were on average utilized at $<5\%$ during the sending, while the average utilization was $>10\%$ during the receiving. We suspect that these values overcame the threshold for activating higher CPU power consumption.

TABLE 4. Numerical Data Presented on Chart in Fig. 3 Average Transfer Speed for Different Devices and Transfer Protocols, Expressed in Megabytes per Second (MB/s)

Transfer	Send	Receive
Plain (FTP): Nano M0 - AGX M1	108.94	106.67
Plain (FTP): Nano M1 - AGX M0	108.94	105.57
Plain (FTP): TX2 - AGX M1	92.25	108.94
Plain (FTP): RPi (100Mb/s) - AGX M1	10.99	10.99
WireGuard: Nano M0 - AGX M1	102.40	54.47
WireGuard: Nano M1 - AGX M0	46.55	53.06
WireGuard: TX2 - AGX M1	101.39	78.77
WireGuard: RPi (100Mb/s) - AGX M1	10.42	10.53
OpenSSL: Nano M0 - AGX M1	68.72	60.95
OpenSSL: Nano M1 - AGX M0	42.67	40.00
OpenSSL: TX2 - AGX M1	77.58	67.81
OpenSSL: RPi (100Mb/s) - AGX M1	10.10	10.53
OpenSSL-WG: Nano M0 - AGX M1	45.71	35.19
OpenSSL-WG: Nano M1 - AGX M0	25.92	25.47
OpenSSL-WG: TX2 - AGX M1	46.76	40.00
OpenSSL-WG: RPi (100Mb/s) - AGX M1	9.80	10.00

V. CONCLUSION AND FUTURE WORK

In this article, we present the results of an energy efficiency analysis of various protocols used for secure data transfers with various IoT/EdgeAI devices. We measured the energy consumption of devices during 10 GB data transfers in both sending and receiving directions. The energy efficiency of the system was calculated in milliwatts consumed per megabyte of data transferred. We emphasize once more that we measured only the CPU power consumption caused by the transfer protocols. The results can be summarized as following.

The amount of additional power consumed for data encryption/decryption, in the case of low-speed data transfers, greatly depends on how efficiently the device consumes power in general (more precisely, how efficiently the device consumes power for plain data transmissions). For example, when

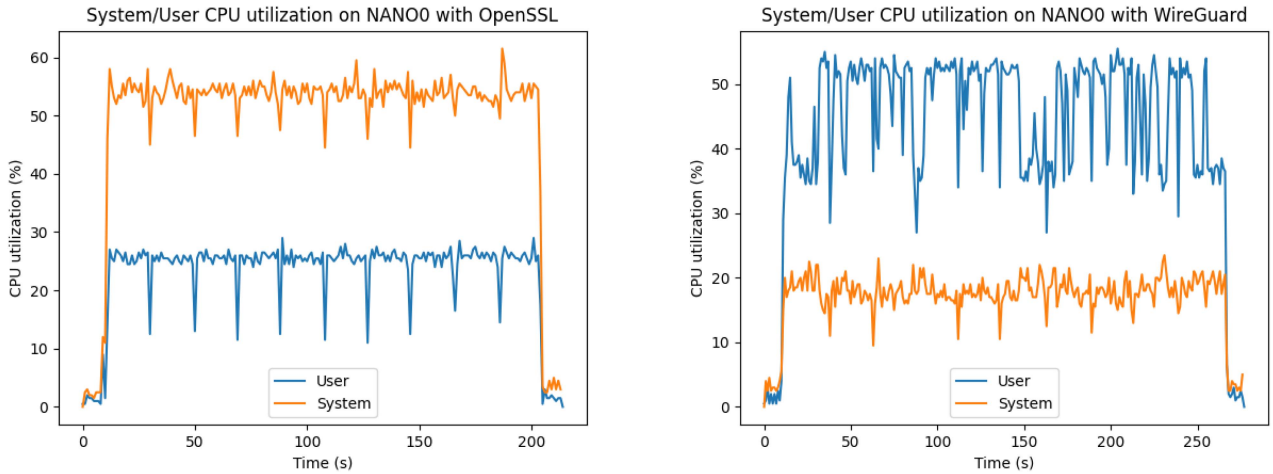

FIGURE 9. Distribution of CPU time on kernel and user space functions, for OpenSSL and WireGuard.

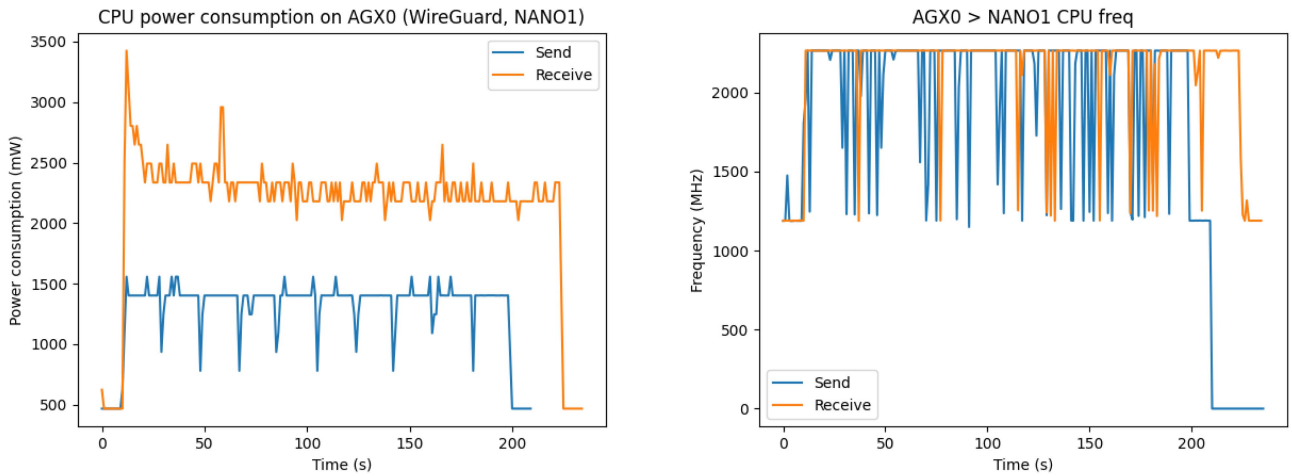

FIGURE 10. CPU power consumption (left) and working frequencies on AGX0 when sending and receiving data via WireGuard from/to NANO1.

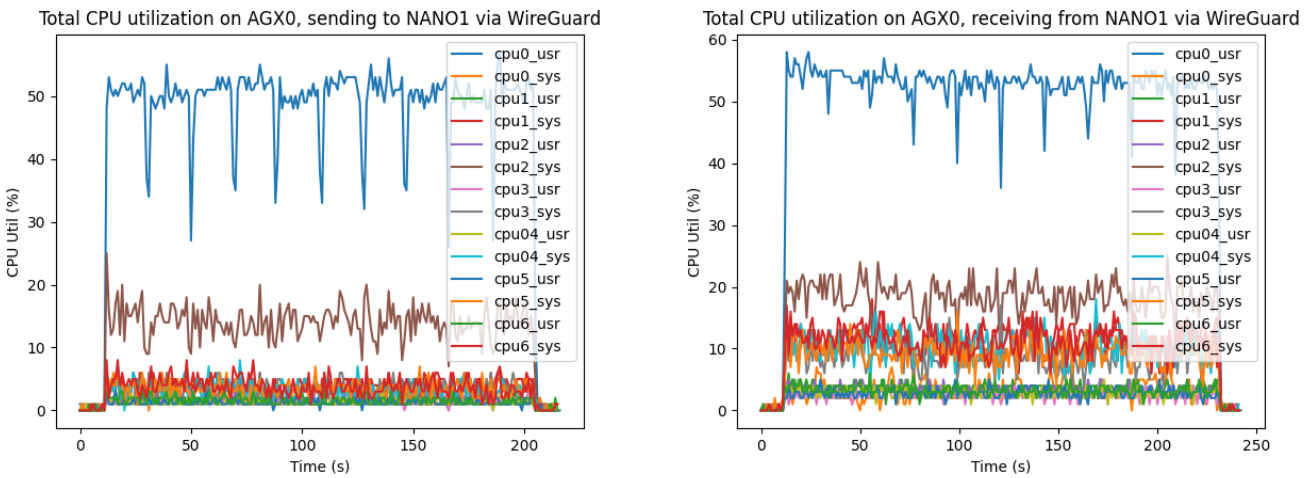

FIGURE 11. CPU cores utilization on AGX0 when sending (left) and receiving (right) data via WireGuard from/to NANO1.

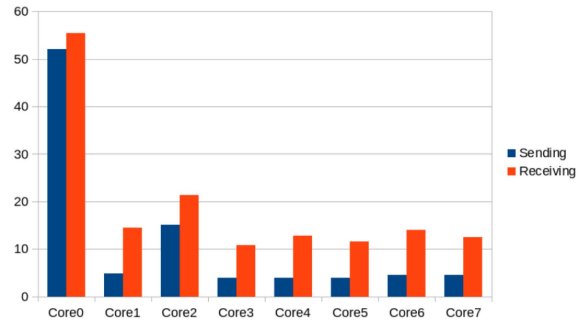

FIGURE 12. Average CPU/core utilization on AGX0 when sending and receiving data via WireGuard from/to NANO1.

sending data from AGX1 to RPI by using secure protocols, the increase in energy consumption was almost insignificant (6.74% or 0.91 mW/MB with WireGuard and 14.61% or 1.98 mW/MB with OpenSSL). On the other hand, the increase in energy consumption in high-speed data transfers was 1111.30% or 15.58 mW/MB with WireGuard and 958.21% or 13.43 mW/MB with OpenSSL.

As expected, almost all tested devices consume more power when receiving data than when sending, and it is true for both plain and encrypted transfers. The increase ranges from 57% (2.53 mW/MB) to 778% (7.59 mW/MB) with plain 1 Gb/s transfers, and from 2% (0.3 mW/MB) to 127% (21.28 mW/MB) with the encrypted ones.

WireGuard showed better performance than OpenSSL, in terms of transfer speed (MB/s). This was expected, since WireGuard operates at the kernel level, while OpenSSL operates in userspace. However, OpenSSL in almost all combinations showed better energy efficiency.

The overall conclusion is that the energy efficiency can't be determined for a device or protocol itself, since it greatly depends on factors like communication speed and direction. For example, NANO0 and TX2 showed almost identical efficiency (21.37 mW/MB and 21.75 mW/MB) when receiving data from AGX1 at very different speeds (54 MB/s and 79 MB/s), while the efficiency on AGX during those transfers was 5.77 mW/MB and 3.9 mW/MB, respectively. Also, the devices in high power modes (e.g. Nano/AGX in power mode 1, compared to mode 0) show significantly worse energy efficiency (on AGX: increase of 335% for WireGuard and 159% for OpenSSL when sending data, and 701% for WireGuard and 216% for OpenSSL when receiving data). The only general rule is that faster data transfers are more likely to result in better energy efficiency (e.g. AGX1 energy efficiency was about two times better in 1 Gb/s communications, compared to 100 Mb/s ones), as well as transfers where the CPU is maximally utilized (or, in other words, when the channel bandwidth is not a bottleneck).

Certain issues remain to be investigated further and in greater detail. For instance, the impact of using less reliable and slower communication technologies (such as WiFi) should be analyzed. Additionally, some discrepancies in power consumption/CPU utilization should be investigated further, as described in the Results and Analysis section. All of the above, whether or not presented in this article or anticipated in future work, may point toward the optimal configuration of IoT/EdgeAI networks for maximizing energy efficiency.

ACKNOWLEDGMENT

This article is based upon work from COST actions 22104 Behavioral Next Generation in Wireless Networks for Cyber Security (BEiNG-WISE), 19121 Network on Privacy-Aware Audio- and Video-Based Applications for Active and Assisted Living, 17124 Digital forensics: evidence analysis via intelligent systems and practices, supported by COST (European Cooperation in Science and Technology).

REFERENCES

[1] P. N. Bideh, J. Sönnerup, and M. Hell, "Energy consumption for securing lightweight IoT protocols," in *Proc. 10th Int. Conf. Internet Things*, 2020, pp. 1–8, doi: 10.1145/3410992.3411008.
[2] A. Fotovvat, G. M. E. Rahman, S. S. Vedaei, and K. A. Wahid, "Comparative performance analysis of lightweight cryptography algorithms for IoT sensor nodes," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 8279–8290, May 2021.
[3] S. Maitra, D. Richards, A. Abdelgawad, and K. Yelamarthi, "Performance evaluation of IoT encryption algorithms: Memory, timing, and energy," in *Proc. 2019 IEEE Sensors Appl. Symp.*, 2019, pp. 1–6.
[4] M. v. d. Hagen and B. Lucia, "Practical encrypted computing for IoT clients," 2021, *arXiv:2103.06743*.
[5] L. Liu, H. Wang, and Y. Zhang, "Secure IoT data outsourcing with aggregate statistics and fine-grained access control," *IEEE Access*, vol. 8, pp. 95057–95067, 2020.
[6] C. Li and B. Palanisamy, "Privacy in Internet of Things: From principles to technologies," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 488–505, Feb. 2019.
[7] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge intelligence: The confluence of edge computing and artificial intelligence," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7457–7469, Aug. 2020.
[8] C.-L. Su, W.-C. Lai, H.-W. Huang, C.-H. Lin, and Y.-B. Chen, "Implementation of face detection and eye tracking with convolution neutral network and edge computing on electromobility," in *Proc. 2021 Int. Symp. Intell. Signal Process. Commun. Syst.*, 2021, pp. 1–2.
[9] A. Khalifeh, F. Alsyayid, H. Armoush, and K. A. Darabkh, "An experimental evaluation of the advanced encryption standard algorithm and its impact on wireless sensor energy consumption," in *Proc. 2020 Int. Conf. Innov. Intell. Inform., Comput. Technol.*, 2020, pp. 1–6.
[10] "Arduino cryptography library," 2024. [Online]. Available: https://rweather.github.io/arduinolibs/crypto.html
[11] F. Mitchell, "The use of artificial intelligence in digital forensics: An introduction," *Digit. Evidence Electron. Signature L. Rev.*, vol. 7, 2010, Art. no. 35.
[12] "Watchout: Analysis of smartwatches for children." 2017. [Online]. Available: https://fil.forbrukerradet.no/wp-content/uploads/2017/10/watchout-rapport-october-2017.pdf
[13] K. Greuter, "Evaluating the quality of the international consumer IoT cyber security standard," B.S. thesis, University of Twente, Enschede, The Netherlands, 2020.
[14] "Rob aitken," 2021. [Online]. Available: https://www.arm.com/blogs/blueprint/performance-per-watt
[15] W. v. Dorst, "The quintessential Linux benchmark," 1996. [Online]. Available: https://www.linuxjournal.com/article/1120
[16] J. Geerling, "Power consumption benchmarks," 2024. [Online]. Available: https://www.pidramble.com/wiki/benchmarks/power-consumption
[17] "Crypto acceleration: Enabling a path to the future of computing," 2022. [Online]. Available: https://newsroom.intel.com/articles/crypto-acceleration-enabling-path-future-computing/

[18] "Energy-efficient platforms–considerations for application software and services," 2011. [Online]. Available: https://www.intel.com/content/dam/develop/external/us/en/documents/green-hill-sw-20-185393.pdf
[19] M. v. d. Hagen and B. Lucia, "Practical encrypted computing for IoT clients," 2021, *arXiv:2103.06743*.
[20] P. Sudhakaran, "Energy efficient distributed lightweight authentication and encryption technique for IoT security," *Int. J. Commun. Syst.*, vol. 35, no. 2, 2022, Art. no. e4198.
[21] L. E. Kane, J. J. Chen, R. Thomas, V. Liu, and M. McKague, "Security and performance in IoT: A balancing act," *IEEE Access*, vol. 8, pp. 121969–121986, 2020.
[22] F. Alsyayid, H. Armoush, and K. A. Darabkh, "An experimental evaluation of the advanced encryption standard algorithm and its impact on wireless sensor energy consumption," in *Proc. 2020 Int. Conf. Innov. Intell. Inform., Comput. Technol.*, 2020, pp. 1–6.
[23] P. Panahi, C. Bayilmiş, U. Çavuşoğlu, and S. Kaçar, "Performance evaluation of lightweight encryption algorithms for IoT-based applications," *Arabian J. Sci. Eng.*, vol. 46, no. 4, pp. 4015–4037, 2021.
[24] P. N. Bideh, J. Sönnerup, and M. Hell, "Energy consumption for securing lightweight IoT protocols," in *Proc. 10th Int. Conf. Internet Things*, 2020, pp. 1–8.
[25] A. Shalaginov, O. Semeniuta, and M. Alazab, "MEML: Resource-aware MQTT-based machine learning for network attacks detection on IoT edge devices," in *Proc. 12th IEEE/ACM Int. Conf. Utility Cloud Comput. Companion*, 2019, pp. 123–128.
[26] S. Gupta, R. Garg, N. Gupta, W. S. Alnumay, U. Ghosh, and P. K. Sharma, "Energy-efficient dynamic homomorphic security scheme for fog computing in IoT networks," *J. Inf. Secur. Appl.*, vol. 58, 2021, Art. no. 102768.
[27] M. Kaur, D. Singh, V. Kumar, B. Gupta, and A. A. A. El-Latif, "Secure and energy efficient-based e-health care framework for green Internet of Things," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 3, pp. 1223–1231, Sep. 2021.

ALEKSANDAR JEVREMOVIC (Member, IEEE) received the Ph.D. degree. He is currently a Full Professor with the Faculty of Informatics and Computing, Singidunum University, Belgrade, Serbia, a Guest Lecturer with Harvard University, Cambridge, MA, USA, and a Visiting Research Fellow with Cyprus Interaction Laboratory, Limassol, Cyprus. He is also recognized as an Expert Level Instructor at Cisco Networking Academy Program. He has authored or coauthored number of research articles and made contributions to three books about computer networks, computer network security, and web development. Since 2018, he has been a Serbian Representative at the Technical Committee on Human–Computer Interaction, UNESCO, International Federation for Information Processing.

ZONA KOSTIC (Member, IEEE) received the Ph.D. in computer science from the University of Belgrade, Belgrade, Serbia, in 2014. She is currently an Experienced Faculty with interests at the intersection of data visualization, computer vision, and digital realities. She was also a member of the Visual Computing Group (Harvard SEAS) in 2016 and the Innovation Labs (Harvard Business School) in 2018. She has authored or coauthored six books, and a number of research papers at scientific journals and conferences. Her professional contributions were awarded with the Distinction and Excellence Award for data science capstone courses at the Harvard John A. Paulson School of Engineering and Applied Sciences.

IVAN CHORBEV (Member, IEEE) received the Ph.D. degree in computer science and engineering in 2009. He is currently a Professor and the Dean of the Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University in Skopje (UKIM), Skopje, North Macedonia. He has authored or coauthored more than 90 papers in the area of heuristic optimization algorithms, combinatorial optimization, knowledge discovery, machine learning, medical data mining, telemedicine, machine translation, information retrieval, assistive technologies, and software engineering. He is the Member of the Board of Directors in the Business Accelerator UKIM representing the University's share.

DRAGAN PERAKOVIC (Member, IEEE) received the Ph.D. degree. He is currently the Head of the Department for Information and Communication Traffic and the Head of Chair of Information Communication Systems and Services Management, the University of Zagreb, Zagreb, Croatia, where he is currently a Full Professor. He is Visiting Professor with the University of Mostar and the University of East Sarajevo, BiH. He is International Expert with the International Center for AI and Cyber Security Research and Innovations, Asia University, Taiwan.

ANDRII SHALAGINOV (Member, IEEE) received the Ph.D. degree in information security from the Norwegian University of Science and Technology, Trondheim, Norway. He is currently an Associate Professor and the Head of SmartSecLab, Kristiania University College, Oslo, Norway. His work in academia and industry is widely related to the application of AI for cybersecurity, detection of computer viruses, network attacks, and protection of IoT devices. He is an affiliated member of the Malware Lab and Digital Forensics Group, Norwegian University of Science and Technology. He was involved as a cybersecurity Researcher in the EUIPO framework related to malware analysis on copyright-infringing websites.

IVAN CVITIC received the Ph.D. degree. He is currently an Assistant Professor with the Faculty of Transport and Traffic Sciences, University of Zagreb, Zagreb, Croatia, and the Head of the Laboratory for Cybersecurity and eForensics. He has authored or coauthored more than 80 scientific papers at the international conferences, scientific books, and highly-rated scientific journals. His research domain and interests include cybersecurity, applied machine learning and artificial intelligence, modeling network traffic anomalies, DDoS, the Internet of Things, digital forensics, and quantum communications. He is a member of the editorial board and reviewer board, and a guest editor for several highly rated scientific journals and international conferences.