On the Evaluation of the Total-Cost-of-Ownership Trade-offs in Edge vs Cloud deployments: A Wireless-Denial-of-Service Case Study

Panagiota Nikolaou, Yiannakis Sazeides, Alejandro Lampropulos, Denis Guilhot, Andrea Bartoli, George Papadimitriou, Athanasios Chatzidimitriou, Dimitris Gizopoulos, Konstantinos Tovletoglou, Lev Mukhanov and Georgios Karakonstantis

The final version of this paper appears in IEEE Transactions on Sustainable Computing (TSUSC) 2019, January 2019.

Abstract— We are witnessing an explosive growth in the number of Internet-connected devices and the emergence of several new classes of Internet of Things (IoT) applications that require rapid processing of an abundance of data. To overcome the resulting need for more network bandwidth and low network latency, a new paradigm has emerged that promotes the offering of Cloud services at the Edge, closer to users. However, the Edge is a highly constrained environment with limited power budget for servers per Edge installation which, in turn, limits the number of Internet-connected devices, such as sensors, that an installation can service. Consequently, the limited number of sensors leads to a reduction in the area coverage provided by them and puts in question the effectiveness for deploying IoT applications at the Edge. In this paper, we investigate the benefits of running an emerging security focused IoT application, (jamming detection), at the Edge vs. the Cloud by developing a Total Cost of Ownership (TCO) model, which considers the application's requirements as well as the Edge's constraints. For the first time, we build such a model based on realistic performance and energy-efficiency measurements obtained from commodity 64-bit ARM based micro-servers that are excellent candidates for supporting Cloud services at the Edge. Such servers represent the type of devices that can provide the right balance between power and performance, without requiring any complicate cooling and power supply infrastructure, which will not be available at the de-centralized deployments. Aiming at improving the energy efficiency, we exploit the pessimistic design margins adopted conventionally in such devices and investigate their operation under lower than nominal supply voltage and memory refresh-rate. Our results show that the jamming detection application deployed at an Edge environment is superior to a Cloud based solution by up to 2.13 times in terms of TCO. Moreover, when servers operate below nominal conditions, we can achieve up to 9% power savings which enables in several situations 100% gains in the TCO/area-coverage metric, i.e double area can be served with the same TCO.

Index Terms— Internet of Things, Cloud Computing, Edge Computing, Power Efficiency, Operation beyond Nominal Margins

1 Introduction

The number of intelligent Internet-connected devices is growing daily and will soon be in the order of tens of billions, forming the Internet of Things (IoT). Each of these devices is pushing data to the Internet that are soon expected to reach 24.3 exabytes [1]. This rapid data growth will put an unprecedented pressure on the current Internet infrastructure and centralized (Cloud) datacenters, which are already oversubscribed. Coping with this imminent, data flood requires not only enhancement of the processing capabilities of the current servers but also rethinking of the way we communicate and process data across the Internet.

Edge/Fog computing is a recently introduced approach that has the potential to ensure the sustainability and scaling of the Internet in the IoT era. This paradigm advocates for the

- *P. Nikolaou and Y. Sazeides are with University of Cyprus*
- *A. Lampropulos, D. Guilhot and A. Bartoli are with WorldSensing*
- *G. Papadimitriou, A. Chatzidimitriou and D. Gizopoulos are with National and Kapodistrian University of Athens*
- *K. Tovletoglou, L. Mukhanov and G. Karakonstantis are with Queen's University Belfast*

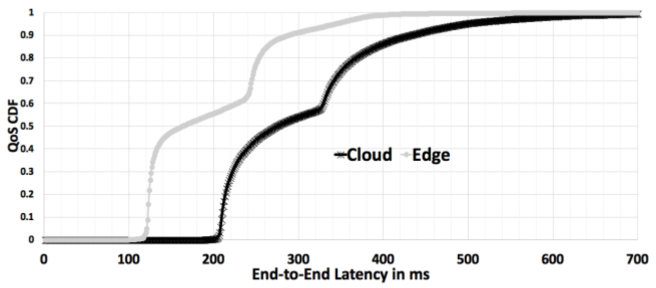

Fig. 1. End-to-End Latency for running the application in Edge and Cloud deployments

execution of services closer to the sources of data [2], [3], aiming this way to reduce application latency between the end user and the datacenter and at the same time relax the pressure on network bandwidth. Figure 1, shows the cumulative distribution of the end-to-end latency when a specific application runs in Edge and Cloud deployments. The end-to-end latency includes the network and compute time of the application (details about the methodology used to obtain these results are given in Section 4). Figure 1 reveals considerable difference between the End-to-End latency for the Cloud and Edge deployments. As can be seen, this difference may be as large as 100 milliseconds. For a high quality of service (QoS) response time constraint application this extra 100 milliseconds latency may render infeasible to run the application on the Cloud or in the best case requires an expensive Cloud deployment to ensure fast processing latency.

Recent studies develop schemes that manage the data processing of IoT applications across distributed datacenters [4], [5], [6]. In these studies, data are transferred from IoT sensors to local micro-datacenters for pre-processing and selection which of the data to forward to a centralized datacenter. Examples of IoT applications with a tight response time and QoS constraints include face recognition [27], [28], traffic counting and video processing applications [57], as well as, applications for detecting jamming attacks of wireless networks [7], [41]. All these applications consist of sensors that collect and send data to a processing device. Their main QoS requirement is the response-time and, therefore, are naturally suited for Edge deployments. However, servers used to run these applications can only process, within a required detection time window, data from a limited number of sensors, or put in another way, servers oversubscribed to process data from many sensors will suffer from QoS violations. Moreover, each sensor covers a fixed area and Edge deployments have limited power budget for servers per installation. Consequently, an Edge installation may be able to support a limited number of sensors and cover a limited area. This highlights a key challenge for the successful realization of Edge computing: the area covered by the sensors. Evidently, the most critical challenge for the successful Edge deployment is the efficient use of the limited power of an Edge installation. Exceeding the power cap of the facility is unacceptable as there will be a disruption of power. To avoid such overloads, both Edge and Cloud deployments rely on power capping schemes that enforce power budgets of individual servers [9], [10] or over ensembles [11], [12]. In this regard, the use of more power efficient servers, facilitates the increase of area coverage without exceeding the Edge's or Cloud's power budget.

In this paper, we characterize an important security IoT application with a tight response time QoS requirement using a state-of the art 64-bit ARMv8 based micro-server. Such a server is an excellent representative of the high-performance devices based on energy efficient-embedded devices that are required to support Cloud services at the Edge without complex cooling and power supply infrastructures. We evaluate the trade-offs among the area coverage, power efficiency and QoS when running the application in an Edge vs a Cloud environment. To accomplish this, we rely on a new metric: the Total Cost of Ownership (TCO) over area coverage.

To the best of our knowledge, this is the first work that provides a holistic evaluation that considers different metrics, such as TCO, QoS, area-coverage and power efficiency, using an energy efficient and high-performance device.

The application that we evaluate is a Wireless denial-of-service (WDoS) attack's detection application [58]. Current wireless networks are vulnerable to attacks by devices readily available in the market [8]. Such devices can essentially jam a wireless network and thus disrupt any running application. The WDoS application processes data sent by sensors that continuously scan the wireless spectrum and, with the assistance of signal processing algorithms and filters, detects jamming attacks. WDoS prevention applications can detect jamming attacks and increase the availability of secure and resilient wireless networks used to connect the IoT devices at the Edge.

The main contributions of this paper are the following:

i) We analyze the limits of operation of the WDoS application and reveal possible Extended Operating Points (EOP) (*i.e.,* voltage, refresh rate) of each hardware component (*i.e.* cores and memories) for enhancing the energy efficiency of micro-servers.

ii) We evaluate the possible gains of an Edge deployment compared to a Cloud one, using a TCO model. Moreover, we evaluate the TCO/area-coverage gains of Energy efficient Edge micro-servers compared to Edge micro-servers operating at nominal margins. As far as we know, this is the first work that evaluates both, Capex and Opex costs, in conjunction with the area coverage for an IoT application for Cloud, Edge and Energy Efficient Edge deployments.

The rest of the paper is organized as follows. Section 2 provides a background on Edge computing and wireless denial-of-service attacks. Section 3 presents the basics of the TCO model, as well as the characterization framework used to expose the below-nominal but safe operation points of the micro-servers. Section 4 describes the experimental methodology. Section 5 presents the characterization results, while Section 6 discusses TCO based analysis of Edge vs Cloud operation and Edge energy efficient operation. Finally, Section 7 concludes.

2 Background and Challenges

2.1 Edge computing

The need for fast response times in many sensor-based IoT applications necessitates deployments with tight QoS timing requirements. Many applications cannot tolerate latencies that exceed one or two hundreds of milliseconds [13], [14]. Even though Cloud Computing is centralized and requires minimal management effort or service provider's interaction, it hardly meets the QoS and response time requirements for IoT applications, due to the network latency between the sensors and a remote datacenter. On the other hand, Edge deployments closer to the data, facilitate meeting QoS requirements by avoiding network latency [15], [16]. This distributed deployment near the sources of data at many sites is referred as Edge (or Fog) computing [2]. Each Edge deployment site can contain one or even numerous servers, called, Cloudlets [17] and Micro DataCenters [18]. Edge Computing is not meant to replace traditional Cloud architectures, but Cloud and Edge computing can work in unison to reduce the total end-to-end response time. Edge is well suited for IoT applications, where sensors collect data and send them to Edge sites for processing, thus avoiding high network latencies compared to a centralized datacenter. The Edge deployment acts as a filter that reduces the network bandwidth pressure to the Cloud [19].

Apart from the need to reduce the latency to satisfy an IoT application's QoS time requirement, the communication of the data to the Cloud can lead to serious security and privacy issues, which in some cases is unacceptable to the end users [20].

Figure 2, shows an IoT system architecture that includes both Edge and Cloud deployments. The

Edge servers are near the data and are responsible for data collecting from various devices, data processing and transferring a concise report to the Cloud.

Despite the substantial advantages, Edge Computing has some limitations. A major one, is that Edge sites/facilities are power constrained [4]. Thus, the number of servers per site needs to fit the power budget that is provided by an electricity provider and is not already allocated for other uses. Edge facilities can be ordinary buildings with several other electrical appliances in use. Certainly, an electricity provider can increase the power budget at a facility but this comes with an extra cost. Consequently, Edge servers that are more power efficient may hold the key for successful Edge deployment since they will allow more servers per site and processing of data from more sensors without the extra costs to the electricity providers.

Several prior works, consider the Edge-Cloud trade-offs to decide where to place highly constrained applications and satisfy their requirements without compromising power and availability [4], [5], [6], [23], [24], [25], [26], [27], [28]. It is also worth highlighting that existing studies of Edge deployments relied on measurements obtained from either too simple devices (e.g. raspberry-pi) or too powerful ones (e.g. classical high-end servers) which do not strike the right balance between energy and performance that is essential in Edge installations [21], [22].

2.2 Wireless Denial of Service Application

Wireless networking plays an important role in achieving ubiquitous computing where network devices are embedded in environments that provide continuous connectivity and services, thus improving human's quality of life.

However, due to the exposed nature of wireless links, current wireless networks can easily be attacked by jamming technology. Jammer detectors are commercially available as

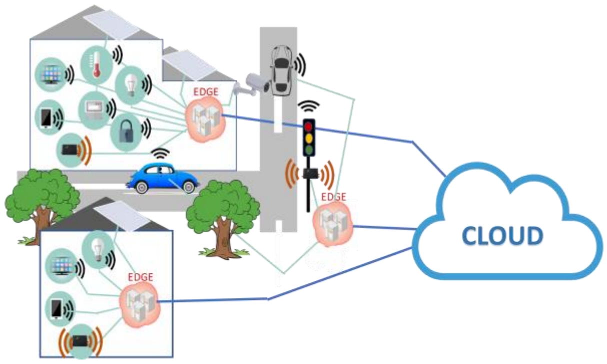

Fig. 2. IoT System Architecture including Edge and central Cloud deployments

countermeasures against jamming systems [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [58]. In this paper, we evaluate one of these Wireless Denial of Service attack (WDoS) solutions [58].

WDoS is a standalone solution that monitors the entire wireless spectrum using various sensors to detect anomalies derived from a Denial of Service attack which renders all the wireless devices useless. This solution does not need to be integrated internally in the wireless network, and offers a wide and easy-to-deploy solution for the most heterogeneous and challenging critical infrastructure wireless environments. To that end, the WDoS firstly performs a detailed analysis of the radio frequency spectrum, and then processes the acquired data to identify potential anomalies, giving rise to alarms and warning messages.

The WDoS solution uses several sensors with antennas connected to a Software Defined Radio (SDR) module which digitalizes the radio spectrum to a binary stream and transmits this to a processing board. The board processes in real time the incoming data while applying different filters and algorithms to match the signals found to four types of well-known jamming signals: Pulsed Jammer, Wide Band Jammer, Continuous Wave Jammer and LFM Chirp Jammer. When the WDoS application detects an attack, this incident is communicated by the processing board to the monitoring server that runs on a separate machine. Finally, a visualization tool visualizes all the attacks reported to the

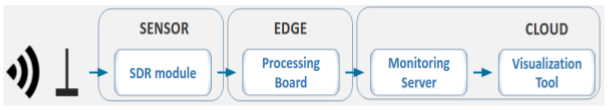

Fig. 3. Architecture of DoS Jammer Detector Application including Edge and Cloud Deployments

monitoring server in real time. The main architecture of the solution is shown in Figure 3.

Powerful Edge servers would allow the processing board to be simple and low cost, as it can concentrate the processing of high amounts of radio frequency spectrum data at the Edge. With powerful Edge servers, more than one instances of the processing application can be executed on the processing board (Edge server) by connecting various sensors on it and reducing the number of processing boards. The jammer detection results can be transmitted to the Cloud for storage, visualization, and post-processing.

2.2.1 WDoS Application Requirements

The WDoS application has several requirements in terms of availability, timing QoS, data transition ratio and sensor's area coverage.

2.2.1.1 Availability Requirements

Availability greatly depends on the type of installation in which the solution is deployed.

Some of the most demanding installations require 99% availability of the attack detection service, i.e. the service should be available in 99% of the total service time. For some installations, such as smart construction service monitoring deployments, the availability is not so critical because of a low level of criticality of a service or a small amount of data transferred for the processing, which could be a few bytes per hour or even less. Availability of 50% or less should be optimal for shopping malls or train stations, where the wireless network is used by users for non-critical purposes, such as recreational activities.

In this work, we consider the high availability requirement of 99% to evaluate a highly constraint deployment of the application.

TABLE 1
WDOS Jammer Detection Application's Requirements

Requirement	Description
Availability	99%
QoS (End to End Latency)	90th percentile of the decisions need to be under 400ms

2.2.1.2 Quality of Service (End-to-End Latency) Requirements

We measure the WDoS latency as the time it takes to detect an attack. This time is a function of the width of the band that is analyzed. The QoS for the detection time is 400 milliseconds for the 90% of the decisions (if it is jammer detection or not) on a 5 MHz band. This is the end-to-end time that includes both the transmission time of the data to the processing board, as well as, the compute time for processing the data. Thus, QoS is determined by the sum of the processing time and the data transmission time over the network. Overall, a high-performance computing server may help to reduce the detection latency but this comes at a cost of increased energy consumption.

2.2.1.3 Data Transmission Rate

The highest data rates exist between the SDR module and the Processing Board. In this case, the maximum rate is 305 Mbps (5 Msps) and the lower rate is 30.5 Mbps (0.5 Msps). Higher data rates enable better detection accuracy, however lower rates can be also useful. The data transmission rate between the processing board and the monitoring server is much lower, as the processing board transfers only about detected attacks, i.e. the type of a jammer attack, frequency, jammer power and a timestamp. This will result in about 100 bytes of payload per packet and a minimum of 40 packets per second (one packet per decision per algorithm).

In our evaluation study, to assess the benefits of Cloud deployment, we use the lowest data transmission rate, imposing an assumption that there is no any bandwidth degradation.

2.2.1.4 Sensor's Area Coverage

In our WDoS solution, one sensor that monitors the wireless band of 2.4GHz covers an area of approximately 25 square meters. Thus, in real deployments, several sensors should be used together to cover a large area, such as a shopping center floor or an airport security screening area. For such areas, multiple instances of the WDoS application should be run on the same processing board. However, this might stress the hardware and increase the processing time, as well as power consumption. Power consumption is directly related to the number of hosted servers and running workloads. The peak power consumption became an increasingly important hardware feature in many facilities, since it cannot exceed the power budget provided by the electricity suppliers. This implies that the number of sensors and processing boards that can be installed in a facility depends on the available power budget at a given site. At the same time, a low power budget may not allow to use as many sensors as required to cover a specific area. As a result, the 100% area coverage may not be achieved for some deployments.

All the described requirements that used for this analysis are summarized in Table 1.

3 Total Cost of Ownership model and Server Characterization

This Section describes the model and parameters that are used in this work to measure the Total Cost of Ownership (TCO) and assess the tradeoffs of Edge and Cloud deployments. Also, it describes the characterization process used to enhance the power efficiency of a servers by revealing possible Extended Operating Points (EOP) (*i.e.*, below nominal supply voltage, and memory refresh rate) of each hardware component (*i.e.* processors and memories) in a micro-server.

3.1 TCO Model and Parameters

TCO is a key optimization metric for the design of a system. The TCO is determined by the sum of capital and operational expenses. Capital expenses (Capex) include the cost of acquisition of a building, the power capex costs with the electricity payment, the cooling equipment acquisition cost, the cost of acquiring the servers including all their components and networking equipment costs. On the other hand, operational expenses (Opex) include operation power and maintenance costs.

Prior works proposed to guide the Datacenter design, accounting TCO as the key parameter [43], [44], [45], [46], [47], [48], [49]. This work uses two previously proposed TCO models [43], [49] to perform a TCO evaluation and compare Edge and Cloud deployments.
To the best of our knowledge, this is the first work that evaluates both Capex and Opex costs for an IoT application for Cloud, Edge and (energy-)Efficient Edge deployments.

The total TCO of a deployment consisting of *N* servers is determined as the sum of Capex Cost (C_{Capex_i}) and Opex Cost (C_{Opex_i}) of all the servers (*i*) as follows [43]:

$$TCO = \sum_{i}^{N} [\ C_{Capex_i} + C_{Opex_i}\ ] \quad (1)$$

3.1.1 System Architecture

To estimate the TCO and area coverage of a specific deployment we use the architecture described in Figure 4. The figure shows that the electricity provider delivers a specific power budget to each site. The figure shows five facilities with different power budgets. Let us assume that three of them (facilities 1, 3, 4) use basic Edge servers for processing the data, whereas facility 2 sends and processes the data in the Cloud and facility 5 uses more power efficient Edge servers to process the data.

Figure 4 also shows three bars next to each facility that represent the power budget for each facility (red bar), the cost that depends on the servers that can be operated within the specific power budget (green bar) and the area coverage of the sensors for the specific location which depends on the sensors that can be processed by each server (blue bar). For simplicity, we assume facilities with 100m2 area. Consequently, to ensure a 100% area coverage for this application, four sensors per facility are required (one per 25 m2 as described in the application requirements).

As can be seen, for facilities that use basic Edge servers to process the data (facilities 1, 3, 4), area coverage, power budget and cost are strictly correlated. So, the more the area coverage is achieved, the more power budget is available, more servers are accommodated and the higher the cost. On the other hand, even though facility 2 provides full area coverage, it needs less power budget because it transfers and processes the data only on the Cloud. However, this option may not meet the QoS time requirement. Finally, facility 5 is shown to have full coverage with the same power budget as facility 4, that only achieves three quarters of the area coverage. This is made possible from the use of more power efficient servers that allow more servers to be operated within the same power budget. In Section 5 we explore in detail these types of trade-offs.

The following equation shows how the area coverage is determined:

$$Area\ Coverage = \frac{EstimatedSensors}{RequiredSensors} \quad (2)$$

Equation 2 shows that area coverage is correlated with the number of sensors *(EstimatedSensors)* that can be placed in a facility and the required number of sensors *(RequiredSensors)* that are needed to cover 100% of the specific area which is equal to: (total facility area)/(area covered per sensor). To determine the *EstimatedSensors* number we use the following equation:

$$EstimatedSensors = MAX_i^{maxInstances} \left\lfloor \frac{PowerBudget}{(ServerPower_i * PUE)} * i \right\rfloor \quad (3)$$

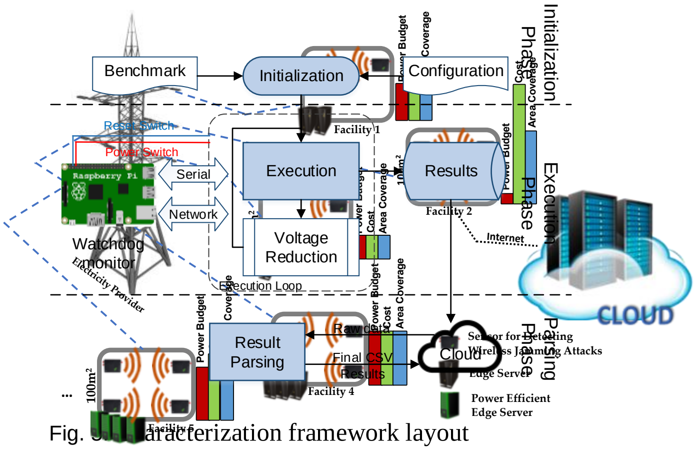

Fig. 3. Characterization framework layout

Fig. 4. Architecture of Edge servers in different locations

The actual number of sensors that can be deployed is estimated by considering the number of instances per server (number of sensors that data are getting processed on a server). The power of each server configuration that runs specific number of instances is multiplied by the power usage effectiveness (PUE) and then divided by the *PowerBudget* that corresponds to the power available at a given facility. Then, the result is multiplied by the number of instances, *i*. The total number of servers that are needed to host the specific number of instances is then obtained from:

$$N = \frac{EstimatedSensors}{i} \quad (4)$$

The number of servers, *N*, and power per server is fed to the TCO model to determine the TCO of the deployment.

The metric that we optimize in this work is the TCO over Area coverage which captures both metrics of interest, as follows:

$$\text{OptimizationMetric} = \frac{TCO}{Area\ Coverage} \quad (5)$$

3.2 Characterization framework

To improve energy efficiency of a micro-server, we need to investigate the operation limits of voltage and memory refresh rates.

Exposing the safe voltage margins of an application is a time-consuming and difficult process due to several abnormal behaviors that can exist [52], [53], [54], [55]. To this end, we developed an automated characterization framework, which is outlined in Figure 5, (1) to identify the target system's limits when it operates at scaled voltage, frequency conditions and DRAM refresh rates, and (2) to record/log the effects of a program's execution under these conditions. The automated framework (outlined in Figure 5) is easily configurable by the user and can be embedded to any Linux-based system, with similar voltage and frequency regulation capabilities. The characterization framework [52], [53] consists of three phases (Initialization, Execution, Parsing). During the initialization phase, a user can declare a benchmark list with corresponding input datasets to run in any desirable characterization setup. The characterization setup includes the voltage and frequency (V/F) values on which the experiment will take place and the cores where the benchmark will be run. To reduce the DRAM power, we adopt the framework to characterize DRAM reliability operating under different refresh rates and the supply voltage. Particularly, we use this framework to identify the optimal DRAM refresh rate and voltage which does not trigger uncorrectable errors or system crashes. The execution phase consists of multiple runs of the same benchmark, each one representing the execution of the benchmark in a pre-defined characterization setup. The set of all the characterization runs running the same benchmark with different setups represents a campaign. In the parsing phase of our framework, all log files that are stored during the execution phase are parsed in order to provide a fine-grained classification of the effects observed for each characterization run.

We have also extended the error reporting capabilities of existing mechanisms (i.e. ECC in caches and DRAMs) with system configuration values, sensor readings and performance counters for identifying correctable (CE) and uncorrectable errors (UE). In addition, to account for any undetected error and essentially detect any Silent Data Corruption (SDC) that could go undetected by ECC, we compare the output of each execution with a golden reference.

The framework provides the following features; it:

- compares the outcome of the program with the correct output of the program when the system operates in nominal conditions to record Silent Data Corruptions (SDCs),
- monitors the exposed corrected and uncorrected errors from the hardware platform's error reporting mechanisms
- recognizes when the system is unresponsive to restore it automatically,
- monitors system failures (crash reports, kernel hangs, etc.),
- determines the safe, unsafe and non-operating voltage regions for each application for all frequencies, and
- performs massive repeated executions of the same configuration.

4 Experimental Setup

4.1 Cloud and Edge Architecture

To evaluate Cloud and Edge deployments we use real network traces from both Amazon servers and local servers, respectively. Specifically, for the Cloud evaluation we profile network latency by pinging for one week an Amazon server located in London [56]. London is chosen because, after profiling several sites, it has the lowest network latency. So, it would be obviously a better choice in the Cloud setup. On the other hand, for the Edge evaluation, we profiled for one week the network latency of a server hosted in the same building.

For the compute time, in this analysis we measure the time spend in the ARM processors by emulating the processing board of WDoS application, as shown in Figure 3, because this is

TABLE 2
Edge and Cloud Configurations

	Cloud Configuration	Edge Configuration
Total Number of Sensors	8000	
Number of Locations	1	100
Cost of Cooling	3.5 $/kwh	0.019 $/kwh
Cost of electricity	0.08$/kwh	0.07671$/kwh
Network per rack [49]	5000$	1190$
Maintenance salary per rack	208$	8.68$
Power Usage Effectiveness (PUE) [50]	1.3	1.1
Mean time to replace a faulty component	1 h	24 h

the most critical processing component of the application. For the Cloud versus Edge evaluation we considered that the same type of processing board is used either in the Cloud or in the Edge, in order to make them comparable. To estimate the total QoS, we convolute the distributions of network latency results for one week and the compute time results of the processing board.

Several sensors can be attached to each of the processing boards. For this analysis we assume that up to 8 sensors can be attached, equal to the number of cores in the Processing Board CPU. Each sensor corresponds to one application instance. So, the processing board can collocate a maximum of eight instances. Attaching more sensors per server was not feasible due to the excessively high compute time that leads to QoS violations.

4.2 TCO Input Parameters

The main TCO input parameters used in this analysis are shown in Table 2. We consider 8000 sensors, in total, that cover an area of 200,000 m2. This is representative of a large public building. In the Edge deployment we assume that the micro-servers are distributed in 100 different locations within the building, with 80 sensors located nearby location. On the other hand, for the Cloud deployment all servers are assumed to be placed in one location. For the centralized Cloud, cooling cost, cost of electricity, network cost and the maintenance personnel salary per rack are higher than the Edge configuration. The cooling cost of the Cloud deployment is expected to be much higher than the cooling cost of the Edge deployment as a centralized large datacenter requires much more sophisticated and expensive cooling infrastructure. Regarding electricity costs, the more devices that the centralized Cloud hosts, the higher power budget is needed and the higher costs are paid to the electricity provider, as this large absolute peak power is reflected in the cost of the electricity. Also, network per rack cost is related to the servers that are placed per rack. For Cloud configuration we assumed racks of 42 servers, whereas for Edge deployments we assumed at most racks with 10 servers, due to the power constrained Edge facilities. In addition, Cloud Power Usage Effectiveness (PUE), is significantly more that the Edge configuration, since cooling is a power-hungry system and uses a non-negligible fraction of the datacenters power. Finally, for Mean Time to Repair (MTTR) of a faulty component we assume that an Edge faulty component can be replaced within 24 hours whereas in a Cloud configuration, the faulty component can be replaced within 1 hour. This difference is primarily justified by considering the geographical distribution of Edge facilities. The MTTR difference is also reflected in the maintenance personnel salary per rack.

4.3 Micro-Server Architecture

The server that we use to characterize the WDoS application on, is a state-of-the-art 64-bit ARM based Server-on-Chip, is Applied Micro's (now Ampere Computing) X-Gene 2. X-Gene 2 platform provides knobs for undervolting the various components that are explored. The micro-server consists of eight 64-bit ARMv8-

TABLE 3
Nominal and Efficient Operating Settings

	Nominal Settings	Efficient Settings
PMD Voltage	980	920
SoC Voltage	950	870
DRAM Voltage	1500	1428
DRAM Refresh Rate in ms	78	2783

compliant cores running at 2.4 GHz, grouped in 4 Processor Modules (PMD), which have a separate 32 KB LI instruction cache and 32 KB L1 data cache for each core and a 256 KB unified L2 cache for each PMD. There is also an 8 MB L3 cache which is shared across the whole chip (all 8 cores). There are 4 available memory channels with DDR3 memories.

The X-Gene 2 provides access to a separate Scalable Lightweight Intelligent Management Processor (SLIMpro), a special management core, which is used to boot the system and provide access to on-board monitors for measuring the temperature and power of the SOC and DRAM. The SLIMpro, also reports to the Linux kernel all errors corrected or detected by the provided error-correcting codes (ECC) and the parity. Finally, SLIMpro has configuration parameters of the Memory Controller Units (MCUs), such as refresh period (TREFP). The server runs a fully-fledged OS based on CentOS 7 with the default Linux kernel 4.3.0 for ARMv8 and supports 4KB and 64KB pages.

After characterizing it we choose the voltage levels that do not affect the availability of the system, called safe margins. These margins are used to evaluate the efficient Edge deployment.

5 Characterization Results

The WDoS application running on the Processing Board and its dependencies have been ported and tested on the X-Gene 2 host platform by using the characterization framework. The generated results need to be deterministic and repeatable. This is mandatory because in order to detect SDCs (Silent Data Corruption) among different runs, the output needs to be compared

TABLE 4
Characterization results running WDoS application with Normal Setting and Efficient Settings

Running Dos Application with Normal Settings						Running Dos Application with Efficient Settings					
	Idle	1 Instance	2 Instances	4 Instances	8 Instances		Idle	1 Instance	2 Instances	4 Instances	8 Instances
Peak Server Power		67.91	72.81	80.66	88.26	Peak Server Power		63.53	67.64	74.11	80.86
Avg. Server Power	57.43	64.70	69.18	75.79	80.77	Avg. Server Power	54.19	61.21	64.90	70.77	75.17
Avg. PMD Temperature	43.38	59.96	62.15	65.36	69.01	Avg. PMD Temperature	42.98	57.71	58.77	60.81	64.68
Avg. SoC Temperature	44.61	59.41	60.49	62.18	65.17	Avg. SoC Temperature	44.05	58.65	59.29	60.19	62.97
Avg. DRAM Temperature	47.58	62.45	63.06	64.05	66.82	Avg. DRAM Temperature	47.71	62.51	62.91	63.61	66.17

and verified. For this purpose, data sets obtained from recording real life jammer signals, were used as inputs for the application tests. We run the application with various numbers of instances to obtain the trends of the effectiveness. The characterization process reveals the lowest operating limits that achieve the highest power savings without compromising the availability of the system as shown in Table 3. Table 4, shows the characterization results running with the nominal and the most energy efficient settings. As can be seen from the Table 4, the peak and average power can be decreased by 8 and 5 watts, respectively. This reduction corresponds to around 9% savings of the processor power. Moreover, temperature can be decreased by about 3 degrees Celsius and thus, help reduce the need for cooling and can help lifetime reliability.

The findings of the characterization in this Section are used as inputs for the TCO analysis.

6 TCO Analysis

This Section reports the TCO analysis of the Edge compared to the Cloud deployment and examines the benefit from more energy efficient micro-servers in the Edge.

6.1 Selection of the number of Instances in Edge and Cloud Deployments

We first evaluate the end-to-end latency of Cloud and Edge deployments in order to select the appropriate number of instances to run in the processing board and at the same time not violate the QoS constraints of the application.

Figures 6(a) and 6(b) show the cumulative distribution of the Cloud and Edge End-to-End latency for different number of instances, respectively. As the Figure 6(a) shows running 8 instances per server, at the Cloud, is not feasible because this configuration violates the QoS requirement of 400ms for the 90th percentile of the decisions. So, the preferable configuration is the one that uses 4 instances per processing board that corresponds to 4 sensors per board. On the other hand, the QoS results of Edge deployment show that the processing board can simultaneously run 8 instances without violating the QoS requirements of the application. This happens due to the lower network latency of the Edge deployment.

For the rest of the results we use maximum 4 instances per processing board for the Cloud deployment and maximum 8 instances per processing board for the Edge deployment, as well. This requires the use of 2000 servers for the centralized Cloud deployment and 1000 servers for the distributed Edge deployment, in total. In addition, for each Edge locations, there is a placement of 10 servers per location that we assume can operate within the available power at each facility.

6.2 Edge Versus Cloud TCO

Figure 7, illustrates the Normalized TCO breakdown results with Edge Deployment, for both the Edge and the Cloud. As the Figure shows, the Cloud TCO is 2.13 times higher than the Edge TCO. This happens, as the breakdown shows, due to the double number of servers that are needed for hosting the total number of sensors to cover the specific area. The obvious difference in the server cost explains the large TCO difference. Also, as Figure 7 shows, the Cloud deployment consumes exactly 2.5 times more power than the Edge deployment. So, except of the double server number, Cloud consumes more power due to the cooling power consumption and the PUE. Additionally, maintenance cost is also around 3.25 times higher in the Cloud than in the Edge deployment. This is due to the lower replacement frequency of the faulty components in the Edge (MTTR). Measuring the availability, we observed that the Cloud can provide four nines of availability (0.9999), whereas Edge provides only two nines of availability (0.99), which still does not violate the availability requirement of the application.

This analysis highlights that WDoS application can be deployed more efficiently in the Edge than in the Cloud.

6.3 TCO and Area Coverage results for Efficient Edge and Normal Edge deployments

Figure 8, illustrates an investigation as a function of per Edge site power budgets in Watts (x-axis) and the area in square meters (m^2) that needs to be covered by jamming detector sensors (y-axes). All the presented graphs in Figure 8 show the ratio of Efficient Edge (Edge servers that operate with energy-efficient settings) over Normal Edge (Edge servers that operate with nominal settings). The arrows next to each graph that show which direction represents improvement. Also, we use numbered labels for better explanation of the trends in five cases. In all graphs of Figure 8 the labels represent the same case. The first result, in Figure 8(a), shows the metric that we optimize, the TCO over Area coverage. The 3D representation shows that the TCO over the area coverage exposes a sharp increase of the Efficient Edge over the Normal Edge server by reaching the 100% around 73 Watts power budgets for all the cases with label 1.

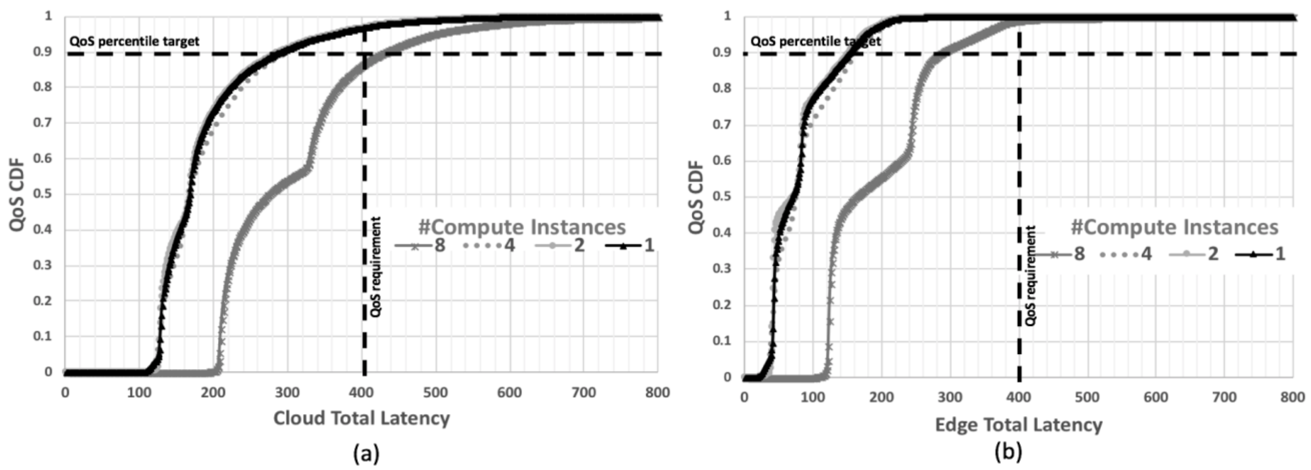

Fig. 6. QoS results for different number of instances of the WDoS application running in the Cloud (a) and in the Edge (b)

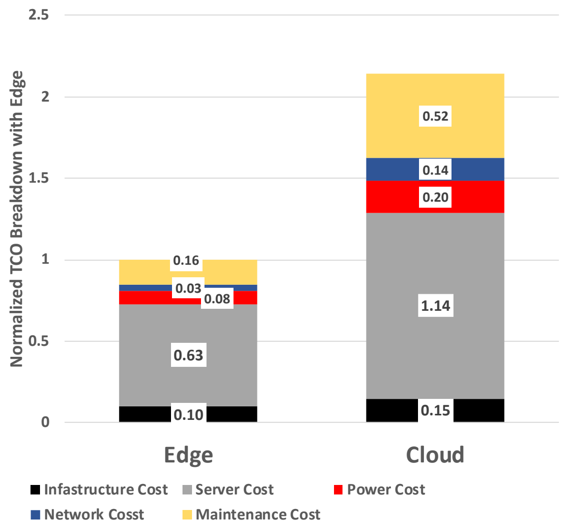

Fig. 7. TCO results for Edge and Cloud deployments

For this case the normal Edge setup cannot even host one server because the individual server consumes more power than the provided power budget. As the power budget increases, the benefit of the Efficient Edge remains for several power budgets by approaching 60% (label 2), 40% (label 3) and 20% (label 4). After the 400-Watt power budget, the TCO over area coverage has very small difference, around 2% (label 5). The trends in the graph show peaks and valleys due to the discrete power that is needed to fit an extra server, i.e. when power allows the Efficient and Normal Edge to have the same servers the benefits drop (label 5) and otherwise are high (labels 1, 2, 3, 4). The relative benefits drop as we increase power budget since with higher budget more serves are used and the relative impact of an extra server decreases. But power is a tight resource on the Edge. These benefits shown in Figure 8(a) will be more pronounced with more efficient power servers. The trends observed in Figure 8(a) can be better understood by examining the rest of the graphs in Figure 8. Figure 8(b) shows the area coverage and clearly shows that Efficient Edge can provide always better area coverage and, in some cases, considerably more. These cases correspond to the peak values of Figure 8(a) (labels 1, 2, 3, 4). Also, Figure 8(c), shows the TCO of the (energy-)Efficient Edge over the Normal Edge. This graph shows some points that the Efficient Edge has higher TCO than Normal Edge (for example labels 1, 3). This can be explained by observing the corresponding labels in Figure 8(d) that present the number of servers that can be placed in the deployment. As seen there is a peak on the Figure 8(d) showing that the Efficient Edge uses double number of servers (labels 1, 3). This costs in TCO but provides better area coverage (Figure 8(b), labels 1, 3). On the other hand, these extra servers increase the total power consumption of the deployment, as shown in Figure 8(e) (labels 1, 3). Except these peak numbers in increased power consumption the rest situations of Efficient Edge provide around 9% power savings as compared to the Normal Edge (label 5). The last Figure 8(f), shows the total number of instances placed per server. The number of instances that the Efficient

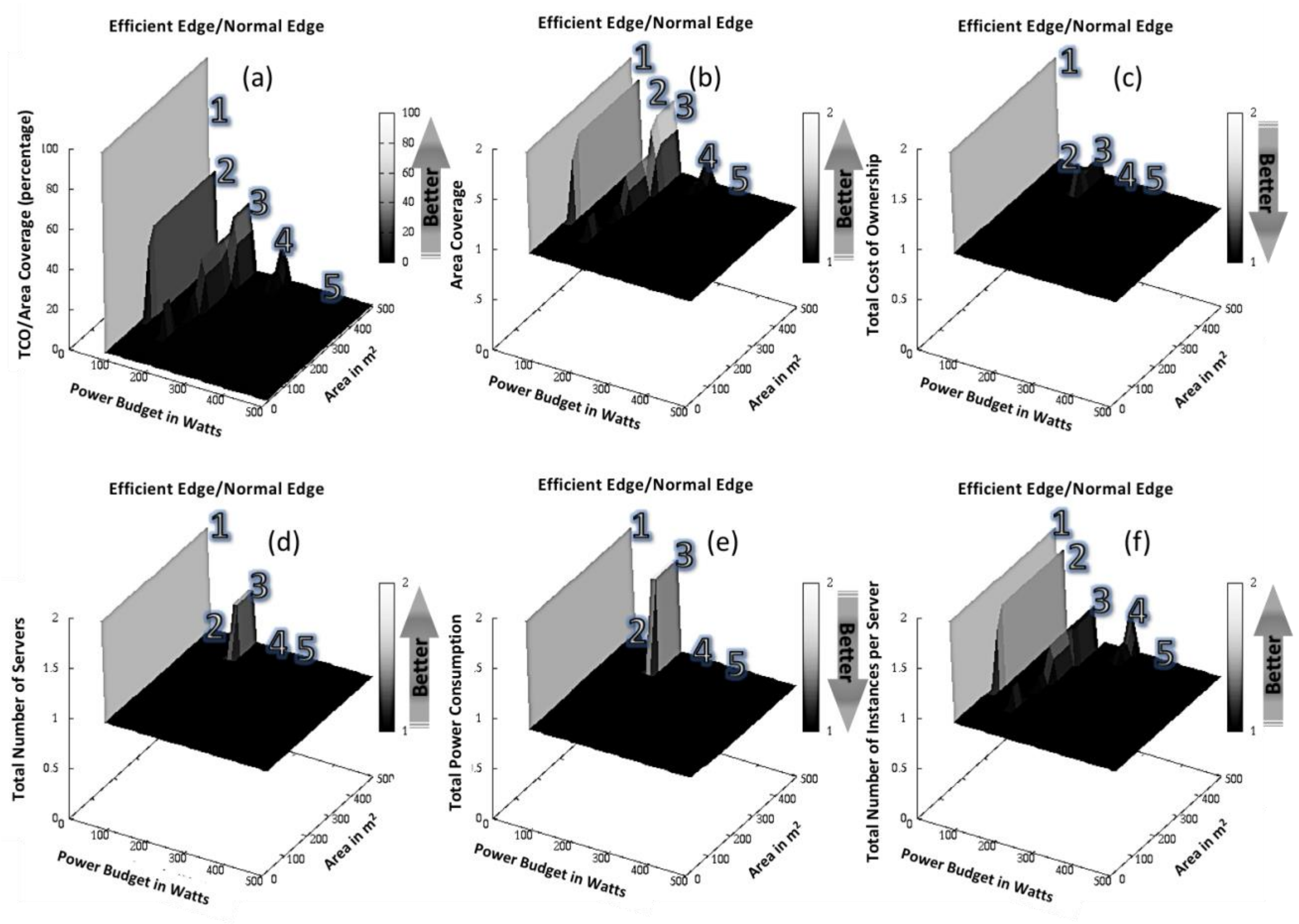

Fig. 8. Efficient Edge over Normal Edge results in (a) TCO over Area Coverage, (b) Area Coverage, (c) Total Cost of Ownership, (d) Total Number of Servers that are placed in the deployments, (e) Total Power Consumption, and (f) Total Number of Instances per server.

Edge can place in a server is significantly more than the Normal Edge, especially in the peak points of Figure 8(a) (labels 1, 2, 3, 4). Note that the number of servers and the number of instances trends directly correlate with the area coverage trend, in Figure 8(c). This analysis underlines the significance from operating more energy-efficiently on the edge and provides motivation for exploring additional means to increase efficiency on the Edge.

The experimental results, presented in the paper, clearly indicate the importance of having power efficiency in the Edge. This observation comes from the analysis of the TCO over area coverage optimization metric that is limited from the power budgeted Edge facility.

As far as we know, this is the first time that all these parameters are explored and analyzed together for evaluation of an IoT application, for both Edge and Cloud deployments. TCO/Area coverage is a useful metric for IoT evaluation and we strongly advocate its use for future IoT application evaluations.

7 Conclusions

This paper articulates that Edge servers need to be energy-efficient to allow under tight Edge power constraints to deploy as many of them at a facility since more servers enable processing data from more sensors of IoT applications. To this end, we develop a framework to understand the limits of operation revealing possible extended operation points at the Edge in order to make it more power efficient and finally, we evaluate the gains of the Efficient Edge deployment in a TCO compared to the Normal Edge and Cloud deployments. Our results, illustrate that Edge computing is more TCO efficient than the Cloud for the jamming detection application used in this study. Also, the results show that by making Edge more power efficient, approximately by 9%, we can achieve in many situations considerable gains in the TCO/Area-coverage metric. This work provides a clear motivation for even more power-efficient solutions at the Edge and the use of such evaluation metrics.

Acknowledgment

This work is supported by the H2020 Framework Program of the European Union through the UniServer Project (Grant Agreement 688540) – http://www.uniserver2020.eu.

References

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013–2018. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-indexvni/white_paper_c11-520862.pdf

[2] Bonomi, Flavio, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. "Fog computing and its role in the internet of things." In Proceedings of the first edition of the MCC workshop on Mobile Cloud computing, pp. 13-16. ACM, 2012

[3] Bonomi F. "Connected vehicles, the internet of things, and fog computing." The Eighth ACM Int. Workshop on VehiculAr InterNETworking VANET, Las Vegas, USA, 2011.

[4] Shekhar, Shashank, Ajay Dev Chhokra, Anirban Bhattacharjee, Guillaume Aupy, and Aniruddha Gokhale. "INDICES: exploiting Edge resources for performance-aware Cloud-hosted services." In Fog and Edge Computing (ICFEC), 2017 IEEE 1st International Conference on, pp. 75-80. IEEE, 2017.

[5] Tong, Liang, Yong Li, and Wei Gao. "A hierarchical Edge Cloud architecture for mobile computing." INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE. IEEE, 2016.

[6] El-Sayed, Hesham, Sharmi Sankar, Mukesh Prasad, Deepak Puthal, Akshansh Gupta, Manoranjan Mohanty, and Chin-Teng Lin. "Edge of things: the big picture on the integration of Edge, IoT and the Cloud in a distributed computing environment." IEEE Access 6 (2018): 1706-1717.

[7] Roman, Rodrigo, Javier Lopez, and Masahiro Mambo. "Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges." Future Generation Computer Systems 78 (2018): 680-698.

[8] FCC public notice 202/418-0500, January 27, 2015 Enforcement Advisory No. 2015-01

[9] C. Lefurgy, X. Wang, and M. Ware, “Power capping: A pre- lude to power shifting,” Cluster Computing, vol. 11, no. 2, 2008.

[10] Luiz Andre Barroso and Urs Holzle, The Datacenter as a Computer. Morgan Claypool, 2009.

[11] P. Ranganathan, P. Leech, D. Irwin, and J. Chase, “Ensemble- level power management for dense blade servers,” in Proceed- ings of the 33rd Annual International Symposium on Com- puter Architecture (ISCA), 2006.

[12] X. Wang, M. Chen, C. Lefurgy, and T. W. Keller, “SHIP: Scal- able hierarchical power control for large-scale data centers,” in Proceedings of the 18th International Conference on Parallel Architectures and Compilation Techniques (PACT), 2009.

[13] M. Jarschel, D. Schlosser, S. Scheuring, and T. Hoßfeld, “Gaming in the Clouds: Qoe and the users perspective,” Mathematical and Computer Modelling, vol. 57, no. 11, pp. 2883–2894, 2013.

[14] R. Kohavi, R. M. Henne, and D. Sommerfield, “Practical guide to controlled experiments on the web: listen to your customers not to the hippo,” in 13th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2007, pp. 959–967.

[15] Y. A. Wang, C. Huang, J. Li, and K. W. Ross, “Estimating the performance of hypothetical Cloud service deployments: A measurement-based approach,” in INFOCOM, 2011 Proceedings IEEE. IEEE, 2011, pp. 2372–2380.

[16] Luan, Tom H., Longxiang Gao, Zhi Li, Yang Xiang, Guiyi We, and Limin Sun. "A view of fog computing from networking perspective." ArXivPrepr. ArXiv160201509 (2016).

[17] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, “The Case for VM-Based Cloudlets in Mobile Computing,” Pervasive Computing, IEEE, vol. 8, no. 4, pp. 14–23, 2009

[18] V. Bahl, “Cloud 2020: Emergence of micro data centers (Cloudlets) for latency sensitive computing (keynote),” Middleware 2015, 2015.

[19] Andrew Froehlich, “How Edge Computing Compares with Cloud Computing”, Networking Computing Blog, 2018

[20] Chang, Yu-Shuo, and Shih-Hao Hung. "Developing Collaborative Applications with Mobile Cloud-A Case Study of Speech Recognition." J. Internet Serv. Inf. Secur. 1.1 (2011): 18-3

[21] Chang, Hyunseok, et al. "Bringing the cloud to the edge." Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on. IEEE, 2014..

[22] El-Sayed, Hesham, et al. "Edge of things: the big picture on the integration of edge, IoT and the cloud in a distributed computing environment." IEEE Access 6 (2018): 1706-1717.

[23] Clinch, Sarah, Jan Harkes, Adrian Friday, Nigel Davies, and Mahadev Satyanarayanan. "How close is close enough? Understanding the role of Cloudlets in supporting display appropriation by mobile users." In Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on, pp. 122-127. IEEE, 2012.

[24] Chang, Yu-Shuo, and Shih-Hao Hung. "Developing Collaborative Applications with Mobile Cloud-A Case Study of Speech Recognition." J. Internet Serv. Inf. Secur. 1, no. 1 (2011): 18-36.

[25] Fesehaye, Debessay, Yunlong Gao, Klara Nahrstedt, and Guijun Wang. "Impact of Cloudlets on interactive mobile Cloud applications." In Enterprise Distributed Object Computing Conference (EDOC), 2012 IEEE 16th International, pp. 123-132. IEEE, 2012.

[26] Shekhar, Shashank, and Aniruddha Gokhale. "Dynamic resource management across Cloud-Edge resources for performance-sensitive applications." In Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, pp. 707-710. IEEE Press, 2017.

[27] Powers, Nathaniel, Alexander Alling, Kiara Osolinsky, Tolga Soyata, Meng Zhu, Haoliang Wang, He Ba, Wendi Heinzelman, Jiye Shi, and Minseok Kwon. "The Cloudlet accelerator: Bringing mobile-Cloud face recognition into real-time." In Globecom Workshops (GC Wkshps), 2015 IEEE, pp. 1-7. IEEE, 2015.

[28] Soyata, Tolga, Rajani Muraleedharan, Colin Funai, Minseok Kwon, and Wendi Heinzelman. "Cloud-vision: Real-time face recognition using a mobile-Cloudlet-Cloud acceleration architecture."

In Computers and communications (ISCC), 2012 IEEE symposium on, pp. 000059-000066. IEEE, 2012.

[29] http://road.cc/content/news/76427-thousands-using-gps-jammers-disguise-over-long-hours-or-stolen-cars

[30] https://www.cnet.com/news/truck-driver-has-gps-jammer-accidentally-jams-newark-airport/

[31] https://www.pdaelectronics.com/index.php

[32] http://www.suresafe.com.tw/Signal-Jammer-Detector.html

[33] https://simplisafe.com/home-3

[34] https://www.forbes.com/sites/marcwebertobias/2015/01/29/this-popular-wireless-alarm-system-can-be-hacked-with-a-magnet-and-scotch-tape/#6b466ec450de

[35] http://www.chronos.co.uk/en/news-and-pr/news-page/1216-chronos-enables-exelis-to-develop-new-capability-in-gps-interference-detection-and-geolocation

[36] http://www.chronos.co.uk/files/pdfs/pres/2010/GPSJamming/Generic_Briefing_Presentation.pdf

[37] https://www.gps.gov/cgsic/meetings/2014/curry.pdf

[38] http://www.sekotech.com/gsm-3g-4g-jammer-detector/st-171-gsm-3g-4g-gps-jammer-detector.html

[39] http://www.pki-electronic.com/products/jamming-systems/jammer-detector/

[40] https://play.google.com/store/apps/details?id=com.microcadsystems.serge.jammerdetector&hl=en

[41] Pelechrinis, Konstantinos, Marios Iliofotou, and Srikanth V. Krishnamurthy. "Denial of service attacks in wireless networks: The case of jammers." IEEE Communications surveys & tutorials 13.2 (2011): 245-257.

[42] Xu, Wenyuan, et al. "The feasibility of launching and detecting jamming attacks in wireless networks." Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing. ACM, 2005.

[43] D. Hardy, M. Kleanthous, I. Sideris, A. Saidi, E. Ozer, and Y. Sazeides, “An analytical framework for estimating tco and exploring data center design space,” in International Symposium on Performance Analysis of Systems and Software, pp. 54–63, 2013.

[44] C. Patel and A. Shah, “Cost model for planning, development and operation of a data center,” HP TR, 2005.

[45] J. Karidis, J. E. Moreira, and J. Moreno, “True value: assessing and optimizing the cost of computing at the data center level,” in 6th ACM Conference on Computing Frontiers, pp. 185–192, 2009.

[46] J. Moore, J. Chase, P. Ranganathan, and R. Sharma “Making scheduling ”cool”: temperature-aware workload placement in data centers,” in Annual Conference on USENIX, pp. 5–5, 2005.

[47] J. Koomey, K. Brill, P. Turner, J. Stanley, and B. Taylor, “A simple model for determining true total cost of ownership for data centers,” White Paper, Uptime Institute, 2007.

[48] K. V. Vishwanath, A. Greenberg, and D. A. Reed, “Modular data centers: how to design them?,” in 1st Workshop on Large-Scale System and Application Performance, pp. 3–10, 2009.

[49] Nikolaou, Panagiota, Yiannakis Sazeides, Lorena Ndreu, and Marios Kleanthous. "Modeling the implications of DRAM failures and protection techniques on datacenter TCO." In Proceedings of the 48th International Symposium on Microarchitecture, pp. 572-584. ACM, 2015.

[50] “Intel cpu configuration,” http://www.rect.coreto- europe.com/rack-server/1u- intel- server/2428- short- 1u- intel- single- cpu- rack-server.html.

[51] L. A. Barroso, J. Clidaras, and U. Ho ̈lzle, “The datacenter as a computer: An introduction to the design of warehouse-scale machines,” Synthesis lectures on Computer Architecture, pp. 1–154, 2013.

[52] G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, D. Gizopoulos, G. Favor, K. Sankaran and S. Das, “A System-Level Voltage/Frequency Scaling Characterization Framework for Multicore CPUs”, IEEE Silicon Errors in Logic – System Effects (SELSE 2017), Boston, MA, USA, March 2017.

[53] G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, D. Gizopoulos, P. Lawthers, and S. Das, “Harnessing Voltage Margins for Energy Efficiency in Multicore CPUs”, IEEE/ACM International Symposium on Microarchitecture (MICRO 2017), Cambridge, MA, USA, October 2017.

[54] G. Karakonstantis, K. Tovletoglou, L. Mukhanov, H. Vandierendonck, D. S. Nikolopoulos, P. Lawthers, P. Koutsovasilis, M. Maroudas, C. D. Antonopoulos, C. Kalogirou, N. Bellas, S. Lalis, S. Venugopal, A. Prat-Perez, A. Lampropulos, M. Kleanthous, A. Diavastos, Z. Hadjilambrou, P. Nikolaou, Y. Sazeides, P. Trancoso, G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, D. Gizopoulos, and S. Das, “An Energy-Efficient and Error-Resilient Server Ecosystem Exceeding Conservative Scaling Limits”, ACM/IEEE Design, Automation, and Test in Europe (DATE 2018), Dresden, Germany, March 2018.

[55] K. Tovletoglou, L. Mukhanov, G. Karakonstantis, A. Chatzidimitriou, G. Papadimitriou, M. Kaliorakis, D. Gizopoulos, Z. Hadjilambrou, Y. Sazeides, A. Lampropulos, S. Das, P. Vo, “Measuring and Exploiting Guardbands of Server-Grade ARMv8 CPU Cores and DRAMs”, IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2018), Luxembourg, June 2018.

[56] Amazon, “Amazon Web Services, EC2 Reachability Test”, http://ec2-reachability.amazonaws.com

[57] Ballas, Camille, et al. "Performance of video processing at the edge for crowd-monitoring applications." (2018).

[58] WorldSensing, https://www.worldsensing.com