INTELLIGENT SURVEILLANCE SYSTEMS ON THE INTERNET OF THINGS BASED ON SECURE APPLICATIONS WITH THE IBM CLOUD PLATFORM

*Ioannis Adamopoulos (1), Aristidis Ilias (1), Christos Makris (1), Yannis C. Stamatiou (1, 2) **

(1) Computer Engineering and Informatics Department; (2) Department of Business Administration; University of Patras and CTI-"Diophantus" Greece.

* Corresponding author, e-mail: stamatiu@ceid.upatras.gr

Abstract: The Internet of Things (IoT) is one of the most fast-evolving and most pervasive technological advancements of the last decade. Driven by IoT's promise for an interconnected world for all people, through a shared Internet-based infrastructure, more and more businesses expand their activities to embrace, competitively, this new concept. More importantly, the ubiquitous, highly granular, and of increasing interconnection and computational capabilities nature of the IoT fabric creates a unique ecosystem for developing powerful, intelligent applications not limited to simply gathering and analyzing environmental and people data. This potential combines the IoT's physical capabilities with the achievements of the more mature discipline of Artificial Intelligence (AI). However, there is still a large number of businesses with no clear plans of investing in the IoT-plus-AI potential, citing multiple obstacles mainly related to the difficulty of understanding, adapting, and developing advanced AI techniques. Our goal is to show that advanced AI functionalities can, now, be easily implemented and operated through existing IoT platforms, by developing an intelligent, remote, surveillance application based on IBM's Cloud and Watson platforms as well as the Raspberry Zero W low-cost IoT device.

Key words: Internet of Things, Artificial Intelligence, Machine Learning.

1. INTRODUCTION

While most measures taken to ensure a system's security focus on the mitigation of threats such as cyberattacks, malware and cyber espionage, the physical security of the system is often neglected or delegated to less sophisticated technologies. Although, in some cases, the system can be isolated in a controlled environment

under complete control (e.g. bank vaults), there are cases in which this isolation is not possible, especially when outdoor establishments are involved. Consequently, the need arises for security monitoring using intelligent, specialized, surveillance equipment of sophisticated design.

The traditional surveillance systems depend on human operators for detecting potential threats and require a significant amount of storage to record hours of video footage. Due to the high cost involved in such systems, it is useful to examine alternative solutions to the issue at hand. Smart surveillance systems could instead be employed to automate the detection of potential trespassers in a protected area, thus eliminating human errors, minimizing the storage requirements and lowering the overall cost. The project's goal is the rapid development, towards achieving reduced-time-to-market, of such a system within the Internet of Things context using the machine learning and AI functionalities provided by the IBM Cloud platform.

2. IBM's CLOUD

The *IBM Cloud*, formerly known as IBM Bluemix, is a platform provided by IBM, that combines PaaS (Platform as a Service), IaaS (Infrastructure as a Service) and SaaS (Software as a Service) to enable rapid application deployment and reduced time-to-market (see, e.g., [1]). The platform enables the development of applications in a variety of programming languages, including Java, Python, Swift and Node.js, while it also supports the flow-based development tool Node-RED [2].

2.1. Node-RED

Node-RED is a programming tool based on Node.js developed by IBM in 2013. It follows a flow-based programming model and enables the interconnection of devices, APIs and online services. An application's behaviour can be described in a flow as a network of black boxes referred to as nodes. Each node receives data, processes or modifies them and forwards them to the next node. By default, Node-RED provides a significant number of available nodes to support various operations and IBM Cloud services. In addition, a user can include community-made nodes or create custom nodes using JavaScript and HTML (see, e.g., [3]).

2.2. Services

The IBM Cloud platform also provides numerous services which can be connected to applications to provide additional functionality. These services cover a wide range of functions, from data analytics and databases to artificial intelligence and blockchains. The current project utilizes numerous services, including the Cloudant NoSQL DB (see [4]) to store the events, the Object Storage service (see [5]) to store the captured images and the Watson Visual Recognition service, as described in [6], to classify the photographs.

One of the most important services provided is the Watson IoT service (see [7]), which enables the registration and management of devices, while it also handles the

communication between the device and the cloud. The devices communicate with the service using the MQTT (Message Queuing Telemetry Transport) protocol (see [8] for a description), which is a lightweight protocol ideal for communication in cases where the available bandwidth is limited and the communications link is of poor quality. MQTT employs a publish-subscribe communication model in which the messages are not sent directly to the receivers, but are published in a topic and sent to a broker which is then responsible for forwarding them to any hosts subscribed to the topic. The data sent through the MQTT protocol can also be encrypted using TLS.

3. HARDWARE

The current project required a low-cost, low-consumption device with the purpose of controlling the sensors and interfacing with the IBM Cloud platform. The application did not require significant processing power and thus a Raspberry Pi Zero W device was deemed most suitable for the task.

3.1. Raspberry Pi Zero W

The Raspberry Pi Zero W device, as described in [9], is a low-cost single board microcomputer which includes a Broadcom BCM2835 SoC (System on Chip) and has the following specifications:

CPU	*ARM11 running at 1GHz*
RAM	*512MB*
Wireless	*2.4GHz 802.11n wireless LAN*
Bluetooth	*Bluetooth Classic 4.1 and Bluetooth LE*
Power	*5V, via USB connector*
Video & Audio	*1080P HD video & stereo audio via mini-HDMI connector*
Storage	*MicroSD card*
Output	*Micro USB*
GPIO	*40-pin GPIO*
Pins	*Run mode, unpopulated; RCA composite, unpopulated*
Camera	*Camera Serial Interface (CSI)*

Since the device was intended for use in a “headless” setup, in which there is no monitor or other user interface peripherals attached, the Raspbian Lite operating system was chosen in order to minimize power consumption. Raspbian Lite is based on Debian Linux and only provides a command line interface to the user.

3.2. PIR motion sensor

For the task of detecting motion in the area, a passive infrared (PIR) motion sensor was used, with a detection range of about 7 meters and a coverage area of a 100o cone. The motion sensor provides two distinct modes of operation, with one

being a retriggering mode and the other a non-retriggering mode (see [10]). Their difference lies in the fact that when the motion sensor operates in the retriggering mode, its output will stay high (3.3V) for as long as motion is being detected. By contrast, while in non-retriggering mode the sensor output will become high in the event of motion detection, but will inevitably change to low after a set amount of time, regardless of whether motion is still being captured. After becoming low, the output can then change back to high if motion is detected anew. The difference between the two modes can be seen in the following diagrams:

3.3. Camera

The camera chosen for the project was the official Raspberry Pi Camera Module v2, which has the ability to capture high resolution images of up to 8 Megapixels as well as video footage. The camera module is connected to the Raspberry Pi via the Camera Serial Interface and thus its power consumption is significantly lower than its USB counterparts. In addition, a lot of image capture parameters can be controlled with the use of the provided programming libraries (see, e.g., [11]).

4. VISUAL RECOGNITION

One of the most important aspects of the system is its ability to correctly identify people who enter the protected area, while at the same time filtering out noise. To that end, the process of classification was employed with the purpose of determining whether a particular image depicts an intruder entering the area.

4.1. Classification

The process of classification (see, e.g., [12]) is a machine learning task which involves the creation of a classification model (also known as a classifier) with the intention of using it to predict the classes relevant to an unknown object. The classes represent the categories the object to be classified may belong to. Classification is a two-step process, in which the first step involves training the classifier with the use of appropriate training sets, whereas in the second step the classifier is used to predict the class labels of new objects.

Classification is regarded as an instance of supervised learning due to the fact that the training sets consist of example pairs, with each object being accompanied by a set of relevant class labels. The classifier is trained using the data of the training sets as examples or counterexamples of the respective classes. Afterwards, it can receive unknown data as input to be classified into one (or more) of the known classes.

4.2. Evaluation metrics

When evaluating the performance of a classification model, the results of a test fall in one of these categories: (i) True Positive: A positive test result that was correctly labeled by the classifier (ii) False Positive: A negative result that was

incorrectly labeled as positive by the classifier (iii) True Negative: A negative result that was correctly labeled by the classifier (iv) False Negative: A positive result that was incorrectly labeled as negative by the classifier

There are various metrics suitable for evaluating a classifier's performance, with some of the most well-known being precision and recall (see, e.g., [13]). Precision is defined as the ratio of the number of objects classified correctly into a class to the total number of the objects classified into that class. As a more intuitive explanation, precision provides information on how many of the samples classified into a class truly belong to that class. Recall on the other hand, is defined as the ratio of the number of objects correctly classified into a class to the sum of these objects and those incorrectly deemed irrelevant by the classifier. Recall provides information on how many of the total relevant objects were classified into the class.

$$Precision = \frac{True\ Positive}{True\ Positive\ +\ False\ Positive} \tag{1}$$

$$Recall = \frac{True\ Positive}{True\ Positive\ +\ False\ Positive} \tag{2}$$

In the current project, high precision can be interpreted as a minimization of the number of false alarms. While high precision is desirable, given the system's function as a surveillance device, the correct identification of all potential threats is of greater importance and thus achieving high recall is essential.

Because of the trade-offs involved in the metrics described above, it is often difficult to compare different classifiers when one achieves a high precision value and the other achieves a high recall value. To remedy this difficulty, there is a number of metrics that combine both precision and recall.

One such metric is known as F-measure, or F1 score, and is defined as follows:

$$F - measure = \frac{2}{\frac{1}{recall} + \frac{1}{precision}} = 2 * \frac{recall * precision}{recall + precision} \tag{3}$$

F-measure weights both precision and recall equally and that is why it also known as Harmonic Mean. While F-measure can be used to judge the overall quality of a classifier and facilitate comparisons, it cannot be tailored to the needs of the current project.

As a means to overcome the aforementioned limitation imposed by F-measure, a similar combinational metric known as E-measure can be used. While E-measure also takes into consideration both precision and recall, the weights are not necessarily equal. The user can modify these weights by setting an appropriate value for the factor "b". The value of "b" must be positive or zero, with values lower than 1 indicating a greater interest in precision, while values greater than 1 indicate recall is more important in this task. In the case where b=1, the E-measure acts as the complement of the F-measure. The E-measure is defined as follows:

$$E - measure = 1 - \frac{1 + b^2}{\frac{b^2}{recall} + \frac{1}{precision}} \quad (4)$$

Looking at the above definition, it is apparent that lower E-measure values are preferable.

In addition to the above metrics, one can also use Accuracy as a way to describe how many of the total classifications were correct. Accuracy is defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

4.3. ROC curve

Given that the project involves the training of a binary classifier, a Receiver Operator Characteristic curve (see [14]) can also be used to depict the classifier's ability to differentiate between positive and negative samples. The curve is created by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at a number of threshold values. TPR and FPR are defined as:

$$TPR = Recall \quad (6)$$

$$FPR = \frac{False\ Positive}{Negative} = \frac{False\ Positive}{False\ Positive + True\ Negative} \quad (7)$$

The ROC curve of an optimal classifier will converge towards the upper left corner of the plot, i.e. towards the solid green line as shown in Figure 1, indicating an excellent ability to differentiate between positive and negative samples. By contrast, a bad classifier will have a ROC curve similar to the dashed blue line in Figure 1, where it cannot differentiate between positive and negative samples, effectively operating as a random predictor.

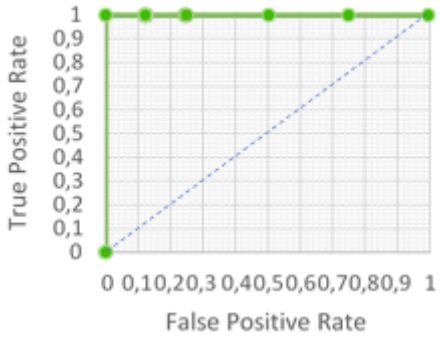


Figure 1. ROC curve

Another important metric resulting from ROC curves is the Area Under the Curve (AUC) which quantifies the results of a ROC curve facilitating easy

comparisons between different classifiers. The AUC's values can range from 1, in the case of an ideal classifier, to 0.5 in the case of the worst possible classifier.

5. APPLICATION

In this section we provide the details and the components of the implementation.

5.1. Python

For the needs of our project, a Python 2.7 script was developed with the purpose of interacting with the IBM Cloud and controlling the hardware modules. Its logic can be briefly described by the following flow diagram (Figure 2).

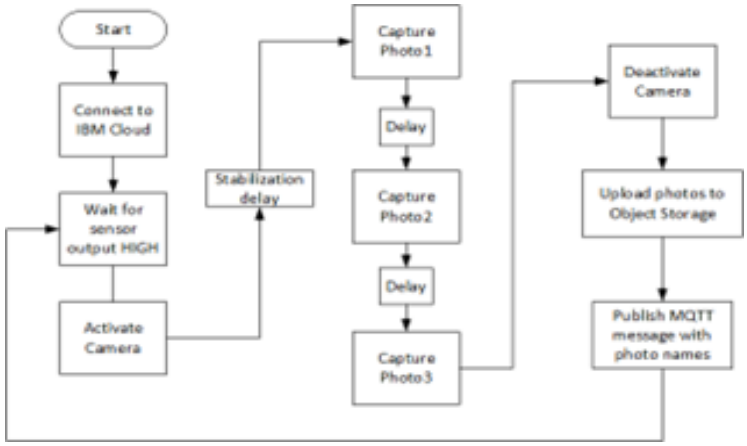


Figure 2. Flow diagram

The script begins by establishing a connection to the IBM Cloud platform and the awaits a positive edge on the motion sensor's output. When the sensor is triggered, the camera is activated and after a brief stabilization delay it captures 3 consecutive photos with a small delay of about 0.6s between them. The stabilization delay is necessary for the camera to automatically adjust to the scene and cannot be reduced below 1 second without impacting the quality of the photos. The motion sensor operates in the non-retriggering mode so that the photos are captured with edge-triggered events. This choice was made to avoid the potential waste of resources and increase in power consumption that the alternative approach of polling with retriggering would entail. In order to aid the logging of events, the photos contain a timestamp of the moment they were taken in their name.

After the photos are captured, the camera module is powered off to save energy and a connection is established with the IBM Swift Object Storage before uploading the photos. Following the completion of the upload, the python script publishes an event for every photo with the use of the MQTT protocol, including the names of the uploaded photos in the messages. The events are published with the default QoS

(QoS1), as it is important to deliver every message, but there is no problem with having it delivered multiple times in case of transmission errors. Finally, the script returns to the point where it awaits a new rising edge from the motion sensor's output.

5.2. IBM Cloud

The part of the application operating in the Cloud involved a number of services which were connected to the Node-Red service. The services involved are: Availability Monitoring, Cloudant NoSQL DB, Watson IoT, Object Storage and Watson Visual Recognition.

After creating an instance of the Watson IoT service, a new device type "RPiZeroW" was configured to represent the model of the device used. An instance of that device type was then created with a unique Device ID for the particular device. As a means to secure the communication between the service and the device, TLS with token authentication was used. TLS client certificates were not used, as they are not yet supported by IBM's Python API.

An instance of the IBM Object Storage was then created, along with a storage space named "RPi-imgstr" to store the photos. Similarly, an instance of the Cloudant NoSQL DB was created and within it a database named "rpi_storage" with the purpose of logging the events.

Finally, an instance of the IBM Watson Visual Recognition service was created, although it did not require additional setup through the IBM Cloud.

5.3. Visual Recognition classifier training

For the purpose of directly interacting with the Visual Recognition instance, the Curl tool [15] was used.

A simple python script was used to quickly capture a set of images to be used as positive examples by the classifier. The images depicted people moving around different areas in a variety of clothing and were captured in various lighting conditions in an attempt to avoid overfitting. A similar script was used for the capture of images to be used as negative examples, with the images not including any people, while in some cases they included animals. In some cases a few extra samples were created by applying simple pre-processing techniques (cropping, resizing etc.) on a selection of original samples, according to IBM' training guidelines (see [16]).

5.4. Node-RED

For the part of the application residing in the IBM Cloud, the following Node-RED flow was created – Figure 3.

The nodes in dark green are debug nodes and do not impact the execution of the program. The orange nodes are function nodes which execute JavaScript code, while the yellow node is a switch node which forwards the message to one of its outputs based on a set condition. The rest are specialized nodes, and their functions are described below, along with the flow's operation.

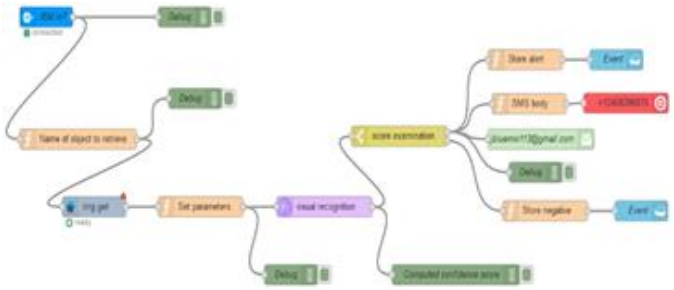


Figure 3. Node-RED flow

The blue node labelled "IBM IoT" represents the entry point of a message sent from a device and specifically from the Python script. After the message is received, the name of the photo stored in the Object Storage is retrieved from the message's payload. It is subsequently passed on to an Object Storage node as input, in order to retrieve the image itself. The image is then stored in a buffer as a section of the output message's payload. The message is forwarded to a function node which sets some parameters regarding the Visual Recognition API and is then sent to a specialized Visual Recognition node tasked with classifying the image. The parameters include the classifier's id and the decision threshold. In this case, the decision threshold is set to 0, as the classification results are processed in another node which provides a greater degree of control.

The Visual Recognition node classifies the image in the buffer according to the parameters set and appends the computed confidence score to the message. The message is then sent to a switch node which examines the confidence score and compares it to the decision threshold. If the confidence score is lower than the threshold, then a message is sent to Cloudant DB node to store a negative event. Should however the confidence score exceed the decision threshold, a positive event is registered in the Cloudant database, while at the same time an email with the image is sent to the user, along with an SMS alert. The SMS is sent with a Twilio node, while the email uses a dedicated built-in node.

6. RESULTS

6.1. Classification results

While there are several ways to evaluate the system's performance, the one chosen for this project is the following procedure, as described by IBM: (1) An image set labelled "L" is constructed. (2) The aforementioned set is split into two equal sets labelled "V" and "T" for validation and testing respectively. (3) The classifier is used to calculate the confidence score of the images in the "V" set and then a decision threshold "R" is chosen in a way as to maximize the chosen metric. (4) After a random subset "Q" of the "T" set is selected, the classifier is used to classify the images using the chosen decision threshold. (5) Step 4 is repeated with different "Q" subsets and the average classification accuracy is computed.

According to the procedure described above, the following "L" set was constructed using 32 images depicting a variety of areas with or without people, in various lighting conditions. The constructed image set is the one presented below, along with the "V" and "T" sets that resulted from the split (Figure 4).

Figure 4. Image sets

The classifier was then used to compute the confidence scores for the images in the "V" set, with the results presented in the following table:

Confidence score (Validation set)			
0.906	*0.401*	*0.143*	*0.896*
0.098	*0.055*	*0.919*	*0.058*
0.92	*0.018*	*0.173*	*0.154*
0.92	*0.9*	*0.598*	*0.92*

For the sake of completeness, the experiment was then performed with three distinct values of the decision threshold "R". The first value of "R" was chosen in a way that maximizes recall, while in the second case the decision threshold was chosen to maximize precision. Finally, an appropriate value of "R" was used in an attempt to find a middle ground between the two prior choices, maximizing both recall and precision simultaneously. For each of the aforementioned choices, the metrics of recall, precision, F-measure and E-measure are computed. Given that the intended use of the system is to perform surveillance on an area, the correctly classifying the positive samples is more important than minimizing the number of false positives. Thus, a value of b=3 was used when computing E-measure, indicating that recall is more important than precision for this experiment. The confidence scores of the images in the "V" set can be seen in the following plot, along with the three values of the decision threshold "R" (Figure 5).

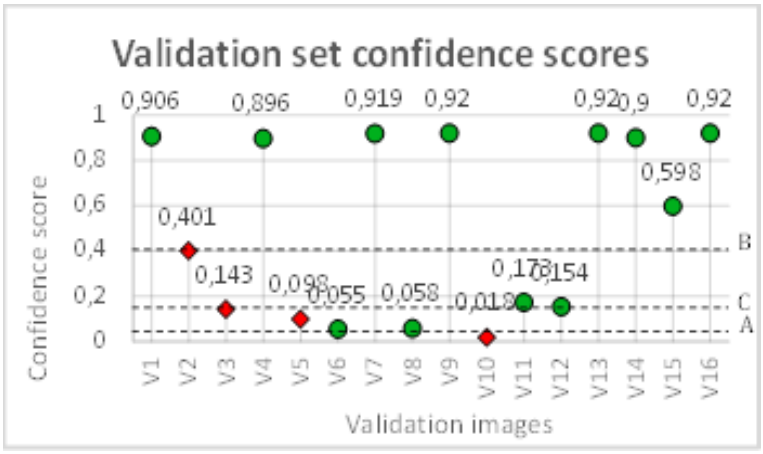


Figure 5. The V set of confidence scores

Case A: According to the computed confidence scores, maximizing recall requires a decision threshold lower than 0.055 in order to eliminate false negatives. Consequently, the value of selected in this case was R = 0.05.

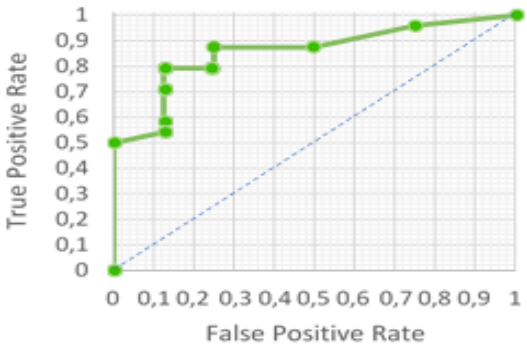
Case B: Maximizing precision requires the elimination of false positives. Thus, an appropriate value for the decision threshold would be R = 0.41.

Case C: When attempting to reach a middle ground between high recall and high accuracy, a compromise has to be made, as the two metrics cannot reach the maximum value of 1 simultaneously. To this end, a value of R = 0.15 was chosen.

The classifier then used the selected values of "R" to classify three random subsets of the "T" set, namely Q1, Q2, and Q3 in order to compute the average classification accuracy. The computed metrics for each value of R are presented in the following table:

	R = 0.05	***R = 0.41***	***R = 0.15***
Precision	*0.8*	*1*	*0.9091*
Recall	*1*	*0.6667*	*0.8333*
F-measure	*0.8889*	*0.8*	*0.9918*
E-measure (b=3)	*0.0244*	*0.3103*	*0.1599*
Average classification accuracy	*0.7083*	*0.85*	*0.85*

As a means of evaluation for the classifier's ability to distinguish between positive and negative samples, a ROC curve was constructed using the confidence scores computed for the images in the "L" set – Figure 6.



Area Under Curve (AUC) = 0.85677

Figure 6. ROC Curve, Area Under Curve (AUC) = 0.85677

6.2. Pilot operation

The system's correct operation was tested in a typical usage scenario. After the device was placed in an outdoors area it was powered on. A person was then introduced in the scene, along with a dog, which triggered the motion sensor. The dog was included in the picture to test the classifier's ability to recognize people even when negative samples are present in the image. The system then proceeded to capture and upload three images, while also notifying the user by SMS and email. In this case, the computed confidence score for all images was 0.92 and thus the images were classified as positive. The entire sequence for one of the three images can be seen in Figure 7 below.

Figure 7. Example of operation

6.3. Power consumption

The power consumption of the system was measured during various modes of operation, using th "Fluke 289 True-RMS Industrial Logging Multimeter". The system was powered by a portable 30000mA powerbank operating at 5V DC. The powerbank also includes a 1.5W solar panel. After measuring the voltage and current draw, the system's power consumption was calculated. The system showed small voltage drops during its cycle of operations, ranging from 5.115V DC at full load to 5.135V DC when idle. The average voltage measured was 5.1116V DC. When measuring the system's current draw, however, the differences in values were more pronounced, ranging from 111mA when idle, to 220mA when capturing photos. The system's current draw during a typical cycle of operation can be seen in the following graph. After the initial setup at the start of the script's execution, the motion sensor is activated twice, which triggers the capture and upload sequence of the script. Assuming average current draw of 150mA and average voltage of 5.1116V DC, the average power consumption of the system is $P = I \cdot V = 0.15A \cdot 5.1116V = 0.7667W$.

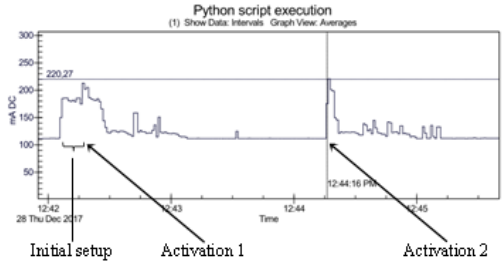


Figure 8. Current draw

The powerbank's energy would be: $E = Q \cdot V = 30\ Ah \cdot 5\ V = 150\ Wh$. Consequently, the system can operate (with the given power source) for a time period equal, approximately, to $T = E / P = 150 / 0.7667 \simeq 196\ hours$. Given that the power bank is also equipped with a 1.5W solar panel, this time period can be extended significantly, depending on the ambient light strength.

7. VULNERABILITIES

Despite improving on certain aspects of the traditional surveillance systems, the developed system is not entirely secure in its current, prototype, form. A number of physical and digital attacks can be used to bypass it and gain access to the area. Some of them are briefly described below.

7.1. Physical attacks

Due to the way the system operates, as well as the limitations of the deployed hardware, a potential attacker with knowledge of the system's structure can exploits the vulnerabilities to enter the area. Specifically, the camera's stabilization delay

provides a time window which may be sufficient for an attacker to move outside the camera's view. Additionally, the deployed PIR motion sensor has a range of 7m, a characteristic that makes it unsuitable for some environments. Both of these issues can be mitigated with proper positioning of the device, with respect to the environment's characteristics (e.g. obstacles, size etc.). Furthermore, the time it takes for the system to return to its ready state, after capturing images, depends on the connection's bandwidth. A slower connection means that the system remains inert for a prolonged period of time. Finally, an attacker could use jamming to disrupt the system's connection by injecting significant levels of noise to the frequencies used by the 802.11 standard, making it impossible for any hosts to establish reliable communication (see [17]).

7.2. Digital attacks

The attacks mentioned in this section target the system's communication with the IBM Cloud, removing its ability to detect threats and report potential incidents.

- *Deauthentication and disassociation attacks.* These attacks exploit a vulnerability in the 802.11 standard in order to send arbitrary management frames to hosts and disrupt their communication with the access point ([18]).
- *ARP cache poisoning.* Also known as ARP spoofing, this technique attempts to perform a man-in-the-middle attack by sending arbitrary ARP replies to impersonate other hosts. While in this case the traffic is encrypted and cannot be intercepted, the attack can still compromise the device's communication with the Cloud ([19]).

8. CONCLUSION – FUTURE WORK

In this work we investigated the issue of overcoming the obstacles that often deter businesses from implementing AI based IoT applications, by developing a smart surveillance solution using the IBM Cloud platform and its AI capabilities. The platform facilitated the rapid deployment of the targeted application and allowed the seamless integration of machine learning and AI techniques for the purpose of visual recognition, without the need to delve into the finer details of these advanced techniques or implement them from scratch requiring specialized personnel and disproportionate, for the type of the application we targeted, effort. Nevertheless, the results of the operation of the final system show that, given proper training, the system can reliably recognize potential intruders entering the protected area and notify the are owner with high accuracy and in real time. The system can be further improved by integrating specialized hardware to overcome its current limitations and by implementing TLS authentication using client certificates. This functionality is not yet supported by the Python API.

REFERENCES

[1] IBM, "*IBM Cloud technologies*" [Online]. Available: https://www.ibm.com/developerworks/cloud/library/cl-cloud-technology-basics/index.html

[2] IBM, "*IBM Cloud docs*" [Online]. Available: https://console.bluemix.net/docs/.

[3] "*Node-RED documentation*" [Online]. Available: https://nodered.org/docs.

[4] IBM, "*Cloudant NoSQL Database*" [Online]. Available: https://www.ibm.com/cloud/cloudant/details.

[5] IBM, "*IBM COS SDK for Python Documentation*" [Online]. Available: https://ibm.github.io/ibm-cos-sdk-python/.

[6] IBM, "*Watson Visual Recognition API reference*" [Online]. Available: https://www.ibm.com/watson/developercloud/visual-recognition.

[7] IBM, "*IoT Platform - Reference*" [Online]. Available: https://console.bluemix.net/docs/services/IoT/reference.

[8] (Edited by) Andrew Banks and Rahul Gupta, "[mqtt-v3.1.1-plus-errata01] MQTT Version 3.1.1 Plus Errata 01. OASIS Standard Incorporating Approved Errata 01. (Latest version: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html)," 10 December 2015. [Online]. Available: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.html.

[9] Raspberry Pi Foundation, "*Raspberry Pi Zero W specifications*" [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-zero-w/.

[10] Adafruit, "*Testing a PIR*" [Online]. Available: https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/testing-a-pir.

[11] "picamera documentation," [Online]. Available: http://picamera.readthedocs.io/en/release-1.12/.

[12] Zaki M. J., W. Meira Jr., *Data Mining and Analysis: Fundamental Concepts and Algorithms*, Cambridge University Press, 2014.

[13] Evangelopoulos, X. *New evaluation techniques for Information Retrieval*, Patras: University of Patras, 2015.

[14] Fawcett, T. An introduction to ROC analysis. *Pattern Recognition Letters*, vol. 27, p. 561–874, 2006.

[15] D. Stenberg, Everything Curl.

[16] "*Guidelines for training classifiers*," IBM, [Online]. Available: https://console.bluemix.net/docs/services/visual-recognition/customizing.html#guidelines-for-training-classifiers.

[17] Benslimane, A., Bouhorma, M. and Yakoubi, A. El. Analysis of Jamming effects on IEEE 802.11 Wireless Networks, in *IEEE International Conference on Communications (ICC)*, 2011.

[18] Bellardo, J., Savage, S. 802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions, in *USENIX Security Symposium*, 2003.

[19] Mangut, H. A., Al-Nemrat, A., Benzaid, C., Tawil, A.-R. H. ARP Cache Poisoning Mitigation and Forensics Investigation. in *The 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (IEEE TrustCom-15)*, Helsinki, Finland, 2015.

Information about the authors:

Ioannis Adamopoulos – a graduate from the Computer Engineering and Informatics Department, Patras, Greece. His interests lie in the Internet of Things and its applications in monitoring and security (adamopoi@ceid.upatras.gr,).

Aristidis Ilias – PhD student at the Computer Engineering and Informatics Department, Patras, Greece and, also, senior information systems manager at the same department. His interests lie in computer security and cryptography as well as security auditing of information systems (aristeid@ceid.upatras.gr).

Christos Makris – Associate Professor at the Computer Engineering and Informatics Department, Patras, Greece. His interests lie in Structures, Information Retrieval, Data Mining, String Management and Processing Algorithms, Computational Geometry, Internet Technologies (makri@ceid.upatras.gr).

Yannis Stamatiou – Professor at the Business Administration Department, Patras, Greece. His interests lie in computer systems security, cryptography and cryptanalysis, intrusion detection algorithms and privacy protection (stamatiu@ceid.upatras.gr).

Manuscript received on 15 April 2023