Philéas: Anomaly Detection for IoT Monitoring

Alberto Franzin[1][0000−0002−4066−0375]*, Raphaël Gyory[1]*, Jean-Charles Nadé[2]*, Guillaume Aubert[2], Georges Klenkle[2], and Hugues Bersini[1]

[1] IRIDIA-CoDE, Université Libre de Bruxelles, Brussels, Belgium
{afranzin,raphael.gyory,bersini}@ulb.ac.be
[2] Degetel Belgium, Brussels, Belgium
jcnade@gmail.com,gaubert@degetel.com,gklenkle@eugeka.com

Abstract. A growing number of private companies and public administrations is adopting Internet of Things (IoT) technologies to monitor resources, spaces, activities and events. Ensuring the required levels of quality of service and security is a key aspect in managing a service based on IoT. We introduce Philéas, a joint project between Degetel Belgium and the IRIDIA laboratory of the Université Libre de Bruxelles to develop a framework to analyze the activity of IoT systems and to identify possible issues via anomaly detection. In this paper we describe our framework, and we present as a demonstration two real cases that have been tackled using this framework.

Keywords: Anomaly Detection · Industrial applications · Internet of Things · Machine Learning · Quality of Service · Security.

1 Introduction

Internet of Things (IoT) devices are increasingly deployed to address a variety of real world tasks. The umbrella term Internet of Things encompasses a variety of technologies that collect, elaborate and transfer data over a network to other devices and servers in an automated and pervasive fashion [2]. They find application in a variety of domains from smart homes [42, 45] to smart cities [35, 50], from agriculture [18, 47] to manufacturing [23, 34], from healthcare [3, 29, 49] to transportation and mobility [17, 43], and many more. Due to the potential impact on society, public administrations also take a great interest in these technologies, from local to international scale [22, 37]. Though estimates on the actual number of devices vary wildly, there is strong consensus on the fact that the exponential growth will only continue in the next years, reaching the tens of billions of devices in the very near future [12, 36].

The growth of IoT technologies is however not without concerns. The complexity and scale of their applications, their ubiquity and interconnection, the heterogeneity and limited capabilities of technologies involved, the collection and treatment of personal and/or sensitive data are all factors that pose serious

*A.F., R.G. and J-C.N. contributed equally to this work.

challenges regarding energy management, security and privacy [15, 19]. IoT technologies offer several vulnerabilities for malicious actors to exploit or, simply, for faults and issues to happen [7]. The timely detection of such issues plays a key role in the management of an IoT network and application.

Big Data technologies and Artificial Intelligence (AI), in particular Machine Learning (ML), techniques are instrumental in assisting human operators in the monitoring, analysis and resolution of such issues [21, 26, 28, 31, 32, 40, 46, 52]. Devices collect a huge amount of data that is sent to other devices and central servers, where it is stored to be processed. While the data collected can be analyzed for the specific application, the metadata consisting in the message headers is useful to understand the state of the network and application. In large amounts of data, patterns of behaviour are likely to emerge, and deviations from them can be an indication of potential problems.

In this work we introduce Philéas, a framework conceived to assist managers and operators of IoT networks and applications in the analysis of IoT data. In particular, while in this project we have analyzed several different cases and applications, our main focus is the detection of anomalies in the metadata received from the devices. Philéas focuses on the analysis of anomalies from a centralized, application perspective, as independent as possible from the technical details of the devices and the network protocol, and not relying on external information about the network status. Philéas is a joint project between Degetel,[1] a consulting and services group specialized in digital transformation in France and Belgium with 15 years of experience in the IoT domain, and IRIDIA, the AI laboratory of the Université Libre de Bruxelles (ULB). Philéas answers specific market demands from the clients of Degetel regarding the management and securitization of IoT infrastructures. The project is funded by Innoviris, the institute of the Brussels Capital Region for technological innovation, with the goal of transferring academic knowledge into the industrial domain, and aims at exploring advanced AI solutions that can accompany more traditional approaches.

In the following Section we review the context of this project, the challenges in IoT systems that we address within this project and some existing relevant approaches. In Section 3 we describe the Philéas framework, including its infrastructure and the algorithms we implemented. In Section 4 we present two real world cases tackled in this project, before concluding in Section 5.

2 Background and related work

2.1 Internet of Things

Internet of Things (IoT) is a set of technologies based on uniquely identifiable devices capable of communicating with each other over a network without human interaction [2, 7]. Though the definition is rather broad, with IoT we usually refer to low power embedded devices with limited computational capabilities devoted to one task, or a specific set of tasks. A common characteristic of IoT devices

[1] https://www.degetel.com

is their pervasivity, that is, the possibility of deploying them in countless places and applications. They enable the collection of huge amounts of data, which is usually collected and analyzed. IoT devices include sensors and smart meters that measure one single value or event (e.g. temperature, humidity, the opening of a door, the failure of a mechanical component in an industrial machine) and transmit it to a central server. The use of IoT in the industry is at the base of the so-called fourth industrial revolution [23]. But in IoT we can also include vehicles capable of communicating with other vehicles and the environment (e.g. road infrastructure) [20]. IoT is also progressively entering private homes, with home automation and intelligent appliances [45].

From an economic perspective, IoT technologies create a new market, whose actors are device manufacturers, network and service providers, and application developers. Public administrations also play a role in this: for example, in Brussels public entities provide support for a smart city initiative[2] and for companies and start-ups to bring AI and IoT solutions to the market. There is also great public interest in the next generation of IoT networks, based on 5G.

Alongside with the many opportunities, the deployment of IoT solutions presents several technical challenges, from the non-interoperability of solutions, to device obsolescence, to the computational challenges of big data analysis [4]. But also the pervasivity of the devices, the possible threats to privacy and security and their implications, even at international level, are key concerns for developers, institutions and regulators [14, 16, 30].

Technical specifications of IoT devices and their interconnection encompass the full stack from the design of the electronic components of a device to the platform and application. Issues can happen at various levels of the IoT system – physical, network, application. Among the several problems that affect IoT systems, here we review the ones that concern the scope of Philéas, and we compile the following list from the management and application perspective, that is, from the central administration of the system.

2.2 Issues and challenges in IoT system administration

Device-related issues A very common situation, especially when using technologies that are inexpensive or with low capabilities, is the malfunctioning of a device, which could fail in some of its parts (e.g. measuring a wrong value, being unable to send or receive messages) or stop working altogether.

Network failure Similarly, a system can fail at the network level, e.g. because of a malfunctioning gateway. Packets can also be lost simply because of a poor network status, caused for example by bad weather conditions. In this case we typically observe a deviation from the usual patterns for a group of devices belonging to the same network, or connected to each other.

[2] https://smartcity.brussels/

Malicious action IoT systems can be the target of criminals with the goal of stealing information, or simply disrupt a service to cause financial damage. Several kinds of attacks are possible on an IoT system, for example, Distributed Denial of Service (DDoS) attacks can compromise a gateway, while the lack of end-to-end message integrity check could be exploited to alter the payload [39].

However, from the perspective of this project, the effect of malicious action results in issues that affect the system at the device or network level, or both. In fact, for both network and device failure a mere log analysis is usually insufficient to distinguish the causes of the failure, whether accidental or caused by malicious actors, and additional knowledge is required to establish the causes of the issue.

System heterogeneity The huge variety in the technologies available makes it very difficult to provide generalized solutions, even for the same kind of task. As an example, and the case that concerns Philéas the most, there are several communication protocols that can be implemented to transmit messages between devices in a network, many of which are proprietary.

2.3 Anomaly detection for IoT system management

The complexity of IoT systems is a perfect application for AI technologies and, in particular, data mining and ML techniques that can be used to process the vast amount of data and metadata collected [26, 28, 46, 52]. A complete review is beyond the scope of this project and of this work; here we limit our discussion to an overview of the techniques that have been applied to monitor the state of IoT systems from the data collected, notably anomaly detection.

An *anomaly* (or *outlier*) is an observation, or a group of observations, that exhibits two characteristics: it differs significantly from the majority of other observations, and it appears rarely in the dataset [5, 25, 27]. Anomalies can take many different forms: the simplest way to define what an anomaly is is therefore to define what *normal* observations (*inliers*) are, and to mark as anomalies all the observations that cannot be considered inliers. The nature of the deviation depends on the particular context and application. We can search for observations that deviate from the regular behaviour in the entire dataset; in this case we are considering *global* anomalies. But anomalies can also occur with respect to a subset of the data, and in this case we identify them as *local* anomalies.

Clustering and neighbourhood-based methods Clustering often is the first step taken to make sense of the data collected. By associating related observations, we can identify groups of devices that exhibit similar behaviour, for some suitable definition of similarity that takes into account relevant features. There are however many possible similarity criteria, and many clustering techniques available, each one possibly entailing different outcomes. Popular techniques include centroid-based algorithms, such as the k-means algorithms, where some observations are chosen as representatives (*centroids*) of the clusters they belong, and the remaining observations are associated to the closest centroid. Another approach is based on *density*, where a cluster is composed by points that have a

minimum amount of neighbouring points under a certain distance; in algorithms from this class, such as DBSCAN, sparse points can be considered outliers. For thorough reviews of clustering algorithms, we refer to [1, 48].

In the case of IoT log analysis, to cluster similar observations we usually need to define a distance function based on a subset of the packet fields. For example, we can group observations by the behaviour they describe, e.g. the number of packets sent by each device in a certain interval of time. But we can also analyze the aggregate of the packets sent by one device, or the devices of a specific client or a certain geographic area.

A notion of proximity between observations is also at the base of the Local Outlier Factor (LOF) algorithm, an anomaly detection technique based on the notion of *local density*, that is, how close each point is to its k neighbours [8]. In a nutshell, LOF classifies as anomalies data points whose local density differs from the local density of its neighbours. LOF is a generic technique that can be applied to various tasks for which we can define a distance between observations.

One-class learning Another approach to anomaly detection is to have a model learn only the "normal" behaviour; this corresponds to a classification task with a single target class. Anomalies are then the observations for which the model performs poorly. Techniques in this family include one-class Random Forests [24] and one-class Support Vector Machines [11]. Isolation Forests exploits the low frequency of outliers to isolate them in leaves of decision trees [33].

A family of artificial neural networks called *autoencoders* is another effective approach to anomaly detection [51]. Autoencoders perform two subsequent actions: first they map (encode) the input to a reduced space of neurons, in order to approximate the input; then they try to re-generate (decode) the input from this approximation. During the training phase they effectively learn a noise-free version of the original model, hence, in general, the reconstruction error will be smaller for observations that match the input model relatively well, rather than for observations that deviate significantly from the majority of the other points in the dataset; the first ones can therefore be considered inliers, while the latter will be identified as outliers.

Time series analysis As devices send packets to the central server either periodically or based on events, the data collected can often be modeled as multivariate time series. A multivariate time series is an ordered set of k-dimensional vectors $\mathbf{X} = \{\mathbf{x}_t\}_{t \in T}$ where each vector $\mathbf{x}_t = \{x_t^1, x_t^2, \ldots, x_t^k\}$ contains the values observed at time t. In our case, the values are the values of the different features we receive, or that we are interested to monitor in the given situation. Anomalies in time series can take different forms. We can look for a point or a sequence of points that deviates from the rest of the points in the time series (*point outliers* and *subsequence outliers*), or for a time series that exhibits a different behaviour from other time series in one or more features (*outlier time series*). Anomaly detection in time series is a rich and active field of research, thanks also to the huge importance of this task for the industry [41], and we refer to [6, 10] for detailed reviews of anomaly detection techniques for time series.

Batch vs real time analysis vs prediction Depending on the context and the specific applications, there are two possible ways of looking for anomalies in log data, batch analysis and real time analysis. Batch analysis processes data about past event, and is performed periodically or occasionally to discover anomalies occurred in the past, e.g. in the context of forensic analysis. Real time analysis is instead applied to the data as it arrives, or in small batches of recent data, and is continuously performed to ensure that potential problems are immediately spotted and taken care of. When a machine learning model is trained on the data available, it can be used to predict the future status of the IoT system, for example the reception of a packet, or the failure of a device or a network. Prediction is always applied to one single instance of the desired target.

Domain expertise Domain expertise is crucial to properly understand and evaluate the results of a data analysis, as the data alone is often not sufficient to fully understand a situation. In particular, in our applications the client needs to be involved in the process and has to analyze the outcome.

3 The Philéas framework

3.1 Scope

In Philéas we implement algorithms to detect anomalies in IoT metadata, processing logs of messages from different sources both in batch and in real time. Philéas provides a framework that can be used not only as stand-alone software, but also to develop specific solutions for different clients. Key elements in the design of the framework are the separation between the data and the AI algorithms and at the identification of common features in the various data sources. Therefore, while we can provide algorithms tailored for specific cases, in general we favour general techniques that can be applied in a variety of contexts. A specific application is then instantiated for each client, selecting the infrastructure and the algorithms that best serve the specific needs for the tasks required. The results of the anomaly detection task are meant to accompany domain expert analyses, to obtain meaningful insights about the status of the IoT system.

3.2 Network protocols

We consider two of the most common options for the communication between the devices and the central infrastructure, Sigfox [53] and LoRaWAN [13, 44]. Both are proprietary technologies, developed in France and available mostly in Europe. They operate in the ISM (Industrial, Scientific and Medical) band, at $867 - 869$ MHz in Europe. The protocols have similar architectures: devices transmit packages to gateway nodes, which in turn communicate with the network server, devoted to manage the data for the various applications. The devices are not associated to a specific gateway, but rather initiate a communication by looking for an available gateway, and continue communicating with a responding one.

Sigfox is a protocol designed to be simple and robust to interference. Its packets contain only nine fields in total including the payload, with additional information about the client, and only the device ID, transmission time and RSSI of the network at the transmission as metadata usable for analysis on the receiving end. It uses an asymmetric link for transmission and reception, so it is a good choice in case of network of sensors that transmit infrequent data (e.g. temperature sensors).

LoRaWAN (Long Range Wide Area Network) is the medium access control and network layer protocol defined in the LoRa standard, designed for long range, low power connectivity: a device can function for over eight years before having to replace its battery. A LoRaWAN packet contains several metadata fields additionally to the payload. These fields include information about the gateway and the antenna, and for both uplink and downlink. LoRaWAN uses a symmetric link, so it is a better choice in case of bidirectional communication.

For more technical details about the specifications of these two protocols, we refer to their official documentations. At the moment we do not consider alternative protocols such as 5G, GPRS or NB-IoT, but the Philéas application is designed to be possibly extended to work with different packet formats.

To provide solutions as general and reusable as possible, we base our analyses on the common fields of both the Sigfox and LoRaWAN packet formats. For Sigfox, the relevant fields include, aside from the payload, the device ID, the client ID, the timestamp, and the RSSI (Received Signal Strength Indicator), a measure of the power of the radio signal, and thus on the quality of the network at the moment of the transmission. Analyzing the content of these fields can already provide several insights on the status of the IoT system. In addition to this, LoRaWAN provides several other information about the network, such as the uplink and downlink gateways, and the connection status, that can be used for deeper analyses when needed.

3.3 Algorithms

Simple rule-based and statistical analyses are very effective in several scenarios. For example, if a device did not send a message in the last x hours, or sent $y\%$ more (or less) messages than the other devices in its network, it may considered a problem. The values of x and y are to be determined by the specific application, either as fixed values provided by the client or after a preliminary data analysis. We can also compare the current behaviour of a device with its past behaviour, to observe whether it changed significantly, according to some threshold values.

In IoT networks the chances of losing a packet are comparatively high with respect to other network technologies, without this necessarily being related to actual problems. Hence, point outliers in metadata, especially multivariate (a single packet received or lost) are at high risk of being false positives. We therefore focus on subsequence outliers and outlier time series, as indications of possible persistent problems.

We use the k-means clustering to identify devices based on their transmission behaviour. For the same task in a big data context we also implement a MapRe-

duce version of the k-center clustering algorithm [9]. The distance function may depend on the specific case, protocol and application, and can therefore be defined with the user. We also use the Local Outlier Factor algorithm to identify devices that have an abnormal frequency of messages, or an abnormal amount of messages received.

While statistical and clustering methods cover most of the needs in our practical cases to characterize, respectively, individual and group behaviour of the devices. However, one of the goals of this project was to investigate more advanced machine learning models for analysis of IoT metadata, for advanced analyses of large batches of data, but also to replace manually-crafted rules and to predict future issues at the device level. We implement autoencoder neural networks, whose hyperparameters are to be set for each specific case.

3.4 Infrastructure note: exchanged place with Algorithms

The database used depends on the amount of data to be collected for each client. We have the option of using the Hadoop infrastructure for managing big data, and PostgreSQL and MongoDB as databases otherwise. Streaming data can be collected using Kafka.

The algorithms of Section 3.3 are built on top of the common Python stack of scientific libraries, based on Spark (Pyspark), Pandas and Scikit-learn for data analysis, feature augmentation and machine learning tasks. We use TensorFlow for implementing deep learning solutions, and Spark for big data analysis. Whenever possible, we use the algorithms available in the Python libraries; we however implement custom algorithms for statistical and time series analyses.

The interface for the application is built using Django and node.js. Communication with the backend is handled by REST services.

4 Use cases

Here we present two examples of issues tackled using the Philéas framework, to showcase the set of algorithms we have currently available. As they refer to specific situations of Degetel clients, the data and some of the specific details are covered by non-disclosure agreements, and we will thus omit from the following presentation any detail that may identify situations, clients or any other party involved unless specifically authorized. We can however present the computational problems, and the approaches we implemented to tackle them.

4.1 Quality of Service

The first case is about Shayp[3], a Brussels-based startup that deploys IoT water telemetry sensors to monitor water consumption in indoor locations. The sensors measure the amount of water used and transmit this value to the central server

[3] http://www.shayp.com

using Sigfox packets, with a frequency of one packet every hour. Some packets are lost, either singularly or in bursts: this normally happens due to poor network conditions. Sometimes packets from a certain device may disappear completely, in case of a faulty device or external intervention (e.g. a device is misplaced after being accidentally hit). To avoid too many false positives, we do not consider a single lost packet as an anomaly; in fact, this can situation can happen for several reasons, and it is not considered problematic in itself. However, two or more consecutive expected packets lost are considered as an anomaly to note.

The dataset for the analysis we report includes anonymized logs of 500 devices for one year of activity, each observation corresponding to one Sigfox packet received ($\sim$ 2.6M packets in total). No information available regarding users is available, and the payload is encrypted. Given the simplicity of the transmission protocol, the relevant information in each packet is only the device ID, the RSSI of the network and the timestamp of the message. The expected periodicity allows us to detect the loss of one or more packets by measuring the time elapsed between two consecutive packets received from the same device.

Monitoring the status of the network and of the devices can be done, in large part, using simple time series and statistical analyses, analyzing the time of each message, and the associated RSSI. We implement rules to detect devices with an anomalous behaviour, with respect to both the other devices in the network and the device expected behaviour.

We use this case also to describe our autoencoder approach to track lost packets. The relevant information for each message is: (i) the device ID, (ii) the RSSI value measured, and (iii) the elapsed time since the previous packet from the same device. Starting from these information, for each device we build two sequences, R_n with the n last RSSI values measured for the device (normalized in the $[0, 1]$ interval, relative to the entire dataset), and T_n, the (normalized) elapsed time between each of the last n packets received. For the autoencoder to learn the "normal" behaviour, we include in the training set only data that corresponds to packages that have at most one packet lost among its predecessors.

The input features for the autoencoder are the two sequences R_n and T_n. The autoencoder has a symmetrical architecture with an input and an output layer of $2n$ nodes, a first and last hidden layer of n nodes and a third and fourth hidden layer of $\lceil n/2 \rceil$ nodes. In our experiments we used $n = 5$, for a total of ten input features. More precisely, the network architecture is the following one:

input layer 10 nodes with ReLu activation;
first hidden layer fully connected, 5 nodes, ReLu with ℓ_1 regularization;
second hidden layer fully connected, 3 nodes, ReLu activation;
third hidden layer fully connected, 3 nodes, ReLu activation;
fourth hidden layer fully connected, 5 nodes, ReLu activation;
output layer fully connected, 10 nodes, ReLu activation.

The autoencoder then computes the reconstruction error of its input. Sequences corresponding to packets considered having normal behaviour have a lower reconstruction error than packets belonging to sequences where many previous packets have been lost. We can thus fix a threshold for the reconstruction error

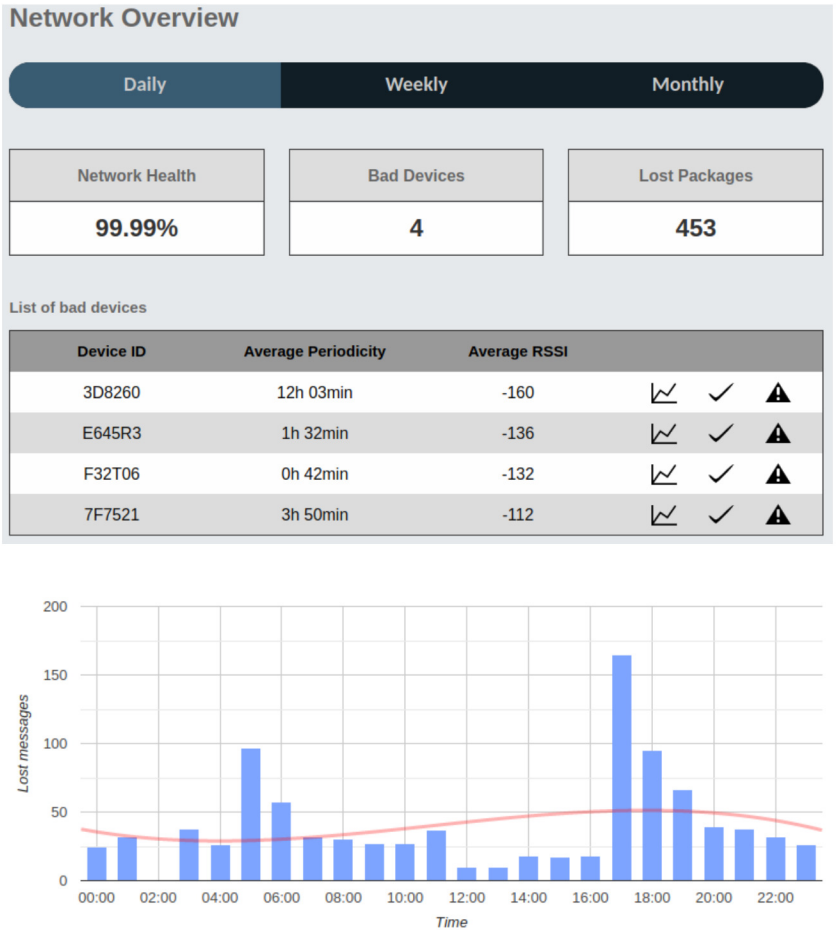

Fig. 1. Daily summary of the network status, with an overview of the main network statistics and a list of the problematic devices (top) and a plot with the hourly amount of lost packets (bottom).

to maximize the correct separation of inliers and outliers. We set the threshold experimentally on the training and validation data. This approach obtains results comparable to statistical and rule-based methods.

From the dashboard of the application the user can monitor the performance of the autoencoder, and choose to modify the error threshold, or to retrain the neural network with more recent data or different time horizons (last week, last month, etc.) should the performance decreases significantly. Philéas allows the users to filter training/validation/test data, include the features they need and set the threshold. In Figure 4.1 we show two examples of information accessible form the Philéas dashboard; the first one is a daily summary of the overall

network status, with the main statistics and a list of the devices that lost too many packets, while the second one is a plot that reports, for a given day, the hourly amount of packets expected but not received.

4.2 Distributed Denial of Service

The second case we present comes from a French company, a major actor in the local IoT market with thousand of clients throughout the entire country. They deploy LoRa sensors for a variety of tasks; with no regular frequency, these devices transmit the information they collect to the central server via LoRaWAN packets. The company requested an analysis of their logs, following an occasional service failure experienced by several of their clients on a certain day d_a, whose devices were unable to connect to the network. The company reported that, overall, nearly 45% of the connections failed, contrarily to a circa 1% of probability of connection failure under normal circumstances. The hypothesis to verify is that this was a case of DDoS [38]. Here we report the analysis on the beginning of the attack.

Due to energy considerations, devices in LoRaWAN networks are not continuously connected to the network, but rather they send a join request to the network when they have to transmit data. If the request is accepted by the server, the device will be assigned a private token to be used during the communication, that will be checked by the gateway. The connection is closed when they stop transmitting, or if they get disconnected from the network. Join requests, both accepted and rejected, are normally received and stored by the central server. Gateways are configured to cap the number of join requests they can handle in a given amount of time; when the limit is exceeded, the gateway will reject all the new incoming join requests, to preserve the central server from the additional load. The recommended practice is therefore to minimize the number of connections, and to avoid repeated retries when a join request fails or in case of a network failure, in order to minimize the load on a network. Unfortunately, IoT protocols only enforce limited secure practices by design, so it is relatively easy for a malicious actor to disrupt a service by making some devices perform an excessive amount of join requests. When this exceeds the network capacity, also non-infected devices are impacted, experiencing more join failures than usual.

We were provided six months of anonymized LoRaWAN logs, for a total of approximately two terabytes of data. The LoRaWAN packets are composed of 12 downlink fields and 60 uplink ones, only one of which is the actual payload; the other ones include many accessory information that is not necessarily useful in many contexts. Moreover, several fields have been anonymized before giving us access to the data, so only partial information was available to us. The relevant fields for this task are the device ID, the client to which the device is associated, the timestamp, and the success status. The first step is to count the packages received by each device, and by the devices of each client. We use Spark to aggregate records in the dataset by device ID, day, and client, to count daily connections. Additionally, the aggregated data is now manageable without big data algorithms or technologies.

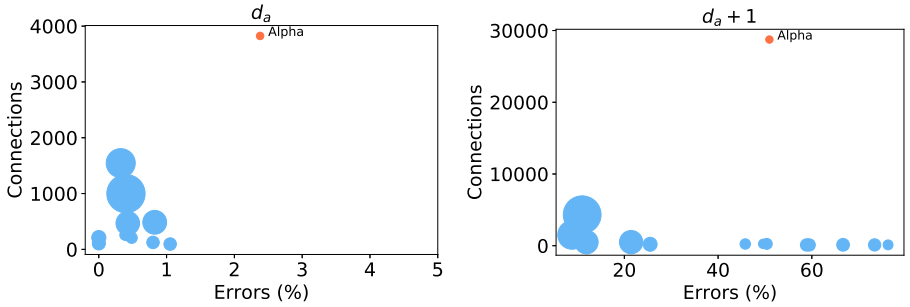

Fig. 2. Number of connections and connection error percentage for the main fifteen clients, on days d_a (the day the DDoS attack started, left plot) and $d_a + 1$ (when the attack continued, right plot). Each circle represent a client, the size of the circle is proportional to the number of devices controlled by that client. The colors indicate the outcome of the Local Outlier Factor analysis: in blue the clients that are considered inliers, in orange the client with anomalous behaviour (conventionally called Alpha).

The analysis by client shows clearly how the attack on one client impacted the other clients of the network. In Figure 4.2 we report the number of connections (on the y axis) and the error percentage (on the x axis) experienced by each client. The client that we call Alpha is the one hit by the attack and on day d_a it starts requesting an unusually high number of connections; on this day, the network load is still under control, and the other clients do not experience any particular issue. However, as the attack continues on day $d_a + 1$ and the network load increases to an excessive level, not only Alpha experiences a higher ratio of rejected joins, but also other clients become affected. In fact, all the clients experience, to various extents, an increase in the number of rejected connections. A consequence is the need for the devices to issue more join request than usual, in order to be able to transmit the information, even if, respecting the protocol recommendations, they do not issue nearly as many new join requests as the compromised client. The same outcome is observed with a Local Outlier Factor on the number of connections, normalized in the $[0, 1]$ interval, which confirms the abnormal behaviour of the devices of client Alpha.

This ex-post analysis serves also as a blueprint for the periodic monitoring of the network status. We can in fact periodically aggregate the data and spot anomalous behaviour by one or more clients; thanks to the reduced size of the aggregated dataset this can be done almost in real time.

5 Conclusions

With the growing interest in IoT technologies and applications, there is also a growing request in the market for data-based solutions to monitor IoT services. We introduced Philéas, a framework to analyze IoT logs to find anomalies in the

metadata, as an indication of potential problems in an IoT network. In Philéas we implemented several machine learning and anomaly detection techniques, and we have applied them to real-world cases of Degetel clients.

The framework can be used to implement custom solutions for clients with particular requirements; to this purpose, and depending on the requests, we are also going to include additional anomaly detection techniques, and network protocols.

Acknowledgements

The work has been made possible by the Innoviris project 2018-SHAPE-25a "PHILEAS: smart monitoring par détection de comportements anormaux appliquée aux objets connectés". M. Wattez contributed to the graphic interface and part of the implementation of the Philéas framework. We thank Shayp for the concession of using their data and use case in this paper. We thank Prof. G. Bontempi, J. De Stefani and G. Buroni for precious discussions and suggestions.

References

1. Ahmad, A., Khan, S.S.: Survey of state-of-the-art mixed data clustering algorithms. IEEE Access **7**, 31883–31902 (2019)
2. Atzori, L., Iera, A., Morabito, G.: The internet of things: A survey. Computer networks **54**(15), 2787–2805 (2010)
3. Baker, S.B., Xiang, W., Atkinson, I.: Internet of things for smart healthcare: Technologies, challenges, and opportunities. IEEE Access **5**, 26521–26544 (2017)
4. Banafa, A.: Three major challenges facing IoT. IEEE Internet of things (2017)
5. Ben-Gal, I.: Outlier detection. In: Data mining and knowledge discovery handbook, pp. 131–146. Springer (2005)
6. Blázquez-García, A., Conde, A., Mori, U., Lozano, J.A.: A review on outlier/anomaly detection in time series data. arXiv preprint arXiv:2002.04236 (2020)
7. Borgia, E.: The internet of things vision: Key features, applications and open issues. Computer Communications **54**, 1–31 (2014)
8. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: Lof: Identifying density-based local outliers. SIGMOD Rec. **29**(2), 93–104 (2000)
9. Ceccarello, M., Pietracaprina, A., Pucci, G.: Solving k-center clustering (with outliers) in mapreduce and streaming, almost as accurately as sequentially. arXiv preprint arXiv:1802.09205 (2018)
10. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM computing surveys (CSUR) **41**(3), 1–58 (2009)
11. Chen, Y., Qian, J., Saligrama, V.: A new one-class SVM for anomaly detection. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 3567–3571 (2013)
12. Cisco, C.V.N.I.: The zettabyte era—trends and analysis, 2015–2020. (2016)
13. Committee, L.A.T., et al.: Lorawan 1.1 specification. LoRa Alliance, Standard (2017)
14. Conti, M., Dehghantanha, A., Franke, K., Watson, S.: Internet of things security and forensics: Challenges and opportunities (2018)

15. Covington, M.J., Carskadden, R.: Threat implications of the internet of things. In: 2013 5th International Conference on Cyber Conflict (CYCON 2013). pp. 1–12. IEEE (2013)
16. for Cybersecurity, E.U.A.: Good practices for security of IoT. Tech. rep., European Union (02 2019)
17. Din, S., Paul, A., Hong, W.H., Seo, H.: Constrained application for mobility management using embedded devices in the internet of things based urban planning in smart cities. Sustainable Cities and Society **44**, 144 – 151 (2019)
18. Elijah, O., Rahman, T.A., Orikumhi, I., Leow, C.Y., Hindia, M.N.: An overview of internet of things (IoT) and data analytics in agriculture: Benefits and challenges. IEEE Internet of Things Journal **5**(5), 3758–3773 (2018)
19. Fu, K., Kohno, T., Lopresti, D., Mynatt, E., Nahrstedt, K., Patel, S., Richardson, D., Zorn, B.: Safety, security, and privacy threats posed by accelerating trends in the internet of things. arXiv preprint arXiv:2008.00017 (2020)
20. Gerla, M., Lee, E.K., Pau, G., Lee, U.: Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds. In: 2014 IEEE world forum on internet of things (WF-IoT). pp. 241–246. IEEE (2014)
21. Ghosh, A., Chakraborty, D., Law, A.: Artificial intelligence in internet of things. CAAI Transactions on Intelligence Technology **3**(4), 208–218 (2018)
22. Gil-Garcia, J.R., Pardo, T.A., Gasco-Hernandez, M.: Internet of Things and the Public Sector, pp. 3–24. Springer International Publishing (2020)
23. Gilchrist, A.: Industry 4.0: the industrial internet of things. Springer (2016)
24. Goix, N., Drougard, N., Brault, R., Chiapino, M.: One class splitting criteria for random forests. In: Zhang, M.L., Noh, Y.K. (eds.) Proceedings of the Ninth Asian Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 77, pp. 343–358. PMLR (2017)
25. Goldstein, M., Uchida, S.: A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PloS one **11**(4), e0152173 (2016)
26. González García, C., Núñez Valdéz, E.R., García Díaz, V., Pelayo García-Bustelo, B.C., Cueva Lovelle, J.M.: A review of artificial intelligence in the internet of things. International Journal of Interactive Multimedia and Artificial Intelligence (2019)
27. Hodge, V., Austin, J.: A survey of outlier detection methodologies. Artificial intelligence review **22**(2), 85–126 (2004)
28. Hussain, F., Hussain, R., Hassan, S.A., Hossain, E.: Machine learning in iot security: current solutions and future challenges. IEEE Communications Surveys & Tutorials (2020)
29. Islam, S.R., Kwak, D., Kabir, M.H., Hossain, M., Kwak, K.S.: The internet of things for health care: a comprehensive survey. IEEE access **3**, 678–708 (2015)
30. Kaska, K., Beckvard, H., Minárik, T.: Huawei, 5G and China as a security threat. NATO Cooperative Cyber Defence Center for Excellence (CCDCOE) **28** (2019)
31. Kotenko, I.V., Saenko, I., Branitskiy, A.: Applying big data processing and machine learning methods for mobile internet of things security monitoring. J. Internet Serv. Inf. Secur. **8**(3), 54–63 (2018)
32. Lee, J., Stanley, M., Spanias, A., Tepedelenlioglu, C.: Integrating machine learning in embedded sensor systems for internet-of-things applications. In: 2016 IEEE international symposium on signal processing and information technology (ISSPIT). pp. 290–294. IEEE (2016)
33. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation-based anomaly detection. ACM Trans. Knowl. Discov. Data **6**(1) (2012)

34. Manavalan, E., Jayakrishna, K.: A review of internet of things (IoT) embedded sustainable supply chain for industry 4.0 requirements. Computers & Industrial Engineering **127**, 925–953 (2019)
35. Mehmood, Y., Ahmad, F., Yaqoob, I., Adnane, A., Imran, M., Guizani, S.: Internet-of-things-based smart cities: Recent advances and challenges. IEEE Communications Magazine **55**(9), 16–24 (2017)
36. Munirathinam, S.: Industry 4.0: Industrial internet of things (IIOT). In: Raj, P., Evangeline, P. (eds.) The Digital Twin Paradigm for Smarter Systems and Environments: The Industry Use Cases, Advances in Computers, vol. 117, pp. 129 – 164. Elsevier (2020)
37. Ponti, M., Micheli, M., Scholten, H., Craglia, M.: Internet of things: Implications for governance (2019)
38. Salim, M.M., Rathore, S., Park, J.H.: Distributed denial of service attacks and its defenses in IoT: a survey. The Journal of Supercomputing pp. 1–44 (2019)
39. Sengupta, J., Ruj, S., Bit, S.D.: A comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT. Journal of Network and Computer Applications **149** (2020)
40. Sezer, O.B., Dogdu, E., Ozbayoglu, A.M.: Context-aware computing, learning, and big data in internet of things: a survey. IEEE Internet of Things Journal **5**(1), 1–27 (2017)
41. Shipmon, D.T., Gurevitch, J.M., Piselli, P.M., Edwards, S.T.: Time series anomaly detection; detection of anomalous drops with limited features and sparse examples in noisy highly periodic data. arXiv preprint arXiv:1708.03665 (2017)
42. Soliman, M., Abiodun, T., Hamouda, T., Zhou, J., Lung, C.H.: Smart home: Integrating internet of things with web services and cloud computing. In: 2013 IEEE 5th international conference on cloud computing technology and science. vol. 2, pp. 317–320. IEEE (2013)
43. Solmaz, G., Wu, F., Cirillo, F., Kovacs, E., Santana, J.R., Sanchez, L., Sotres, P., Munoz, L.: Toward understanding crowd mobility in smart cities through the internet of things. IEEE Communications Magazine **57**(4), 40–46 (2019)
44. Sornin, N., Luis, M., Eirich, T., Kramp, T., Hersent, O.: Lorawan specification. LoRa alliance (2015)
45. Stojkoska, B.L.R., Trivodaliev, K.V.: A review of internet of things for smart home: Challenges and solutions. Journal of Cleaner Production **140**, 1454–1464 (2017)
46. Sun, Y., Song, H., Jara, A.J., Bie, R.: Internet of things and big data analytics for smart and connected communities. IEEE access **4**, 766–773 (2016)
47. Tzounis, A., Katsoulas, N., Bartzanas, T., Kittas, C.: Internet of things in agriculture, recent advances and future challenges. Bios. Eng. **164**, 31–48 (2017)
48. Xu, R., Wunsch, D.: Clustering, vol. 10. John Wiley & Sons (2008)
49. Yuehong, Y., Zeng, Y., Chen, X., Fan, Y.: The internet of things in healthcare: An overview. Journal of Industrial Information Integration **1**, 3–13 (2016)
50. Zanella, A., Bui, N., Castellani, A., Vangelista, L., Zorzi, M.: Internet of things for smart cities. IEEE Internet of Things journal **1**(1), 22–32 (2014)
51. Zhou, C., Paffenroth, R.C.: Anomaly detection with robust deep autoencoders. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 665–674. ACM, New York, NY, USA (2017)
52. Zolanvari, M., Teixeira, M.A., Gupta, L., Khan, K.M., Jain, R.: Machine learning-based network vulnerability analysis of industrial internet of things. IEEE Internet of Things Journal **6**(4), 6822–6834 (2019)
53. Zuniga, J.C., Ponsard, B.: Sigfox system description. LPWAN@ IETF97, Nov. 14th **25** (2016)