Applied graph theory to security: A qualitative placement of security solutions within IoT networks

Tanguy Godquin, Morgan Barbier, Chrystel Gaber, Jean-Luc Grimault, Jean-Marie Le Bars

Applied graph theory to security: A qualitative placement of security solutions within IoT networks

Tanguy Godquin[a,b,*], Morgan Barbier[a], Chrystel Gaber[b], Jean-Luc Grimault[b], Jean-Marie Le Bars[a]

[a]*Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen*
[b]*Orange Labs, France*

Abstract

The rise of edge computing enables local network management. Services are no longer clustered inside the cloud but rather spread all over the whole network. In this paper, we propose a method for deploying security services within an IoT network according to devices capabilities. Our method models an IoT network as a weighted graph using device capabilities. Using the latter, we propose to identify the most suitable location for a security service using dominating sets and the graph weights. We present results obtained by the proposed method on an example of a smart city based on real data using network security functions such as an IPsec service. Results indicate an overall increase in the network security with minimal impact on the information flow while retaining reduced deployment costs.

Keywords: IoT, edge computing security, security as a service, centrality, dominating set.

1. Introduction

Since 2009, the estimated birth of the IoT by Cisco ISBG, the number of connected objects has been growing steadily, to reach in 2020, 25 (Gartner [1]), 50 (Cisco [2]) and even 200 billion devices according to Intel [3]. The communication aspect of connected objects raises important security concerns. Often constrained in terms of hardware, software, and energy, a large majority of devices do not offer sufficient security to their communications. HP reports in 2014 that 70% of devices did not perform communication encryption [4]. The

*Corresponding author
Email addresses: tanguy.godquin@ensicaen.fr (Tanguy Godquin), morgan.barbier@ensicaen.fr (Morgan Barbier), chrystel.gaber@orange.com (Chrystel Gaber), jeanluc.grimault@orange.com (Jean-Luc Grimault), jean-marie.lebars@unicaen.fr (Jean-Marie Le Bars)

Open Web Application Security Project (OWASP) foundation has compiled a top-10 list of mistakes to avoid during the conception of IoT applications, from password quality to the lack of hardware security [5]. Despite an increased use of security elements (mainly crypto-processors) and the ability of IoT devices to perform efficient cryptography [6], a large number of devices are already deployed without these features. Faux et al. [7] report that 84% of manufacturers do not evaluate the security of their products, which is even more alarming considering that 90% of IT security consultants are not trained to secure IoTs. These statistics confirm what Schneier states that "the market still largely rewards sacrificing security in favour of price and time-to-market" [8]. The lack of security surrounding IoT devices is therefore the result of a design choice which leads to the deployment of highly vulnerable devices.

Regarding the Cloud, some concerns were expressed about the decentralization of cloud functionalities to the edge of the network. Edge computing paradigms relate to several computational paradigms such as fog computing or Multi-access Edge Computing (MEC). The fog computing is defined as follows: "a highly virtualized platform that provides compute, storage, and networking services between end devices and traditional Cloud Computing Data Centers, typically, but not exclusively located at the edge of network"[9]. The Multi-access Edge Computing (MEC) refers to a paradigm initially focused on cellular networks that aims to "extend cloud computing capabilities to the edge of cellular networks"[10]. The telco aspect of MEC is particularly visible in the location of MEC nodes (or platforms). Unlike the fog computing, where the fog nodes are located at "any strategic location between end user device and cloud", the MEC platforms may be deployed at various locations within the Radio Access Networks (RANs) such as the Base Stations (which provide mobile services within a particular cell)[10]. The location of these platforms is critical when it comes to edge computing. As part of the MEC, telcos need to deploy services on specific RAN sites. Whereas in the context of fog computing, it will be essential to define what a strategic location is.

The migration from cloud to edge raises major concerns regarding the identification of suitable location for edge nodes and the allocation of services within them. The deployment of security services on every device is not realistic, either due to the deployment costs, the lack of capacities or restricted access to the device. Therefore, we ensure that every device from our network can directly communicate with a security node, a device that hosts a Network Security Function (NSF) such as for example IPsec services [11]. The aim of our work is to propose a method to deploy security services within an IoT network according to the capabilities of the device, including, but not limited to, processor, memory, storage, virtualization capacity and service-specific hardware requirements. The placement problem relates to the selection of nodes to host the security services. To limit the deployment cost of the services a minimal number of nodes have to be selected while providing the most suitable locations for the services. It forms an optimization problem with these two previous criteria. Furthermore, as stated above, our requirements dictate that each node on the network must

be directly linked to at least one secure node. We formalize our problem through an undirected graph where V, the set of vertices, corresponds to the devices and E, the set of edges, refers to devices connections. Afterwards, each vertex is weighted to reflect the capabilities of the devices. The higher the weight, the more it has to be favoured, as it depicts a more compliant device, whereas nodes with a low weight will tend to be avoided. The requirement for direct access from each vertex of the graph to a security node implies that selected vertices forms a dominating set, *i.e.* a set where each vertex of the graph is either in the dominating set or directly linked to a node in this set. In addition, any dominating set, to solve our optimization problem, has to be of minimal size and favour heavier nodes. Our main contribution provides a greedy algorithm that identifies, from an undirected weighted graph, a dominating set satisfying our optimization criteria.

Several studies cover the service placement problem in IoT, edge or fog networks. Most of these studies focus on the placement of general services with numerous objectives. This may involve network concerns such as optimizing bandwidth and response times [12, 13], the minimization of deployment costs [14], or the harmonization of performances [15, 16]. Besides representing IoT networks in graph form, we have not found any approaches to the problem of service placement that rely on the use of graph theory algorithms and more specifically on graph centrality. While Banerjee et al. use graph centrality in IoT networks [17], the authors do not perform a service placement and tackle a completely different problem known as vertex cover. Most related to our contribution is the work of Doriguzzi-Corin et al. [18] which tackles the placement of chained security services using Integer Linear Programming (ILP) and proposes a heuristic for its resolution. This heuristic relies on a graph weighted according to network capacities as well as the computation of the shortest paths between its nodes with no use of graph centrality. The work from Doriguzzi-Corin et al. [18] addresses the placement problem along a given path with fixed points of origin and destination. The objective is to secure these two points with the deployed service. In our case, we wish to secure the overall network, thus finding ourselves with multiple origin and destination points, which represents a different placement problem. Nevertheless, the method of the authors shows a very interesting approach, especially regarding the weighting used and the ILP formulation of their problem. We published in [19] a partial solution of our placement IoT problem where the second criteria – we want the most suitable nodes – is partly considered. Indeed, we supposed that all devices had the same capabilities. Hence only the topology of the IoT network was used to define these nodes. In contrast, we consider in this paper a more realistic model integrating the capability of the devices. On the side of graph theory, we point how carefully the weighting must be performed, revealing a more challenging graph problem than we had before. A wrong choice can dramatically affect the solution. In this new study, we demonstrate that our methodology enables the deployment of security solutions on nodes suitable for hosting, thus drastically reducing deployment costs and more accurately reflecting the IoT environment while maintaining similar performance to previous work. While our

security solution provides interesting benefits, other solutions can be considered, particularly regarding trust management such as [20] and [21].

This paper is organized as follows. Section 2 presents important contextual information regarding security constraints and attack threats on the network. A first approach is discussed, involving the application of edge computing concepts to IoT security. Section 3 provides an abstract methodology for qualitative positioning of edge computing security services on IoT networks. Section 4 outlines the proposed methodology behind a given security service and observes the benefits of this approach. Finally, we conclude this study in section 5 with a summary of our contributions supplemented by perspectives for further development of the model.

2. Security model

Despite an increased number of devices with on-board security features, the security is not always ensured. Features can be poorly implemented with outdated software or cryptographic algorithms. With a lifespan of roughly a decade once deployed, if not maintained correctly, a device will be subject to numerous vulnerabilities. A solution is to substitute all concerned objects by devices with suited capacities, however, it appears neither cost-effective nor realistic. A part of this problem have been addressed by Cloud Computing, in particular by centralizing services in powerful datacenters and servers. However, in the past few years, the trend has been to decentralize functions in the Cloud, a shift that is consistent with edge computing.

In this new paradigm, the gap between processing activities and information producers is reduced. Among the most well-known examples of these mechanisms are the provision of video resources by streaming platforms on servers close to users who are most likely to consume such resources [22] or processing tasks closer to information producers such as providing face recognition at the edge of the network [23]. More generally, when it comes to fog computing, this frequently involves deploying Virtual Network Functions (VNF) to fog nodes. Regarding security, such functions (also called Virtual Network Security Functions: vNSF [11]) may be Virtual Deep Packet Inspector (vDPI), Virtual Firewall (vFirewall), Virtual Intrusion Detection System (vIDS) or even Virtual Encryption Proxy (vProxy) [24].

2.1. Threat model

It is acknowledged that no absolute security exists and the security of computer systems often consists in protecting a system against a specific threat. To propose a securing method, it is, therefore, necessary to define a model that reflects the capabilities and objectives of an attacker to protect a device effectively against that threat.

Stellios et al. [25] discuss an attacker model suitable for the IoT composed of three categories: device accessibility, attacker capabilities and motivations. We consider an attacker based on the *outsider* model defined by Stellios et al., depicted on Figure 1.

The attacker has no physical nor proximal access to the object (in range of targeted device wireless protocols). As it is explained into the telecommunication standard ITU-T Y.4806, we can analyzed the security issues by their impact vector considering which layers is reachable by the attacker [26]. In our working assumptions, the attacker can exclusively perform remote attacks. He can only access to his target from an intermediate device as shown by the arrows in Figure 1. An attack may pass through several nodes as it is the case for Attack Path 1 and Attack path 2. The attack can be prevented at any point along the path.

Resources of the attacker in the model are those of a regular person with moderate technical capacities (between neophyte and expert). Motivations of the attacker model depends on the targeted IoT network. We consider an occasional hacker or a cybercriminal whose resources remain moderate as per the definitions of motivations and attacker profiles provided by Mosenia and Jah [27] and Onik et al. [28].

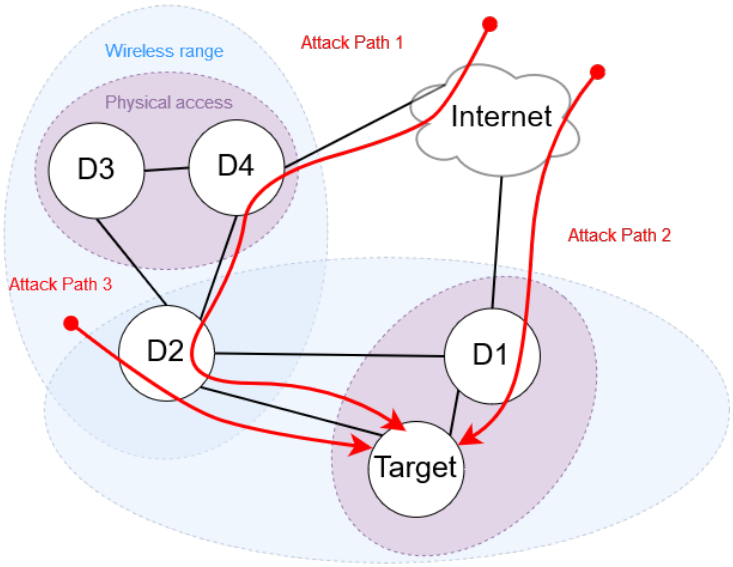


Figure 1: Different examples of attack paths in the IoT context.

Given a target device invisible from the Internet and the capacities of the attacker model, the considered threats use one or multiple intermediate devices to reach the target device, and the source is either on the Internet or in a local network. The presence of the attacker within the wireless range or his ability to physically access the target is not considered due to the higher cost of performing such attacks. Choosing this configuration is a trade-off on the costs of security in accordance with the proportion of attackers evicted.

2.2. Security as a Service Model for Edge Computing

To address this threat model, we propose to deploy NSFs at the edge of the network, thus benefiting from latency reduction, control over data processing and physical proximity of the objects. These security services can be deployed either as dedicated devices with sufficient capacities added to the network or as virtualized services in existing devices or gateways. Examples of such services are IPsec services, firewalls, intrusion detection systems, intrusion prevention systems [11, 24].

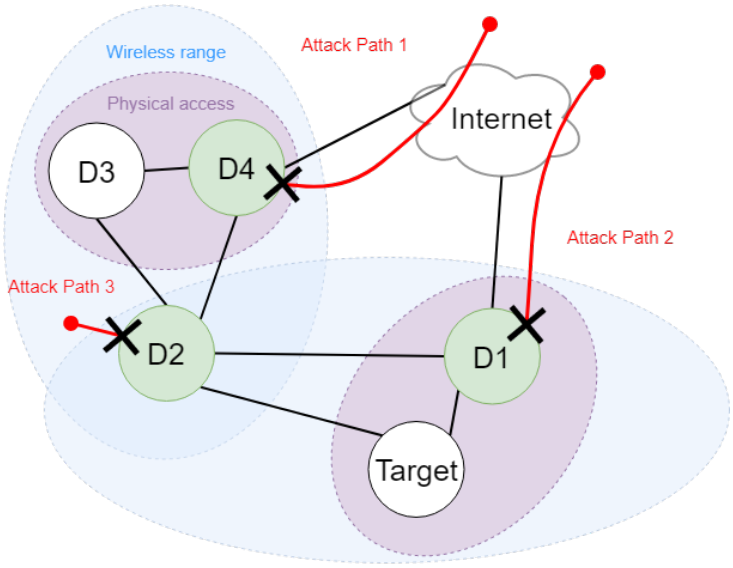


Figure 2: Example of applied security solutions (green plain nodes) over our considered model.

Figure 2 illustrates the deployment of security solutions to prevent threats identified in Figure 1. It is noticeable that all three attack paths (red arrows in Figure 1) have been blocked as a result of the insertion of security solutions (green circles, D1, D2 and D4 in Figure 2) in specific devices on the network. There is no need to deploy security solutions on every device in the network, however, it must be located with regard to identified threats.

2.3. Model relevancy

To achieve the objective described in 2.2, the way security solutions are placed is of paramount concern. Indeed, it is necessary to maximize efficiency and coverage to minimize costs. Section 3 proposes a placement method to address the positioning of security solutions in an IoT network. Studying the most suitable locations for deploying security measures requires modelling an IoT network.

In the first part of our study, we proposed a simplified model for the placement of security functions. We intend to build upon this model to improve the quality of the deployment locations for security solutions. To improve our initial model, we identified two opportunities for improvements. One relating to the inherent capabilities of the devices and the other regarding the communication interfaces among them. The initial model assumed that no single device had sufficient security capabilities by itself and thus required to be improved to accommodate new security solutions. Similarly, we assumed that all the communication interfaces of our initial model had the same characteristics but a slight variation in them may have an impact on the transit of information.

To address the variation between device capabilities and communication interfaces, we propose to consider the inherent characteristics of the devices by introducing the concept of priority within communication interfaces.

3. Placement methodology

This section gives a general method to select which nodes will host the security services. Firstly, we introduce the notion of proximity with secure nodes and give the corresponding graph property. Then, we demonstrate that centrality metrics enable the vertices to be weighted considering the communications between the IoT nodes. Thus, we propose an algorithm that selects the vertices from a weighted graph by favouring the heavier nodes. Finally, we improve our method to consider a more realistic model where the characteristics of IoT nodes become part of the selection of security nodes.

3.1. Formalization of our problem

To provide a better security at the edge of the network, it is important to define the best positioning of this security. Due to the myriad of connected devices, we propose to adopt a more abstract representation of IoT networks by transposing these networks into graphs with objects as vertices and communications as edges. This representation enables a higher degree of abstraction and takes advantage of graph theory concepts to IoT security.

In the following paragraph, d refers to the distance (measured in hop numbers) between an IoT device and its security service. Not being able to always provide security measures on objects ($d = 0$), which is the ideal solution, a compromise must be made. While the usage of the security capabilities of a direct neighbour ($d = 1$) may be acceptable assuming that an attacker does not interfere with this communication, proposing security methods at a higher distance ($d \geq 2$) introduces security issues regarding the entities that are relaying the information. Moreover, a direct access ($d = 1$) follows the fog and MEC paradigms where the devices directly access the closest services [10].

Therefore, we seek to ensure that each object in our network is in direct contact with at least one device that offers security services that we call a security node. Let $G(V, E)$ be an undirected graph where V and E correspond respectively to the vertices and edges of G, the neighbourhood of $x \in V$ is

defined as $N[x]$. The graph G will correspond to a IoT network where V will represent the devices and E will symbolize the connections between devices. Note that we will use the term node when we refer to the IoT while we use the term vertex of the graph to avoid the confusion between these data.

Modelling such conditions on the graph is equivalent to the identification of a dominating set. Indeed, a dominating set is a subset S of vertices of a graph G for which each vertex of G is either in S or adjacent to a vertex of S. To reduce the expense of converting devices into security nodes, one would like to identify a minimum dominating set. Finding a minimum dominating set is proved to be a NP-hard optimization problem where even the estimation of the minimum size remains difficult [29]. Aware of this situation, we do not attempt to identify the minimum dominating set; a minimal dominating set, *i.e.* a local minimum, is sufficient.

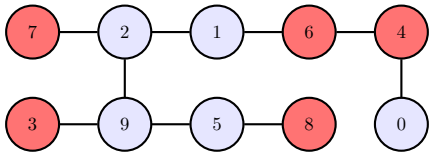


Figure 3: Dominating set (red vertices) located off the data flow.

3.2. Minimal dominating set

We assume to have a vertex-weighting function. Algorithm 2 returns a minimal dominating set depending of these weights. The vertices are visited in ascending order and are kept out of the dominating set until there is no other choice but to place them into the dominating set, which has the effect of selecting heavier vertices. Doing so helps to select the heaviest nodes. Note that this is a general algorithm that can be used in other contexts; furthermore, its effectiveness strongly depends on the weight allocation.

We refer to F as the set of free vertices that have not yet been browsed by the algorithm. Once the vertex is scanned, it is either transferred to the dominating set (DS), to the set of covered vertices (C, vertices adjacent to vertices in DS) or to the set B corresponding to browsed vertices that will later be in C but require an adjacent vertex in DS. An adjacency list labelled $N_F(x)$ tracks the neighbourhood of the vertex x within F at any time. This suggested algorithm requires the following two functions (Algorithm 1) to be specified: $ForcedDominant$ and $Propagation$.

The $ForcedDominant$ function specifies whether a vertex provided as a parameter should be added to a dominating set. To do so, it checks whether the vertex no longer has the possibility of being covered or whether one of its neighbours depends on it for its coverage. In this case, the addition into the DS set is required. The $Propagation$ function is performed when a vertex is added to the dominating set (DS). It covers all the neighbours of this vertex that have already been visited by the algorithm and that were on hold (B set).

Algorithm 1 Functions used on algorithm 2

```
1: function ForcedDominant(node)
2:     if N_F(node) = ∅ then
3:         Return True
4:     for all n ∈ N[node] ∩ B do
5:         if |N_F(n)| = 1 then
6:             Return True
7:     Return False
8: function Propagation(node)
9:     for all n ∈ N[node] ∩ B do
10:        Move n from B to C
```

The proposed Algorithm (2) browses a given set of vertices, if the current vertex analysed is not covered yet but has at least one neighbour to be analysed (meaning it could be covered later by this neighbour), the latter is put on hold. When a vertex no longer has neighbours not analysed to provide coverage, the current vertex is added to the dominating set. Since the list of vertices is sorted by ascending weight given as input to the algorithm ($vList$), the dominating set discovered (DS) favours the heavier vertices. Lighter vertices are first placed in C or B which allows, later on, to integrate heavier vertices in DS.

3.3. IoT topology and centrality metrics

We proposed in [19] a partial solution of our placement problem where all devices had the same capabilities and where only the graph topology was considered. This subsection outlines the method used. Note that in [19] we were not talking about vertex-weight, only centrality value based on the graph topology.

Finding a small dominating set is not our only concern, we also wish to ensure that the positioning of the vertices making the dominating set is consistent with the transit of information in the graph which depends on its topology. Figure 3 illustrates a bad scenario where the vertices of a dominating set (in red) are not positioned along the information path.

Our first approach is to weight vertices according to their importance in the graph. The importance of a vertex can be obtained using centrality metrics. Depending on our security needs, some are more appropriate than others.

We limit our investigations on four of the most popular measures, namely degree centrality, eigenvector centrality, closeness centrality, and betweenness centrality which appear the more relevant for the IoT topology.

Degree centrality. The degree centrality is a measure of the importance of a vertex according to its degree.

Eigenvector centrality. The idea behind eigenvector centrality is that the importance of a vertex depends on the importance of its neighbours [30, 31]. Given a graph G, the value of a vertex $i \in G$ is calculated using the following

Algorithm 2 Central Dominating Set Algorithm.

Input:
$vList$: ordered list of V
Output:
DS: dominating set vertices

```
1: F ← V
2: DS, B, C, N_F ← ∅
3: while vList ≠ ∅ do
4:     a ← vList[0]
5:     if a ∈ F then
6:         if ForcedDominant(a) then
7:             Move a from F to DS
8:             Propagation(a)
9:         else if |N_F(a)| = 1 then
10:            b ← N_F(a)[0]
11:            Move a from F to C
12:            Move b from F to DS
13:            Propagation(b)
14:        else
15:            Move a from F to B
16:    else
17:        vList ← vList − a
18: Return DS
```

equation where n is the number of vertices of G, A is the adjacency matrix of G and λ_{max} is the largest eigenvalue (non-negative due to the Perron-Frobenius theorem):

$$C(i) = \frac{1}{\lambda_{max}} \sum_{j=1}^{n} A_{ij} C(j).$$

Closeness centrality. The closeness value corresponds to the proximity of a vertex with all vertices of the graph [32]. To compute this measure, all the shortest paths between the analysed vertex and the rest of the vertices of the graph are studied. The measurement of a vertex thus corresponds to the average distance between the vertex and the other vertices of the graph. The normalized mathematical representation of this measurement is the following where n is the number of vertices in the graph and g_{ij} the geodesic distance between vertices i and j:

$$C'(i) = \frac{n-1}{\sum_{\substack{j=1 \\ i \neq j}}^{n} g_{ij}}.$$

Closeness centrality is a graph metric adapted when one wishes to favour the vertices from which it is faster to reach all the other vertices of the graphs. It is

used when positioning fire stations in cities [33] and could also be used for the positioning of update servers in IoT networks when required.

Betweenness centrality. This measurement corresponds to the number of times a vertex acts as a bridge on the shortest path between two other vertices in the graph [34]. The value of a vertex i is expressed using the following equation where g_{jk} is the number of shortest paths (or geodesics) connecting the vertices j and k, $j \neq k$, and $g_{jk}(i)$ is the number of these that pass through i:

$$C(i) = \sum_{\substack{i \neq j \\ i \neq k}} \frac{g_{jk}(i)}{g_{jk}}.$$

Assuming the information in the graph flows along the shortest paths, the betweenness measures the contribution of a node to network traffic.

Depending on the centrality measure that is used, the importance of a vertex in a graph may differ. Figure 4 corroborates this statement by displaying the vertex with the highest value for the four defined measures on a given graph.

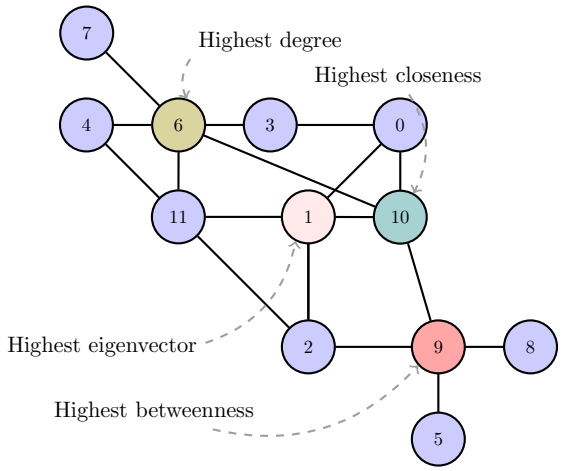


Figure 4: Difference of centrality values on a given graph.

3.4. Real-world constraints consideration

Using centrality metrics exclusively is not enough to limit the deployment costs of security functions as much as possible while simultaneously protecting against the defined attacker model. To achieve this requirement, the devices suitable for the security functions must be identified. We define a priority value for nodes which includes several criteria such as processor capacity, memory, storage, bandwidth, specific hardware. We assume here that the vertices of our graph are weighted so that the most suitable nodes are the heaviest. We

describe in next two subsections different approaches to compute the nodes weights according to their suitability.

The nodes with high priority value will then be prioritized for the selection of the dominating set. For our experiments and the evaluation of the dominating set, we introduce the concept of quality. The quality of a dominating set is the proportion of prioritized objects among this set. Using this methodology, nodes with sufficient hosting capabilities will be able to become security nodes through software updates, which is less expensive, while the others may be given a hardware update or even a complete replacement of the device.

To improve the quality of the selected dominating set, the computation of centrality values may include a priority value according to the characteristics of the devices and their communications. Node priority value will not be used directly to define the vertex weight function. We introduce an edge weighting function to induce the computation of centrality to consider priority node values and network topology.

Indeed, with our approach, edge weights impact the centrality value computation by favouring vertices which correspond to nodes with high priority value. As a result, the vertex weights are altered which implies the order of V in the Algorithm (2) to be changed. In next paragraphs we propose a method to compute the edge weight function from node priority values. Note that the priority value includes different criteria such as processor capacity, memory, storage, bandwidth, specific hardware.

While navigating the graph, if there are several paths available between two nodes, this weighting enables the selection of the path with the lowest weight. This approach consists in favouring communications between high-priority value nodes and thus increasing the quality of the dominating set identified.

Once a good dominating set has been identified, IoT devices corresponding to security nodes can either be updated to deliver desired security features or be replaced if their security capabilities are not sufficient.

Two devices interacting with each other are able to use those security nodes as gateways whenever it is necessary to benefit from the security functions offered by the latter ones. To ensure that communication has been carried out through one (or more if necessary) of these security gateways, the Proof Of Transit [35] could be applied. For this procedure, one or more secrets are shared via Shamir Secret Sharing to nodes where one wishes to attest the passage of the communication. The security nodes would use their secrets (or sub-secrets) to sign communication packets flowing through them. Recipients could then verify that all previously imposed signatures had been applied to communications meaning that the security methods have been enforced.

4. Experiments over a specific use case

4.1. Dataset

Acquiring data related to communications between IoT objects is a difficult task. A large part of IoT datasets only contains values from various sensors.

Researchers working on the concept of Social IoT have made available a few datasets containing information on devices interactions [36]. In our work, we use a graph generated from the OOR (Ownership Object Relationship) adjacency matrix containing information about public and private objects that we have filtered to keep only public static devices. The graph thus obtained is a non-oriented graph composed of 1,458 vertices and 35,657 edges. The vertices correspond to IoT objects while the edges are potential communications between these devices determined according to the available communication protocols of vertices (Bluetooth: 40 meters, WiFi: 400 meters and LoRa: 1,500 meters).

SIoT dataset is accompanied by a short description of the objects with the following information: the device identifier, the owner identifier, the category of the device, as well as information on the brand and model. These data are provided using simple identifiers ranging respectively from 1 to 16, 12 and 24. Details concerning the brand or model is not provided, however, categories are listed and included; Table 1 illustrates the distribution of these categories from the graph.

Table 1: Distribution of device categories in our graph

Category	**Number of devices**	**Proportion**
Point of interest	95	6.5%
Environment and weather	140	9.6%
Indicator	10	0.7%
Street Light	506	34.7%
Parking	677	46.43%
Alarms	30	2%

In our methodology, Section 3, we discussed the need to perform a cleavage between devices according to their priority value. A distinction between two categories has been chosen to represent high-priority and low-priority value devices. In our study, depending on a required security service, an object will either have the ability to host that service or not. Introducing additional categories would allow variations in the hosting capability of a device. However, the difficulties covered in Section 4.7 and illustrated in Figure 7 would be increased exponentially depending on the number of categories.

The Smart Santander project, from which our dataset is derived, presents a smart-city use case for irrigation of parks and gardens where street lights are used as repeaters between the different communication interfaces of the network [37]; thus we have opted to define devices from the "Street Light" category as high-priority value nodes, hence prioritized. Moreover, street lights are no longer only considered as mere lights, many cities like San Diego, Chicago and Los Angeles invest in Smart-Street lights combining a variety of sensors [38, 39, 40]. Now, they can host micro cell towers, provide WiFi coverage and even charge electric vehicles [40]. NEC offers smart street lights with embedded 5G base station and direct access to cloud applications [41]. Deploying security services on such devices would be relevant in a smart-city context. The ubiquity of street

lights and their large number make them ideal targets to provide geographical coverage for different communication services, and therefore have the potential to host varied security functions.

4.2. Priority pre-processing

We fix two priority values L and H over the IoT nodes such that the smaller the value, the higher the priority of the node is. In our dataset the Street Light nodes will have high priority value H while the others have a low priority value L. For any vertex $a \in V$, we note $P(a)$ its priority value. The edge weighting function is directly derived from P: for any $e \in E$, let a and b being its incidents vertices, $W(e) = P(a) * P(b)$.

Table 2 displays the distribution of the weight of these categories over the considered graph.

Table 2: Weight distribution of edges in the graph

Weight	**Number of edges**	**Proportion**
1	961	2.7%
1.2	9,088	25.49%
1.44	25,608	71.82%

Weighting the edges of the graph to this extent affects the computation of the shortest paths used in centralities. The idea is to favour edges between nodes with high priority value while not overly penalizing the others. We show in our experiments that the weighting must be carefully chosen to offer the best results. The values of L were varied from 1 to 5 to identify which value gave the best results, these data are shown in Figure 7. We found that $L = 1.2$ and $H = 1$ are good choices. More details will be provided in Section 4.7.

4.3. Centralities

The effectiveness of an NSF relies on its location. While the NSFs perform operations on the network traffic such as filtering or encryption [11], it is clear that if such services are not located across the information flow, their efficiency will be impacted. Thus, the centrality metric that theoretically seems the most appropriate is the one that considers the flow of information in the graph, namely the betweenness centrality. Another advantage of betweenness centrality, also shared with closeness centrality, is to consider the weight of edges when computing shortest paths, meaning that a priority can be considered without any additional effort.

While computing shortest paths, edges with small weights (high priority value) will be preferred when navigating the graph. On the opposite, edges with a low priority value will be less likely to be selected as the paths will be longer. Using the weights in the centralities computation increases the importance (and therefore the value) of high-priority value nodes.

4.4. Dominating set selection

Once the values for all vertices in the graph have been calculated, our methodology suggests selecting a dominating set by favouring the addition of vertices with the highest priority values. More precisely, the selection operates as follows: the weakest value vertices are first put apart from the dominating set. A vertex is placed in the dominating set when it has no other options.

4.5. Evaluation

The algorithm introduced allows the analysis of the performance offered by applying various metrics. To reflect our requirements regarding the number of security nodes and vertices positioning, we use the following rating methods: the number of vertices in the dominating set, the quality of the dominating set, the protection provided through their positioning as well as a normalized value of protection independent of the graph size.

We defined the quality of a dominating set on Section 3.4 as the percentage of high-priority value nodes within the dominating set. The greater the percentage, the higher the overall quality will be considered. In our experiments, this refers to the proportion of devices from the "Street Light" category that are within the dominating set. Low-priority value nodes will either be upgraded or replaced with higher capacity versions to provide the requested service.

The protection provided by the vertex placement on dominating set is obtained by computing the percentage of vertex pairs from the graph that have at least one shortest path secured as follows:

$$protection = \frac{\sum_{a,b} S(a,b)}{\binom{n}{2}}$$

where:

$S(a,b) = 1$ if there is a secured shortest path between vertices a and b
$S(a,b) = 0$ otherwise.

We define a shortest path to be secured if the vertices (sender and receiver) are adjacent or if each vertex of that pair is either member of the dominating set or adjacent to one. The illustration Figure 5 visually depicts this definition. Red vertices refer to dominating set vertices, while blue vertices are free to belong either to the set or not.

The normalized value of protection corresponds to the protection value normalized by the size of the graph over the size of its dominating set. It is computed using the following equation:

$$normalized\ value = protection * \frac{n}{|DS|}.$$

This process is essential to consider the size of a dominating set when comparing the contributions of different methods of selection. Otherwise, a method returning a dominating set containing all vertices of a graph (100%

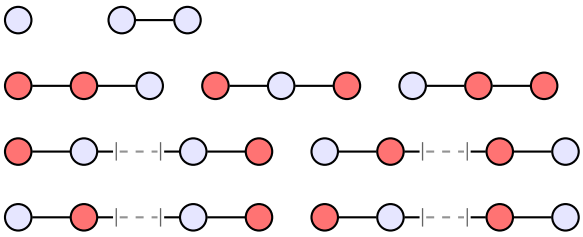


Figure 5: Illustration of secured paths

protection ratio according to our evaluation) could be considered better than any dominating set.

Consider Figure 3, where we described the positioning of vertices not consistent with the information flow. According to our evaluation methodology, the protection provided by the dominating set placement on Figure 3 is 53%. Using the proposed approach for the placement of dominating set using the betweenness on the same graph, we obtain the graph in Figure 6 with a protection index of 100%. The dominating set discovered using our methodology is positioned across the information flow unlike in the one on Figure 3.

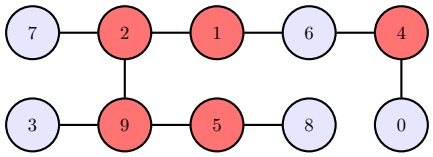


Figure 6: Dominating set (red vertices) identified using our methodology on Figure 3 graph.

4.6. Results

According to the assumption that information flows along an optimal path, using betweenness appears more suitable for the placement of Network Security Functions. Algorithm 2 was used several times with lists of vertices ordered differently, the resulting dominating sets were then examined. We focused our experiments on the four most popular metrics, namely closeness (CC), eigenvector (EC), degree (DC) and betweenness (BC). The proposed algorithm favours selection of vertices located at the end of the list provided, therefore these listings were sorted by increasing order of centrality. The dominating sets obtained are analysed according to our evaluation criteria (size, quality, protection provided by the set and protection brought by one vertex) to be compared with those identified from randomly ordered lists (R) of vertices. Comparison to randomly ordered lists of vertices highlights the importance of vertex order in the selection of dominating sets. The results of this comparison are summarized in Table 3 below while Figure 8 summarizes for each category of devices, their proportion among the dominating sets.

Table 3: Comparison of central dominating set results between ordered and randomly ordered vertex lists.

Graph	Metric	Centralities				Random order	
		BC	*CC*	*DC*	*EC*	*R* seed=82	*R* seed=45
IoT dataset weightless (1,458 vertices)	DS size	107 (7.33%)	107 (7.33%)	155 (10.63%)	176 (12.07%)	191 (13.10%)	212 (14.54%)
	Protection (%)	100	99.92	99.95	99.91	90.87	99.16
	Normalized protection	13.63	13.62	9.402	8.277	6.937	6.820
	Quality	47.7%	48.6%	44.5%	46.0%	51.3%	52.4%
IoT dataset weighted (1,458 vertices) weight ratio: 1.2	DS size	114 (7.819%)	132 (9.053%)	155 (10.63%)	176 (12.07%)	191 (13.10%)	212 (14.54%)
	Protection (%)	99.79	99.79	99.95	99.91	90.87	99.16
	Normalized protection	12.76	11.02	9.402	8.277	6.937	6.820
	Quality	71.9%	81.8%	44.5%	46.0%	51.3%	52.4%

The effect of weight ratio has been analysed using betweenness (BC) and closeness (CC) centralities. The results are plotted in Figure 7 where each curve corresponds to an evaluation criterion according to the ratio between weights of low (L) and high (H) priority nodes: L/H.

4.7. Analysis on IoT dataset

Results from this study indicate that the betweenness and closeness centralities provide smaller dominating sets. It is noticeable that using centrality values in general offers better performance overall. Table 3 presents only two random tests by choice of readability of the data. The dominating sets identified using a collection of random lists of vertices benefit from similar performance to those depicted in the last two columns of Table 3 ranging from 180 to 210 vertices with a proportion of protection from 81% to 99.91%.

Dominating set sizes are about twice as small using betweenness and closeness values as when vertices are sorted in random order. The gain observed is not limited to the number of vertices of the dominating sets but also their positioning. While the majority of the results in Table 3 reveals protection level approaching 100% (only reached with betweenness on weightless graph), it should be remembered that this value depends on the number of vertices in the dominating set. A larger dominating set is more likely to have a high protection

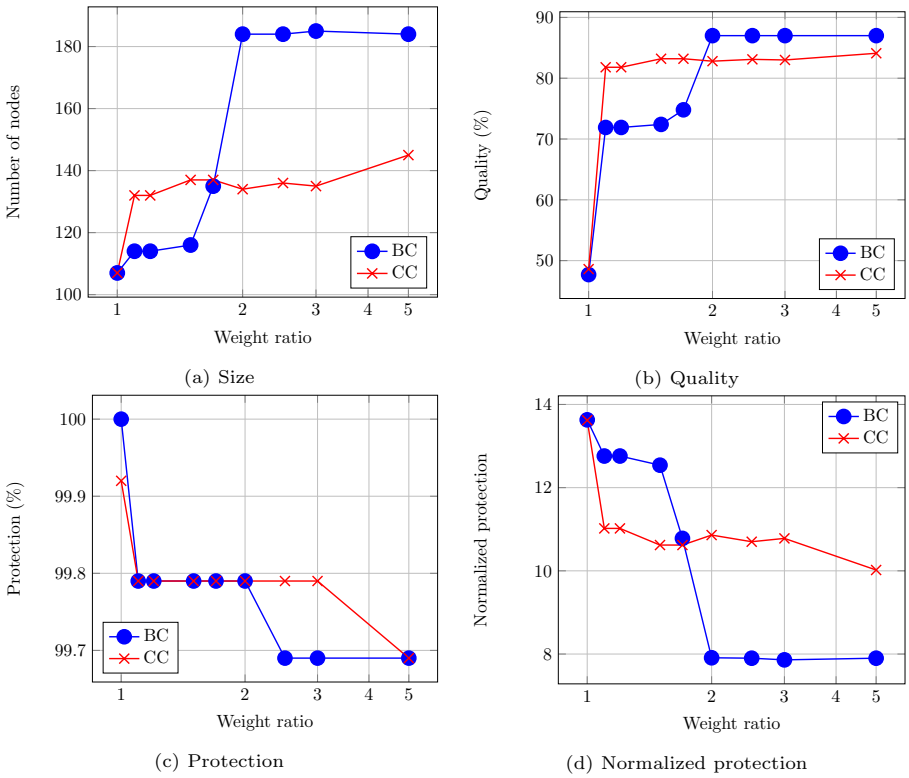


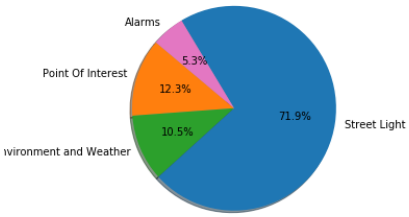
Figure 7: The impact of weight ratio on the evaluation criteria of the dominating set between prioritized and non-prioritized nodes.

value. Normalizing the measure according to the number of vertices of sets is therefore important for an accurate representation of gains.

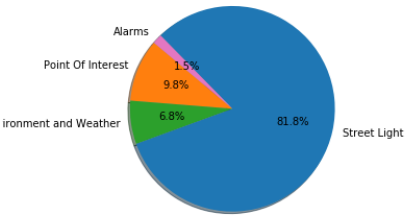
Protection index provided by a vertex from our IoT reference weightless graph using betweenness is almost twice as high as for random lists. The metrics of betweenness and closeness centralities stand out particularly from this analysis by offering a gain of one-third superior to the degree (highest index of all other results).

The dominating sets identified by betweenness and closeness over the weightless dataset, although highly similar from a performance point of view, are not identical. There are 18 vertices that are not shared between them, this similitude lies in the calculation methods of these metrics using both shortest paths within a graph.

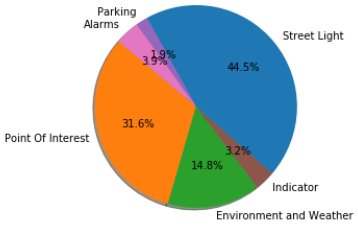
Observations on the non-weighted graph can be reported over the weighted graph. As expected, there is a slight increase in size of the dominating set, due to the additional constraints added on the selection process. With the addition of the weighting, the protection value is reduced. It means that fewer shortest paths in the graph are considered secured as depicted in Figure 5. In our dataset



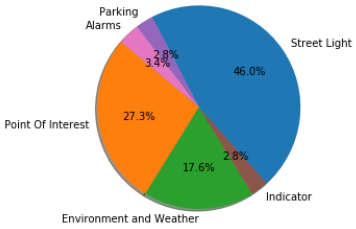
(a) Betweenness (114 nodes)



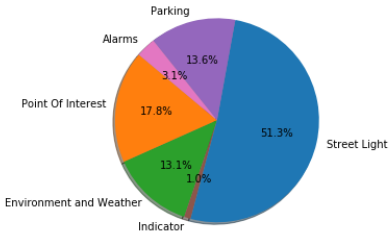
(b) Closeness (132 nodes)



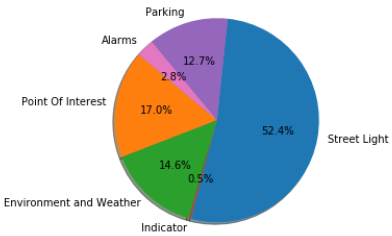
(c) Degree (155 nodes)



(d) Eigenvector (176 nodes)



(e) Random (seed=82) (191 nodes)



(f) Random (seed=45) (212 nodes)

Figure 8: Quality of central dominating sets over our weighted dataset (weight ratio: 1.2).

consisting of 1,458 nodes, 35,657 edges and 1,062,153 shortest paths (potential communications), only 2,275 communications will have additional cost in their communications.

Increasing the quality of the dominating sets identified with weighting is particularly interesting when reducing the cost to deploy security functions. We defined the quality as the proportion of devices that can directly host the desired security function. Software deployment may be sufficient on this

category of devices. Assuming that software deployment is more cost-efficient than upgrading or replacing devices, increasing the quality of the dominating set can significantly reduce the deployment costs of a security function.

It would not be possible to achieve a quality of 100% in our dataset since all 506 street lights nodes do not form a dominating set. Yet, weighting the graph improves the quality of the dominating sets while limiting the number of nodes added and additional communication costs.

Figure 7 shows the impact of the ratio between the low priority value and the high priority value over the number of vertices in the dominating set and the quality of the solution.

Closeness centrality attaches increasing importance to vertices that can spread information the fastest in a graph while betweenness reveals vertices that are located across the information flow. We are more likely to select betweenness to place NSFs across the information flow. Closeness cannot be excluded, however, it could very well correspond to a use case requiring the propagation of information through an IoT network.

We have decided to distinguish the measurement choice from the proposed algorithm and our methodology in order to allow the user to decide which option is best suited to his security needs.

5. Conclusion and future work

We propose a strategy to effectively deploy security solutions within an IoT network while minimizing costs. With the help of graph modelling, we reformulate the security constraints as the identification of dominating set where we favour the heavier vertices. Using centralities to compute the dominating set greatly reduces its size while accepting some adjustments using the graph weighting. We modelled the compliancy (ability to host a requested service) by putting a priority value on the nodes. Our solution has a minimum impact on the information transit while respecting the assumption that it flows optimally through the graph. We demonstrate that classifying nodes into two categories, high and low value, requires a thorough choice of the values used. Increasing the number of categories could offer better results. However the weighting will be exponentially more challenging. Introducing weights in the selection of the dominating set significantly increases its quality while preserving the low impact on communications and limiting the number of nodes added. Adding weighted edges on the graph therefore reduces the cost to deploy security functions by favouring devices that already have sufficient hosting capabilities.

Interesting perspectives could be to improve the proposed model by including other IoT features (more detailed device capabilities, risks or even bandwidth) to achieve a more realistic model, more specifically regarding weight assignment on the graph.

Acknowledgment

The authors would like to thank Paul Dorbec and Nancy Perrot for their help regarding dominating sets and their constructive remarks.

References

[1] Gartner, Gartner Says 4.9 Billion Connected "Things" Will Be in Use in 2015.
URL https://www.gartner.com/newsroom/id/2905717

[2] D. Evans, The internet of things: How the next evolution of the internet is changing everything 1 (2011) 1–11.

[3] A Guide to the Internet of Things Infographic.
URL https://www.intel.com/content/www/us/en/internet-of-things/infographics/guide-to-iot.html

[4] HP, HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack.
URL http://www8.hp.com/us/en/hp-news/press-release.html?id=1744676

[5] OWASP Internet of Things Project - OWASP.
URL https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=IoT_Top_10

[6] G. C. C. F. Pereira, R. C. A. Alves, F. L. da Silva, R. M. Azevedo, B. C. Albertini, C. B. Margi, Performance Evaluation of Cryptographic Algorithms over IoT Platforms and Operating Systems 2017 1–16. doi:10.1155/2017/2046735.
URL https://www.hindawi.com/journals/scn/2017/2046735/

[7] S. Faux, La sécurité à l'ère des objets connectés: Comment s'y prendre ?, in: Workshop "Sécurité Des Objets Connectés".

[8] B. Schneier, Lessons From the Dyn DDoS Attack.
URL https://www.schneier.com/blog/archives/2016/11/lessons_from_th_5.html

[9] F. Bonomi, R. Milito, J. Zhu, S. 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 - MCC '12, ACM Press, p. 13. doi:10.1145/2342509.2342513.
URL http://dl.acm.org/citation.cfm?doid=2342509.2342513

[10] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, T. Taleb, Survey on Multi-Access Edge Computing for Internet of Things Realization 20 (4) 2961–2991. arXiv:1805.06695, doi:10.1109/COMST.2018.2849509.
URL http://arxiv.org/abs/1805.06695

[11] L. R. Battula, Network Security Function Virtualization(NSFV) towards Cloud computing with NFV Over Openflow infrastructure: Challenges and novel approaches, in: 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1622–1628. doi:10.1109/ICACCI.2014.6968453.

[12] F. B. Jemaa, G. Pujolle, M. Pariente, QoS-Aware VNF Placement Optimization in Edge-Central Carrier Cloud Architecture, in: 2016 IEEE Global Communications Conference (GLOBECOM), pp. 1–7. doi:10.1109/GLOCOM.2016.7842188.

[13] Y. Xia, X. Etchevers, L. Letondeur, T. Coupaye, F. Desprez, Combining hardware nodes and software components ordering-based heuristics for optimizing the placement of distributed IoT applications in the fog, in: Proceedings of the 33rd Annual ACM Symposium on Applied Computing - SAC '18, ACM Press, pp. 751–760. doi:10.1145/3167132.3167215.
URL http://dl.acm.org/citation.cfm?doid=3167132.3167215

[14] B. Donassolo, I. Fajjari, A. Legrand, P. Mertikopoulos, Fog Based Framework for IoT Service Provisioning, in: 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), IEEE, pp. 1–6. doi:10.1109/CCNC.2019.8651835.
URL https://ieeexplore.ieee.org/document/8651835/

[15] O. Skarlat, M. Nardelli, S. Schulte, M. Borkowski, P. Leitner, Optimized IoT service placement in the fog 11 (4) 427–443. doi:10.1007/s11761-017-0219-8.
URL http://link.springer.com/10.1007/s11761-017-0219-8

[16] O. Skarlat, M. Nardelli, S. Schulte, S. Dustdar, Towards QoS-Aware Fog Service Placement, in: 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC), IEEE, pp. 89–96. doi:10.1109/ICFEC.2017.12.
URL http://ieeexplore.ieee.org/document/8014364/

[17] P. S. Banerjee, B. Maiti, Optimality criterion for the insertion of multi-interface nodes to improve connectivity in heterogeneous IoT framework, in: 2017 Devices for Integrated Circuit (DevIC), IEEE, pp. 556–560. doi:10.1109/DEVIC.2017.8074012.
URL http://ieeexplore.ieee.org/document/8074012/

[18] R. Doriguzzi-Corin, S. Scott-Hayward, D. Siracusa, M. Savi, E. Salvadori, Dynamic and Application-Aware Provisioning of Chained Virtual Security Network Functions 17 (1) 294–307. arXiv:1901.01704, doi:10.1109/TNSM.2019.2941128.
URL http://arxiv.org/abs/1901.01704

[19] T. Godquin, M. Barbier, C. Gaber, J.-L. Grimault, J.-M. Le Bars, Placement optimization of IoT security solutions for edge computing based on graph theory, in: 38th IEEE International Performance Computing and

Communications Conference (IPCCC 2019).
URL https://hal.archives-ouvertes.fr/hal-02314892

[20] H. Sato, A. Kanai, S. Tanimoto, T. Kobayashi, Establishing Trust in the Emerging Era of IoT, in: 2016 IEEE Symposium on Service-Oriented System Engineering (SOSE), pp. 398–406. doi:10.1109/SOSE.2016.50.

[21] C. Agostino Ardagna, R. Asal, E. Damiani, N. El Ioini, C. Pahl, Trustworthy IoT: An Evidence Collection Approach Based on Smart Contracts, in: 2019 IEEE International Conference on Services Computing (SCC), IEEE, pp. 46–50. doi:10.1109/SCC.2019.00020.
URL https://ieeexplore.ieee.org/document/8814228/

[22] K. Bilal, A. Erbad, Edge computing for interactive media and video streaming, in: 2017 Second International Conference on Fog and Mobile Edge Computing (FMEC), pp. 68–73. doi:10.1109/FMEC.2017.7946410.

[23] Face recognition: Moving more and more to the edge - asmag.com.
URL https://www.asmag.com/showpost/24585.aspx

[24] I. Farris, T. Taleb, Y. Khettab, J. Song, A Survey on Emerging SDN and NFV Security Mechanisms for IoT Systems 21 (1) (Firstquarter 2019) 812–837 (Firstquarter 2019). doi:10.1109/COMST.2018.2862350.

[25] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, J. Lopez, A Survey of IoT-enabled Cyberattacks: Assessing Attack Paths to Critical Infrastructures and Services 1–1doi:10.1109/COMST.2018.2855563.

[26] ITU-T Y.4806 Security capabilities supporting safety of the Internet of things.

[27] A. Mosenia, N. K. Jha, A Comprehensive Study of Security of Internet-of-Things 5 (4) 586–602. doi:10.1109/TETC.2016.2606384.

[28] M. H. Onik, N. Al-Zaben, H. P. Hoo, C.-S. Kim, A Novel Approach for Network Attack Classification Based on Sequential Questions 2 (2) 14.

[29] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness.

[30] B. Ruhnau, Eigenvector-centrality—a node-centrality? 22 (4) 357–365.

[31] P. Bonacich, Power and centrality: A family of measures 92 (5) 1170–1182.

[32] G. Sabidussi, The centrality index of a graph 31 (4) 581–603.

[33] M. Dehmer, F. Emmert-Streib, Quantitative Graph Theory: Mathematical Foundations and Applications, Discrete Mathematics and Its Applications, CRC Press.
URL https://books.google.fr/books?id=NcrMBQAAQBAJ

[34] L. C. Freeman, A Set of Measures of Centrality Based on Betweenness 40 (1) 35–41. arXiv:3033543, doi:10.2307/3033543.

[35] F. Brockners, S. Bhandari, S. Dara, C. Pignataro, H. Gedler, S. Youell, J. Leddy, D. Mozes, T. Mizrahi, Proof of Transit.

[36] Social Internet of Things.
URL http://www.social-iot.org/index.php

[37] SmartSantander.
URL http://www.smartsantander.eu/

[38] Smart Streetlights Program — Sustainability — City of San Diego Official Website.
URL https://www.sandiego.gov/sustainability/energy-and-water-efficiency/programs-projects/smart-city

[39] Chicago smart lighting program.
URL https://chicagosmartlighting-chicago.opendata.arcgis.com/pages/for-residents

[40] E. Garcetti, IES SALC Conference 2016 - Mayor Garcetti Speech.
URL https://www.youtube.com/watch?v=Ii0etmvV2e4

[41] Smart Street Lighting.
URL https://www.nec.com/en/global/solutions/streetlight/index.html