View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Sapientia

LUÍS MANUEL PISCO RODRIGUES

DATA STORAGE SOLUTIONS FOR THE FEDERATION OF SENSOR NETWORKS

UAlg FCT
UNIVERSIDADE DO ALGARVE
FACULDADE DE CIÊNCIAS E TECNOLOGIA

2020

LUÍS MANUEL PISCO RODRIGUES

DATA STORAGE SOLUTIONS FOR THE FEDERATION OF SENSOR NETWORKS

PhD Thesis in Computer Science

Work done under the supervision of:
Professora Doutora Noélia Susana Costa Correia

UAlg FCT
UNIVERSIDADE DO ALGARVE
FACULDADE DE CIÊNCIAS E TECNOLOGIA

2020

Statement of Originality

Data Storage Solutions for the Federation of Sensor Networks

Declaração de autoria de trabalho: Declaro ser o autor deste trabalho, que é original e inédito. Autores e trabalhos consultados estão devidamente citados no texto e constam da listagem de referências incluída.

Candidato:

(Luís Manuel Pisco Rodrigues)

Copyright ©Luís Manuel Pisco Rodrigues. A Universidade do Algarve reserva para si o direito, em conformidade com o disposto no Código do Direito de Autor e dos Direitos Conexos, de arquivar, reproduzir e publicar a obra, independentemente do meio utilizado, bem como de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição para fins meramente educaciomais ou de investigação e não comerciais, conquando seja dado o devido crédito ao autor e editor respetivos.

CEOT
Networks & Systems

Work done at Research Center of Electronics Optoelectronics and Telecommunications (CEOT)

Acknowledgements

I would like to express my deep and sincere gratitude to my supervisor, Prof. Dra. Noélia Correia for all the support, dedication and availability without which it would not have been possible to accomplish this long journey, and also for the opportunity to research and learn about this very interesting world of the Internet of Things.

I would also like to thank Dr. Joel Guerreiro, Director of the Informatics Services of the University of Algarve, for all the motivation, support and help for the completion of this course.

Of course, this would not have been possible without the understanding of my family, my sons Tiago and João Miguel and my beloved wife Margarida. Thank you for your understanding and support and for the precious time you have given me to dedicate to this project.

Finally, I would like to thank my parents, Hermínia and José Manuel, who provided the conditions in my base studies that allowed me to reach this stage. I would like to dedicate this thesis to my father, who is no longer with us, but who through his solid education contributed a lot to many of my successes, and who would certainly be very proud with the completion of this step.

Abstract

In the near future, most of our everyday devices will be accessible via some network and uniquely identified for interconnection over the Internet. This new paradigm, called Internet of Things (IoT), is already starting to influence our society and is now driving developments in many areas.

There will be thousands, or even millions, of constrained devices that will be connected using standard protocols, such as Constrained Application Protocol (CoAP), that have been developed under certain specifications appropriate for this type of devices. In addition, there will be a need to interconnect networks of constrained devices in a reliable and scalable way, and federations of sensor networks using the Internet as a medium will be formed.

To make the federation of geographically distributed CoAP based sensor networks possible, a CoAP Usage for REsource LOcation And Discovery (RELOAD) was recently proposed. RELOAD is a peer-to-peer (P2P) protocol that ensures an abstract storage and messaging service to its clients, and it relies on a set of cooperating peers that form a P2P overlay network for this purpose. This protocol allows to define so-called Usages for applications to work on top of this overlay network. The CoAP Usage for RELOAD is, therefore, a way for CoAP based devices to store their resources in a distributed P2P overlay. Although CoAP Usage for RELOAD is an important step towards the federation of sensor networks, in the particular case of IoT there will be consistency and efficiency problems. This happens because the resources of CoAP devices/Things can be in multiple data objects stored at the overlay network, called P2P resources. Thus, Thing resource updates can end up being consuming, as multiple P2P resources will have to be modified. Mechanisms to ensure consistency become, therefore, necessary.

This thesis contributes to advances in the federation of sensor networks by proposing mechanisms for RELOAD/CoAP architectures that will allow consistency to be ensured. An overlay network service, required for such mechanisms to operate, is also proposed.

Keywords: Federated Networks, Sensor Networks, RELOAD, CoAP, CoAP Usage.

Resumo

Num futuro próximo, a maioria dos nossos dispositivos do dia-a-dia estarão acessíveis através de uma rede e serão identificados de forma única para poderem interligar-se através da Internet. Este novo paradigma, conhecido hoje por Internet das Coisas (IoT), já está a começar a influenciar a nossa sociedade e está agora a impulsionar desenvolvimentos em inúmeras áreas.

Teremos milhares, ou mesmo milhões, de dispositivos restritos que utilizarão protocolos padrão que foram desenvolvidos de forma a cumprir determinadas especificações associadas a este tipo de dispositivos, especificações essas que têm a ver com o facto destes dispositivos terem normalmente restrições de memória, pouca capacidade de processamento e muitos possuirem limitações energéticas. Surgirá ainda a necessidade de interligar, de forma fiável e escalonável, redes de dispositivos restritos. Estas federações de redes de sensores utilizarão a Internet como meio de comunicação. O IPv6 sobre *Low Power Personal Area Network* (6LoWPAN) foi proposto neste contexto como o protocolo de rede que deverá ser usado para ligar dispositivos restritos à Internet, enquanto que o *Constrained Application Protocol* (CoAP), que se enquadra na camada de aplicação, é proposto como o protocolo a adotar para a transferência de recursos Web armazenados em dispositivos com restrições. Isto é, estes protocolos foram desenhados para operar em dispositivos e redes com limitações, fazendo com que dispositivos restritos possam fazer parte da Internet.

Para ser possível a federação de redes de sensores que estão geograficamente dispersas, e tendo em consideração que o CoAP será o protocolo da camada de aplicação a ser adotado para transferência de recursos Web, foi proposto recentemente um CoAP Usage para o protocolo *REsource LOcation And Discovery* (RELOAD). O RELOAD é um protocolo *Peer-to-Peer* (P2P) que assegura um serviço de armazenamento abstrato e a distribuição de conteúdos aos seus clientes, e que assenta num conjunto de nós, chamados *peers*, que cooperam entre si de forma a criarem uma rede virtual, chamada de rede *overlay*, para armazenamento distribuido de conteúdos. Este protocolo permite a definição de *Usages* para que diferentes aplicações ajustem o funcionamento da rede *overlay* aos seus dados e ao seu objetivo. O Usage descreve os tipos

de dados (*data Kinds*) específicos dessa aplicação, e algum comportamento (por exemplo, políticas de acesso). A utilização de CoAP para RELOAD acaba então por ser uma forma de os dispositivos baseados em CoAP armazenarem os seus recursos numa rede P2P distribuida, podendo haver uma pesquisa dos recursos existentes por parte de clientes interessados nesses recursos. Embora a utilização de CoAP para RELOAD seja um passo importante para a federação de redes de sensores, no caso particular da IoC existirão problemas de consistência e eficiência. Isto vem do facto de um objeto de armazenamento na rede *overlay* P2P, chamado recurso P2P, puder incorporar recursos de diferentes dispositivos/Coisas, que são agrupados para efeitos de publicação na rede P2P. Mais concretamente, poderá haver a necessidade de agrupar os dispositivos de acordo com o seu tipo ou objetivo de estudo em particular (por exemplo, "temperatura" poderá ser um recurso P2P que agrega vários leituras/recursos de vários sensores, que podem estar localizados em diferentes lugares). Os recursos dos dispositivos/Coisas podem ainda participar em múltiplos recursos P2P. Assim sendo, eventuais atualizações de recursos de dispositivos/Coisas poderão desencadear múltiplas atualizações de recursos P2P, sendo este processo muito consumidor para a rede P2P (são geradas várias operações de *fetch* e *store*). Consequentemente, deverão ser encontrados mecanismos que permitam assegurar a consistência dos conteúdos anunciados nestas redes P2P baseadas em RELOAD/CoAP.

Esta dissertação de doutoramento contribui para o avanço do estado da arte na federação de redes de sensores, propondo mecanismos que permitem melhorar a organização e pesquisa de recursos nas arquiteturas RELOAD/CoAP, garantindo a consistência dos conteúdos anunciados. Esta característica é extremamente importante quando se antecipa que estas arquiteturas poderão vir a ter de lidar com milhões de recursos disponíveis em dispositivos espalhados por inúmeras redes geograficamente dispersas. Mais concretamente, esta dissertação apresenta abordagens para criação e manutenção de recursos P2P anónimos, que são criados com o objetivo de assegurar a unicidade, e consequentemente consistência, dos conteúdos P2P. Os recursos P2P originais passam a apontar para estes recursos P2P anónimos. São propostas abordagens de otimização matemática e heurísticas não só para o problema de reestruturação de um conjunto de recursos P2P originais, armazenados na rede *overlay*, mas também para o problema de como reorganizar os conteúdos quando chegam novos anuncios de recursos P2P, ou quando estes são removidos (procedimentos *on-demand*). É ainda discutida a operação de um serviço extra que será necessário para manter os vínculos (entre recursos P2P originais e anónimos) atualizados e consistentes. Este serviço garante

também que os clientes, aquando das suas pesquisas, irão receber os recursos P2P com o formato original (conteúdo que cumpre a especificação padrão do CoAP Usage), fazendo com que os clientes não se apercebam da existencia de recursos P2P anónimos. Ou seja, o recurso a anónimos acaba por ser transparente para as aplicações cliente, evitanto que estas tenham que cumprir outras especificações para além da especificação padrão CoAP Usage.

A utilização de estruturas P2P para o armazenamento de recursos IoT tem várias vantagens: *i*) os nós participantes (nós *gateway* ou *proxies* no caso das redes de sensores) contribuem para o armazenamento e processamento associado à rede *overlay*, o que faz com que estas sejam arquiteturas que não têm problemas de escalabilidade; *ii*) os *peers* na rede *overlay* comunicam diretamente, não sobrecarregando nenhum servidor em particular e evitando pontos de estrangulamento de largura de banda; *iii*) podem ser criados túneis de comunicação encriptados para assegurar a existência de segurança na entrega dos dados. As contribuições apresentadas nesta dissertação vêm contribuir ainda para passarmos a ter um armazenamento e gestão dos recursos IoT mais eficiente (dentro da rede *overlay*). Todos estas caracteristicas acabam por ser elementos facilitadores do surgimento de um maior número de federações de redes de sensores. A disponibilização de diferentes agregados de recursos IoT, com recurso a estas arquiteturas distribuidas, permitirá, por sua vez, que surjam aplicações com uma natureza inovadora e que têm por base a integração de recursos IoT.

Termos chave: Redes Federadas, Redes de Sensores, RELOAD, CoAP, CoAP Usage.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

6LoWPAN	:	IPv6 over Low-Power Wireless Area Networks
CoAP	:	Constrained Application Protocol
CoRE	:	Constrained RESTful Environment
DHT	:	Distributed Hash Tables
DTLS	:	Datagram Transport Layer Security
HTTP	:	HyperText Transfer Protocol
ICE	:	Interactive Connectivity Establishment
IETF	:	Internet Engineering Task Force
ILP	:	Integer Linear Programming
IoC	:	Internet das Coisas
IoT	:	Internet of Things
IP	:	Internet Protocol
LoWPANs	:	Low power Wireless Personal Area Networks
M2M	:	Machine-to-Machine
MQTT	:	Message Queuing Telemetry Transport
NAT	:	Network Address Translation
NFV	:	Network Function Virtualization
P2P	:	Peer to Peer
QoS	:	Quality of Service
RD	:	Resource Directory
ReDiR	:	Recursive Distributed Rendezvous
RELOAD	:	REsource LOcation And Discovery
REST	:	Representational State Transfer
SDN	:	Software-Defined Networking
SIP	:	Session Initiation Protocol
TCP	:	Transmission Control Protocol
TLS	:	Transport Layer Security
TURN	:	Traversal Using Relays around NAT
UDP	:	Transmission Control Protocol

URI	:	Uniform Resource Identifier
WoT	:	Web of Things
WSN	:	Wireless Sensor Networks

Sets

$\mathcal{R}$: Original set of P2P resources.

$\bar{\mathcal{R}}$: Redesign of $\mathcal{R}$.

$\mathcal{R}_E(\mathcal{R})$: P2P resources after replacing content by bindings, and when using as goal the minimization of device resource entries.

$\mathcal{R}_F(\mathcal{R})$: P2P resources after replacing content by bindings, and when using as goal the minimization of fetches.

$\mathcal{P}_r$: Set of `(KEY,VALUE)` entries inside P2P resource $r \in \mathcal{R}$.

$\mathcal{P}'_r$: Redesign of $\mathcal{P}_r$.

$\mathcal{E}_{p_r}$: Device resource entries inside `(KEY,VALUE)` entry $p_r \in \mathcal{P}_r$.

$\mathcal{E}^r$: All device resource entries from every `(KEY,VALUE)` entry in $r \in \mathcal{R}$.

$\mathcal{E}$: All device resource entries from every `(KEY,VALUE)` entry, and from every P2P resource in the overlay network.

$\mathcal{S}^r$: Family of P2P resources whose device resource entries form a subset of $\mathcal{E}^r$.

Γ_n : Children nodes of n, where n represents a resource (either P2P or device entry) and is part of a dependency graph.

$\mathcal{A}$: Anonymous P2P resources.

$\mathcal{G}$: Compatibility graph.

$\mathcal{N}$: Nodes in compatibility graph $\mathcal{G}$.

$\mathcal{L}$: Links in compatibility graph $\mathcal{G}$.

$\mathcal{N}_n$: Neighbours of node $n \in \mathcal{N}$ in compatibility graph $\mathcal{G}$.

$\mathcal{C}$: Clique inside compatibility graph $\mathcal{G}$.

$\mathcal{X}(\mathcal{C})$: Nodes that can expand clique $\mathcal{C}$ in compatibility graph $\mathcal{G}$.

Known Values

z : In a maximum cover problem, it represents the maximum number of subsets that can be used when maximizing the covering of elements in set $\mathcal{E}^r$.

Δ : Big value.

k : Pool size increase factor.

m : Random number used to populate a P2P resources with sensor entries.

M : Maximum number of sensor entries in a P2P resource.

Functions

$w(n)$: Weight of node n, where n represents a resource (either P2P or device entry) and is part of a dependency graph.

$\kappa(e)$: Returns the number of P2P resources including device entry $e \in \mathcal{E}$.

$T(r)$: Returns the type of P2P resource, which is either "Anonymous" or "KeyValue".

Variables

y_e : In a maximum cover problem, states if device entry e is being covered or not (binary).

x_r : In a maximum cover problem, states if resource r is selected for the cover or not (binary).

τ_a : One if anonymous P2P resource $a \in \mathcal{A}$ is in use, zero otherwise.

β_e^u : One if P2P resource $u \in \bar{\mathcal{R}} \cup \mathcal{A}$ includes device resource $e \in \mathcal{E}$ as an entry, zero otherwise.

$\alpha_e^{u,u'}$: One if P2P resource $u \in \bar{\mathcal{R}} \cup \mathcal{A}$ receives $e \in \mathcal{E}$ through a reference to P2P resource $u' \in \bar{\mathcal{R}} \cup \mathcal{A} \backslash \{u\}$ (u includes binding to u'), zero otherwise.

γ_e^u : One if P2P resource $u \in \bar{\mathcal{R}} \cup \mathcal{A}$ is a provider of $e \in \mathcal{E}$ to others, zero otherwise.

$\xi^{u,u'}$: One if P2P resource $u \in \bar{\mathcal{R}} \cup \mathcal{A}$ if fed by P2P resource $u' \in \bar{\mathcal{R}} \cup \mathcal{A} \backslash \{u\}$, zero otherwise.

CHAPTER 1

Introduction

1.1 Motivation and Scope

The idea behind the *Internet of Things* (IoT) is for everyday objects to be connected, IP-addressable and integrated into the Internet [30, 51]. This has been boosted by the availability of more affordable wireless modules, making such technologies attractive for various applications. From the available technologies, short-range radio technologies like IEEE 802.15.4 are suitable for *Low power Wireless Personal Area Networks* (LoWPANs), while it is foreseen that cellular mobile connectivity will become the dominant technology for wide area sensor networking [8, 32].

The IoT brings the opportunity for Things to connect globally and exchange data within the existing Internet infrastructure. This means that sensing devices may interconnect for wide-area coverage, eventually cooperating to accomplish certain tasks, allowing for wide-area complex applications to be developed (e.g., environmental monitoring, earthquake/tsunami early-warning systems, correlation of health data, and so on) and also direct *Machine-to-Machine* (M2M) communication. Since 5G technologies meet the requirements of mobile communications and needs for Thing data transmission, these are expected to have a key role in such scenarios, enabling new platforms for global connectivity to emerge [5, 14]. Such platforms may rely on the federation of multiple sensor networks.

In the context of wide-area sensor networks, and federated networks, some open Internet standards become important. For IP-based communications, the *Internet Engineering Task Force* (IETF) defined the *IPv6 over Low-Power Wireless Area Networks* (6LoWPAN) standard that enables the use of IPv6 over very constrained networks [39]. The IP protocol emerges, therefore, as the glue to interconnect heterogeneous devices [52, 56]. Also withing IETF, the *Constrained RESTful Environments* (CoRE) working group has focused on the development of *Constraint Application Protocol* (CoAP), a Web application transfer protocol intended to provide RESTful services in constrained nodes and networks [21]. This protocol together with *REsource LOcation And Discovery* (RELOAD) base protocol, another very relevant open Internet standard for the federation of sensor networks that provides a generic self-organizing *Peer-to-Peer* (P2P) overlay network service [18], provide the building blocks for the federation of wide-area sensor networks.

The RELOAD supports several applications through the use of "Usages". A Usage defines how an application maps its data into something (data object) that can be stored in the overlay, how resources stored at the overlay are identified, how to secure the data, and how data can be retrieved. Although constrained devices will be heterogeneous regarding their radio layer (e.g., long range modules: 4G, 5G; short range modules: xbee, zigbee), CoAP is expected to be a common application layer transfer protocol and, for this reason, a CoAP Usage for RELOAD is proposed in [20]. This brings the opportunity for sensor networks to be federated. That is, proxy nodes in constrained environments, with enough capacity to run RELOAD (either as client or peer), will be able to form a P2P overlay network themselves to announce resources, and clients will be able to discover the available resources. P2P network architectures have gained popularity in recent decades for being dynamic and scalable, becoming efficient architectures for content distribution [4, 10, 50].

Data objects stored in an overlay network are called P2P resources. For the announcement of resources available at devices, P2P resources in RELOAD/CoAP architectures will include references to physical device resources. These references allow clients to reach device resources using the CoAP protocol. Although RELOAD/CoAP is a standard-based scalable architecture, a desirable feature when millions of objects of all kinds are expected to be integrated into the IoT, such approach alone can not completely address another quite important issue: *an efficient storage and lookup of resources*. More specifically, the just mentioned CoAP Usage allows device resources to be announced under different P2P resource umbrellas and, for this reason, different groupings of device resources (according to their type or having similar characteristics) can be built and announced as P2P resources, even if device resources originate from different constrained environments. This poses, however, consistency and efficiency problems to such distributed storage systems because there will be similar device resource references in multiple P2P resources. Whenever a device resource reference or sensor value changes, multiple P2P resources must be updated. This becomes critical as more and more objects integrate the IoT because combinations of resources, for announcement purposes, will increase exponentially. New mechanisms for an efficient storage and lookup of resources are, therefore, required. These mechanisms should rely on adequate optimization models or heuristics for high quality decisions.

1.2 Objectives

The overall objectives of this thesis are the following:

- Contribute to a better understanding of protocol chains in IoT networks, namely those involving 6LoWPAN, CoAP, and CoAP for RELOAD.
- Contribute to the development of more efficient architectures for sensor network federation.
- Develop scalable optimization procedures to keep data consistent in RELOAD/CoAP architectures.
- Encourage the emergence of applications with an innovative nature and based on the integration of IoT resources.

1.3 Contributions

This work has the following contributions:

- Survey on the federation of constrained devices and networks, by going through their characteristics, related standards and proposals from the literature.

- Development of a resource binding model for P2P resources (objects stored in RELOAD/CoAP architectures) to be able to include bindings to other P2P resources available in the overlay network.

- Development of a multi-layer model for P2P resources to be able to keep information at the overlay network consistent, avoiding duplicates.

- Development of optimization procedures and heuristic algorithms for the implementation of the previously mentioned binding and multi-layer models.

- Proposal of a RELOAD/CoAP overlay network service for the operationalization of procedures.

1.4 Thesis Outline

Chapter 1 introduces IoT networks and frames the expected growth in device connectivity. It also presents the main protocols used for the federation of constrained devices and networks, and discusses aspects that need to be improved.

Chapter 2 describes in more detail the standards used in the federation of constrained devices and networks, which is required to frame the contributions that follow.

Chapter 3 proposes the use of resource bindings in P2P resources so that there is the possibility of referencing other existing P2P resources. An extra overlay service, required for the operationalization of bindings, is proposed.

Chapter 4 proposes a multi-layer approach that allows an efficient storage/retrieval of IoT data through the use of anonymous P2P resources. An extra overlay service, required for the operationalization of such approach, is also proposed.

In Chapter 5, on demand procedures to deal the the dynamic arrival/removal of P2P resources are proposed.

Chapter 6 concludes the dissertation by summarizing the features and role of IoT network federations, and frame the contributions includes in this thesis. Future work is also discussed.

CHAPTER 2

Federation of Wireless Sensor Networks

2.1 Introduction

Millions of objects of all kinds are expected to be part of the IoT, and the number of devices communicating over large geographical areas will increase for sure. To support such growth in the number and type of devices, and allow the creation of sensor network federations, adequate standard-based network architectures are required. Cloud storage solutions are now quite popular but a lot of extra storage space and processing can be required if these are used for IoT or federation of *Wireless Sensor Networks* (WSNs). Interconnecting hundreds of millions of sensors can become too demanding. These storage places are also exposed to attacks because they are known as storage places. P2P architectures, on the other hand, do not have these issues not only because the participating pairs/proxies also contribute to storage and processing, but also because pairs communicate directly, not overloading servers and avoiding bandwidth bottlenecks. Encrypted communication tunnels can also be used for security [46].

Organizations are gradually accepting the idea of having multiple independent peers accomplising tasks cooperatively in a self-organized shared environment, rather than having client-server architectures [46]. Designing these systems to be scalable solutions, while building a cohesive system of multiple autonomous computers/devices, can be a challenge [11]. Different P2P systems and protocols have been proposed and used in many application domains (e.g., content delivery, file sharing, multimedia, and other [7, 36, 42]). In this thesis the application domain is the federation of Things and sensor networks. This chapter introduces the protocols and systems applied in this context.

This chapter is organized as follows. Section 2.2 discusses CoRE related standards recently proposed for IoT. Section 2.3 discusses standards required to build self-organizing P2P overlay networks. Based on these, Section 2.4 presents an architecture that is suitable for the federation of Things and sensor networks. Section 2.5 presents work related with all these standards and architectures, and finally Section 2.6 presents a summary of the chapter.

Contributions:

- Survey on the federation of Things and sensor networks. Besides discussing features and related standards, proposals from literature are also presented.

2.2 CoRE Related Standards

The *Constrained RESTful Environment* (CoRE) working group, within *Internet Engineering Task Force* (IETF), is focused on the development of the *Representational State Transfer* (REST) architecture for constrained nodes [55]. In constrained environments, the possibility of discovering resources hosted by constrained nodes is important for applications to run without human intervention, and for flexible interfaces to be provided. Web discovery and linking was initially specified for *HyperText Transfer Protocol* (HTTP) in [40, 41]. In the context of constrained nodes, the discovery of resources, their attributes and relations is referred to as CoRE Resource Discovery [49].

2.2.1 CoAP

While CoRE aims at realizing the REST architecture in a suitable form for constrained nodes and networks, CoAP emerges as the Web application transfer protocol that has been designed for the special requirements of these constrained environments [21]. CoAP provides a request/response interaction model between application endpoints and both `coap` and `coaps` URI schemes can be used to identify CoAP resources. CoAP URI supports the path suffix `/.well-known/core` so that clients can discover resources available at the host, or discover any policy/information about the host, before making a request [41]. The main features of CoAP can be summarizes as follows [21]:

- Web protocol fulfilling M2M requirements in constrained environments.
- UDP binding with optional reliability supporting unicast and multicast requests.
- Asynchronous message exchanges.
- Low header overhead and parsing complexity.
- URI and Content-type support.
- Simple proxy and caching capabilities.
- Stateless HTTP mapping, which allows proxies to be built providing access to CoAP resources via HTTP in a uniform way.
- Security binding to Datagram Transport Layer Security (DTLS).

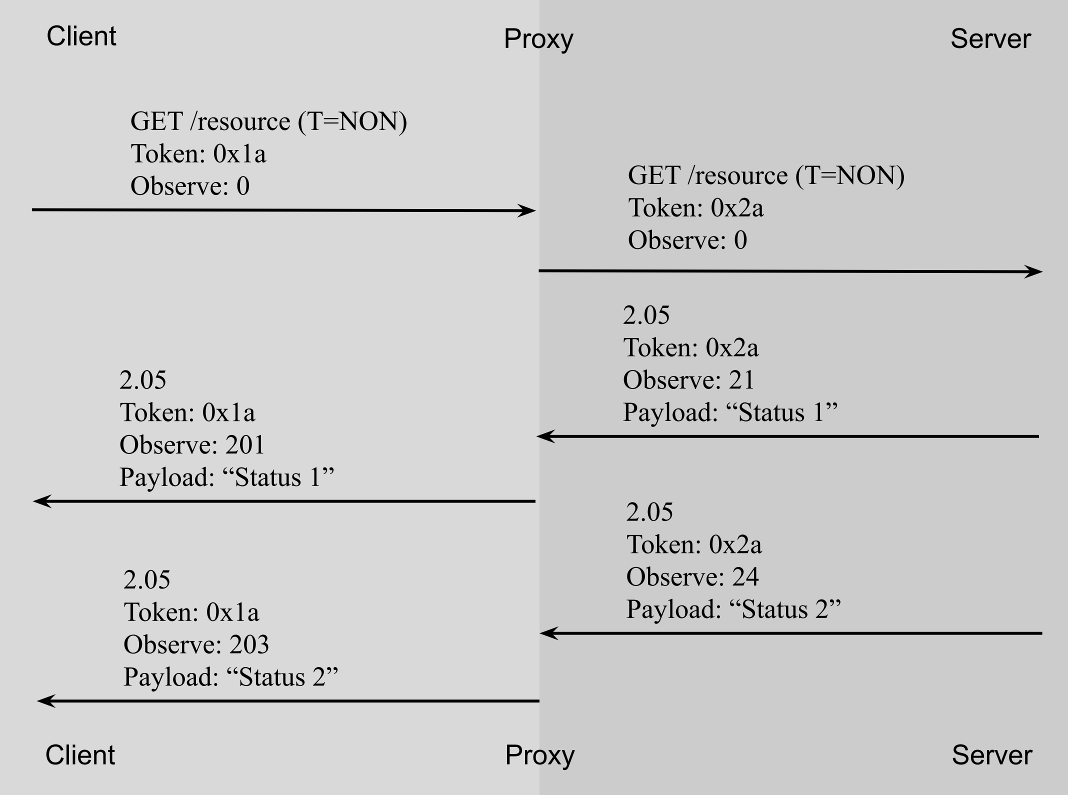


Figure 2.1: Illustration of CoAP Observe operation.

In [25], the Observe extension to CoAP is proposed for clients to be able to observe resources, and keep representations updated over time. After discovering a resource the observer/client can obtain the resource values by sending an extended GET request either to the server, having such resource in its namespace, or to a proxy to be used as an intermediate. The server, or proxy, will register the client as an observer, so that the client starts to receive notifications, and responds with an extended response. Extended requests/responses are CoAP requests/responses with an Observe option. Notifications can be kept in cache until they do not expire, which is controlled by the max-age CoAP option. Multiple proxies can be used for scalability purposes [3, 38]. Figure 2.1 illustrates CoAP Observe operation.

CoAP vs MQTT

The *Message Queuing Telemetry Transport* (MQTT) is now a quite popular protocol in IoT. Since its primary use was to connect oil pipelines via satellite, the initial requirements were simple implementation, lightweight and bandwidth efficiency, quality of service, data agnosticity and continuous session awareness. This protocol was used by IBM until 2010, and at this time MQTT 3.1 was released as a royalty-free version [1]. By 2014 it became an OASIS

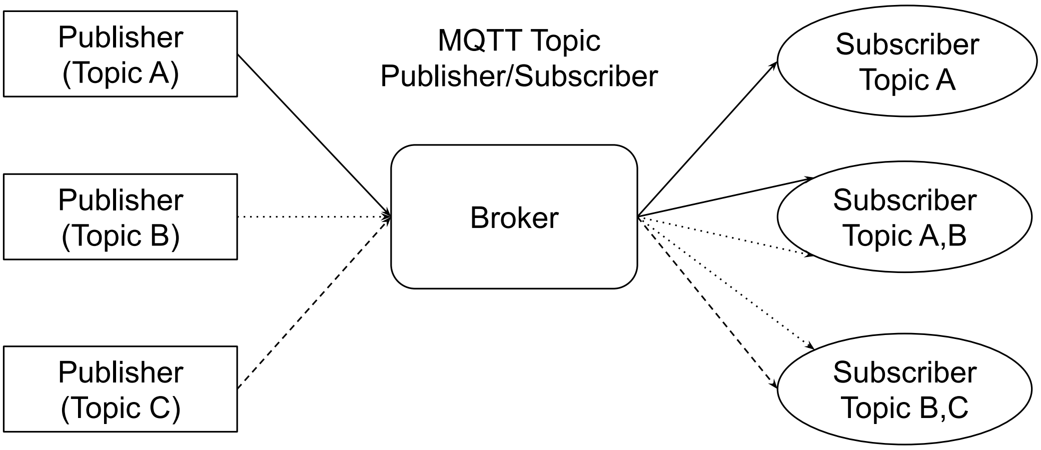


Figure 2.2: MQTT publish/subscriber model.

standard, and the curren version 5.0 was released on March 2019 [2].

MQTT uses a publish/subscriber model and relies on a central broker, also known as server, for message distribution (see Figure 2.2). The resources are grouped into topics, with no formal structure, that can be subscribed by clients. Whenever a publisher sends a message to the broker, such message will be distributed to the clients subscribing that topic, and then the message is removed. Because constrained networks are unreliable, MQTT provides three *Quality of Service* (QoS) levels: 0, if missing messages are acceptable; 1, for “at least one”; 2, for “at most one”. While MQTT uses *Transmission Control Protocol* (TCP), which allows a more robust connection but requires more resources, CoAP uses *Transmission Control Protocol* (UDP). CoAP needs, therefore, less resources from the client and relies on the application for connection reliability.

MQTT and CoAP are the most promising data transfer protocols for small devices. These are both open standards, more adequate to constrained environments than HTTP, allow asynchronous communication and both run on top on IP. But while MQTT acts purely as a pipe for binary data, although providing flexibility in communication patterns, CoAP has been designed for interoperability with the Internet and the Web. That is, CoAP was developed under an Internet Standard Document [21] and is based on the REST model. The goal of CoAP is to bring the REST model to small devices, becoming a lightweight analog of HTTP and easily interfacing with it for integration with the Web, while meeting the requirements of constrained networks and devices.

2.2.2 CoRE Link Format

Upon resource discovery at the CoRE default entry point `/.well-known/core`, specified in [49], a collection of resource links is obtained. These links follow the CoRE Link Format, also specified in [49], and an Internet media type has been assigned for CoRE Link Format payloads (`application/link-format`). An example of a collection of resource links is the following:

```
[
</sensors/temp-1>;rt="temperature-c";if="sensor",
</sensors/temp-2>;rt="temperature-c";if="sensor",
...
]
```

Each link description includes the URI reference plus a set of optional parameters (`rt`, `if`, `rel`, ...), all separated by semicolons. The `rt` parameter is used to assign an application-specific semantic type, either by indicating values/names (e.g., `temperature`) directly or by indicating a URI referencing a specific concept in an ontology (e.g., `http://sweet.jpl.nasa.gov/2.0/phys.owl#temperature`). The `if` parameter is used to specify the interface used to interact with that endpoint. This may include values/names (e.g., `sensor`) directly or a URI defining the interface (e.g., `http://www.example.org/myapp.wadl#sensor`). This way, the allowed methods, the request and the response are formally and precisely described. The `rel` describes the relation type between endpoints.

2.2.3 CoRE Resource Directory

As previously said, CoAP provides a request/response interaction model between endpoints, supporting discovery of resources, and the use of Web linking for description and discovery of resources, hosted by constrained servers, is specified by the CoRE Link Format in [49]. However, direct discovery of resources may not be practical due to sleeping nodes, and for this reason the use of a *Resource Directory* (RD) entity can be used [57]. Such entity would host descriptions of resources held on other servers, allowing lookups from others. For this to work the RD supports interfaces for:

- Discovery of the directory server;
- Registration, update and removal of resource descriptions;
- Lookup of resources, by clients, and group maintenance.

More recently, an alternative to this centralized resource directory approach has been proposed. Such approach has RELOAD as a basis and can be seen as a distributed RD. The RELOAD is discussed in Section 2.3, while the RELOAD/CoAP overlay (just mentioned distributed RD) is detailed in Section 2.4.

2.3 P2P/Overlay Network Standards

RELOAD is a generic P2P framework for the management of self-organizing P2P overlay networks [18]. Important features of RELOAD include security, Usage model, *Network Address Translation* (NAT) traversal, optimized routing and overlay algorithm extension capability.

RELOAD supports several applications through the use of "Usages" that specify application related data types, and rules for how to use services provided by RELOAD. That is, Usages describe specific data "Kinds" and behaviour related to the Usage (e.g., access policies). For example, a *Session Initiation Protocol* (SIP) Usage is defined in [19] and a CoAP Usage, defining a pluggable application layer for constrained networks, is proposed in [20]. Applications can, therefore, be defined on top of RELOAD, which can be seen as a generic overlay service, in order to provide their own services. Please note that an application can require multiple Usages, a Usage may define multiple Kinds of data to be stored in the overlay, and a Usage may rely on Kinds originally defined by other Usages [18].

RELOAD nodes can be clients or peers in the overlay. The term "peer" is used to identify a node in the overlay that is able to route messages towards nodes that are not directly connected to it, and has also storage responsibilities. A "client" refers to nodes that do not have routing or storage responsibilities [18]. The support for clients allows nodes not participating in the overlay (as peers) to utilize the same implementation and benefit from the same security mechanisms as the peers. Both clients and peers require RELOAD NodeIDs.

2.3.1 RELOAD Features

RELOAD is generic service providing several features that are critical for P2P success. In general, the important features of RELOAD are [18]:

- Security: A central enrollment server can be used to provide credentials to each peer, which are then used to authenticate operations.
- Usage model: RELOAD supports several applications through Usages that define specific data types, and rules for how to use services provided by RELOAD.
- NAT traversal: RELOAD utilizes *Interactive Connectivity Establishment* (ICE) to establish/use connections between nodes separated by one or more NATs or firewall.

- Optimized Routing: In a P2P network the participating peers will route requests on behalf of other peers, which introduces load in the form of bandwidth and processing power. In RELOAD the amount of effort for intermediate peers is minimized because a simple lightweight forwarding header is used.

- Overlay algorithm extension capability: An abstract interface to the overlay layer allows different overlay algorithms (e.g., *Distributed Hash Table* (DHT)) to be used. To instantiate a network, such generic structure of RELOAD is combined with an overlay algorithm that defines how the overlay topology is built and how messages are routed in it. DHT is the default overlay algorithm, meaning that its implementation is mandatory.

A RELOAD overlay instance can have a partly connected graph with RELOAD peers and RELOAD clients, each identified by a numeric Node-ID. A Node-ID plus the overlay algorithm determines the position of the node in the graph, and nodes it connects to. Different overlay algorithms will have different connectivity graphs.

A RELOAD network is a messaging and storage network. Each resource has a Resource-ID (numeric address) from the same space (set of numeric addresses) as node identifiers. A peer is responsible for storing data having a Resource-ID that fits in its storage range. Clients do not store or route information on behalf of other nodes on the overlay, and only use the overlay to locate resources/users and/or store information.

2.3.2 RELOAD Components

RELOAD has many components, described next [18, 43]:

- Usage Layer: Describes the data Kinds and behaviour related to each Usage. It is similar to the application OSI layer.

- Message Transport: Ensures reliable end-to-end message delivery, forwards certain RELOAD messages (e.g., Store and Fetch operations to/from the Storage component and packet forwarding to the Forwarding and Link Management Layer), manages the request state for the Usages, and delivers responses to the component initiating the request.

- Storage: RELOAD can be seen as a distributed database, where each peer stores part of the information. This component is responsible for processing messages related with storing/retrieval of data. It talks with the Topology Plug-in component in order to manage data replication and migration, and talks with the Message Transport component to send and receive messages.

- Topology Plug-in: Implements a specific P2P overlay algorithm. It defines the structure of the overlay network and processes for network formation and maintenance. Uses the Message Transport component to send its overlay management messages (e.g., link creation), and talks with Storage component for it to make any required data replication/migration. It talks with the Forwarding and Link Management Layer to control hop-by-hop message forwarding (Topology Plug-in maintains the overlay algorithm routing table).

- Forwarding and Link Management Layer: Provides the packet forwarding service between nodes. It also handles/sets links/connections between nodes, including links across NATs using ICE.

- Overlay Link Layer: While the previous components/layers fit into the application layer of the OSI model, this one directly relates to the Transport layer of the OSI model. It implements *Transport Layer Security* (TLS) and *Datagram Transport Layer Security* (DTLS) for TCP and UDP, respectively.

Besides these components, nodes may communicate with other central infrastructure to get: authentication credentials, initial set of nodes to communicate with when joining the overlay, or configuration information.

Message Transport

A component that is a client of the Message Transport can: *i*) send a message to a given peer specified by Node-ID or to the peer responsible for a particular Resource-ID (only if it is source; forwarding at intermediate hops is done by the Forwarding and Link Management Layer); *ii*) receive messages that other peers send to a Node-ID or Resource-ID for which the receiving peer is responsible (only if it is destination). Note that the Storage and Topology Plug-in are clients of Message Transport because they need to send/receive messages from other peers. All Usages are clients and rely on the Message Transport component to send/receive messages (e.g., store request) to/from peers.

End-to-end reliability is accomplished by timer-based (fixed timeout) retransmissions, and no congestion control is provided/required because RELOAD is a protocol with no more than two pairs of request-response messages in a typical transaction. The timer is adjusted at the overlay configuration (in a central infrastructure) and can, therefore, be dynamically updated at coarse time scales.

Message Storage

The NodeID of a peer determines the set of resources that the peer will be responsible for, in terms of storage. However, the exact mapping between these is specified by the overlay algorithm. When storing:

- The Store component only receives a store request from the Message Transport component if it is responsible for that Resource-ID.
- It then queries the appropriate Usage before storing data value(s) in its local data store (data can be signed and a safety confirmation can be done before storing).
- Sends response to Message Transport for delivery to the requesting node.

The Topology Plug-in component can notify a Message Storage component when the Resource-IDs, for which the latest is responsible, change and the Storage component is then responsible for migrating resources to other peers.

Topology Plug-in

This component is responsible for:

- Maintaining the overlay algorithm Routing Table, which is consulted by the Forwarding and Link Management Layer (for a particular NodeID or ResourceID) before forwarding a message. It issues periodic update requests through Message Transport component to maintain/update its Routing Table.
- When connections are made/broken (e.g., Internet network change) the Forwarding and Link Management Layer notifies the Topology Plug-in and this component: *i*) adjusts the Routing Table as appropriate; *ii*) instructs the Forwarding and Link Management Layer to form new connections as dictated by the requirements of the overlay algorithm topology.

- As peers enter/leave, resources may be stored on different peers, so this component keeps track of which peers are responsible for which resources (it may instruct the Storage component for replication/migration in order to ensure that peers have resources they are now responsible for).
- Redundant data storage to protect against loss of information in the event of peer failure or compromised/subversive peers.

Forwarding and Link Management Layer

This component is responsible for:

- Ensuring that a message arrives to the next peer, as determined by the Topology Plug-in component (a query is issued to the Topology Plug-in for Routing Table info of next hop). If the node is the destination then message is forwarded to the Message Transport. Besides such end-to-end messages, routing updates and data replicas can be exchanged.
- Establishment/maintenance of network connections, as determined by the Topology Plug-in component.
- Setting up connections to other peers through NATs and firewalls using ICE, and it can elect to forward traffic using relays for NAT and firewall traversal.
- Congestion control to protect the Internet paths/links used to form the link in the overlay, and retransmission is performed to improve the reliability of the end-to-end transactions. This is a link layer congestion control process while the Message Transport acts as an end-to-end process.

2.3.3 ReDir-based Service Discovery

In P2P overlay networks, like RELOAD overlay instances, peers share their computing resources to provide the service to which they were designed for [35]. For this system to work some peers may also provide other specific services to their counterparts, and any peer may request such services (e.g., *Traversal Using Relays around NAT* (TURN) relay service to assist in the traversal of NATs or firewalls) [18, 45, 44]. Therefore, peers face the problem of finding the set of peers providing a service. Although RELOAD specifies discovery mechanisms for specific services, like TURN for example, a generic service discovery mechanism is not part of the base protocol. For this reason the *Recursive Distributed Rendezvous* (ReDiR) service discovery mechanism, specified in [48], has been applied to RELOAD overlay networks in [35] in order to provide a generic service discovery mechanism. This is usefull if applications (e.g., SIP, CoAP) on top of RELOAD provide extra services. This will be the case of the proposals included in the following chapters.

A naive service discovery solution would be to store Node-IDs, of nodes providing a certain service, under some key k. This, however, will overload the node responsible for a service identified by key k. Such node not only might end up storing a large number of Node-IDs but also must answer all service lookup requests for that service. The ReDiR-based service discovery mechanism proposed for RELOAD avoids this ensuring that the load related with a certain service is distributed among the nodes providing the service. More specifically, a tree structure of the service providers is stored into the RELOAD overlay instance using Store and Fetch operations. Each node in the tree has pointers to service providers. Whenever a peer wishes to use that service its fetches the tree nodes until it finds a service provider responsible for its Node-ID. This way, no new functionalities from the RELOAD base protocol are required. For more details see [35].

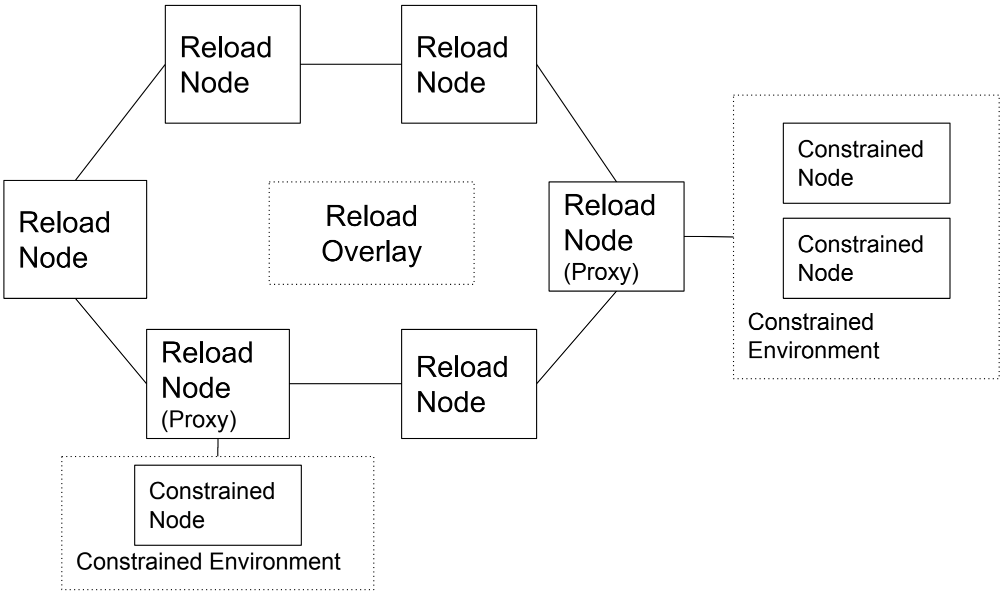


Figure 2.3: RELOAD/CoAP overlay network.

2.4 RELOAD/CoAP Architecture

RELOAD supports several applications through the use of Usages that specify application related data types, and rules for how to use services provided by RELOAD. For the particular case of IoT, and contrained environments in general, a CoAP Usage has been defined in [20]. This CoAP Usage allows a P2P overlay network to be built where sensor networks store their available resources, allowing such sensor networks to be federated while avoiding the use of centralized servers. Such overlay could be used:

- as a lookup service;
- to store available resources (e.g., sensor, controller);
- as a cache for sensor data.

A RELOAD/CoAP architecture is illustrated in Figure 2.3. Proxy/RELOAD nodes are located at the edge of WSNs, connecting them to Internet, and are assumed to have sufficient resources to run RELOAD. Wireless constrained nodes are the ones having limited computational capabilities and use sleep mode to prevent battery drain.

Regarding the advantages of a RELOAD/CoAP architecture, over a centralized RD entity, these include:

- Peers are able to redistribute any P2P resource they have cached, besides making sophisticated routing of queries among themselves, assisting any proxy/server in the delivery of P2P resources to clients;

- Since each peer only holds a portion of all `(KEY,VALUE)` pairs, a specific proxy is not overwhelmed by client requests;
- It solves the storage/lookup problem of P2P resources containing device resource entries managed by different proxies, which otherwise would have to be placed at multiple RD servers (assuming an RD server per constrained network, usually at the proxy) for clients to be aware of them. Such architecture brings, therefore, performance improvement for both clients and proxies/servers.

2.4.1 P2P Resources

RELOAD/CoAP nodes, usually proxies, can announce/store NodeID to resource link mappings, or sensor data, in the overlay. As an example, if a node participating at overlay `overlay1.com`, with NodeID `9996172`, wants to register a structure with its sensors, a mapping similar to the following would be used:

```
Resource-ID=h("coap://overlay-1.com/proxy-1/.well-known/
                                                    core")
KEY=9996172
VALUE=[
</sensors/temp-1>; rt="temperature-c"; if="sensor",
...
]
```

The `h(...)` is the hash over the URI returning a key for indexing purposes[1]. Nodes performing a lookup for `h("coap://overlay1.com/proxy1/.well-known/")` receive the information that the RELOAD node (proxy) with NodeID `9996172` is responsible for the device resources included in the corresponding `VALUE` structure. The `VALUE` provides paths to reach resources using CoAP protocol, and a direct connection (AppAttach) to the RELOAD node can be performed for CoAP instruction exchange. That is, after AppAttach negotiation, the requesting node can access the values (of constrained nodes) stored at the proxy (e.g., `GET /sensors/temp`).

Device resources managed by different nodes can also be announced under the same P2P resource umbrella. This is possible because the data model used in [20] is a dictionary. The following example shows a temperature related P2P resource where multiple RELOAD nodes contribute with device resources:

[1] Each RELOAD node stores part of the resources, according to their keys, resulting into a distributed and scalable storage system.

```
Resource-ID = h(coap://overlay-1.com/temperature/.well-
                                                known/core)
KEY =  9996172,
VALUE = [
   </sensors/temp-1>;rt="temperature-c";if="sensor",
   </sensors/temp-2>;rt="temperature-c";if="sensor"
]

KEY = 9996173,
VALUE = [
   </sensors/temp-a>;rt="temperature-c";if="sensor",
   </sensors/temp-b>;rt="temperature-c";if="sensor"
]
```

2.4.2 Caching Mechanisms

The overlay can also be used as a distributed caching mechanism for sensor information to be stored. This is very important because constrained devices can stay in sleep mode for long periods of time. Therefore, whenever a constrained node wakes up, it can send the most recent data from its sensors to its proxy, which stores the data in the overlay using a RELOAD StoredData structure. Two data models are defined for caching, in [20]: SensorCache and ProxyCache. The first is used for the storage of single data elements, while the second is used for the storage of multiple data elements. Since with ProxyCache multiple sensor values are stored, the corresponding CoAP URIs of sensing devices need to be part of the data being stored. The SensorCache, on the contrary, stores a single sensor value and, therefore, no URI needs to be stored because the URI is the same used to generate the Resource-ID.

2.5 Related Work

The concern for federation has arisen in several contexts. In [9, 22, 24, 34] the focus is cloud federation and aspects related with inter-cloud communication. The reason is that single cloud resources may not be sufficient to perform demanding tasks, and for this reason automated and standardised communication between infrastructures is required to share the resources. Federation-based architectures for scalable design of *Software-Defined Networking* (SDN) controllers in the optical communications ecosystem is addressed in [47]. A new controller architecture is proposed based on a federation of multiple sub-network controllers, each managing only a section of the network and coordinated by a hierarchically-superior controller. More recently the federation in the context of virtualization and 5G has been addressed [29, 6]. In [29], the focus is the end-to-end deployment of composite *Network Function Virtualization* (NFV) network services, which may involve multiple administrative domains and hence will require the federation of network services. In [6], the idea is to ensure the capability of providing networks and services tailored to events. An architecture is proposed that allows on-demand/dynamic creation and deployment of network slices over multiple domains for live content services. In the context of identity management systems, a P2P federated authentication system is proposed in [33]. Federation for energy sharing and trading has also been addressed in [13, 27]. In [13], the authors propose a P2P transactive multi-resource trading framework for the federation of multiple multi-energy microgrids. In [27], a P2P framework to support the federation of energy clusters is proposed. The idea is to study the interaction of consumers and producers in a market of energy resources and services.

The RELOAD/CoAP architecture for wide area sensor and actuator networking, using the previously mentioned CoAP application Usage for RELOAD, was initially proposed in [30]. The advantages of such architecture are discussed, which include integration with the Web, self-organization, scalability, robustness, and simulations are performed to compare its performance against a traditional client/server architecture. Federation of autonomous sensor networks that have to collaborate to achieve a common task is also discussed in [54, 53], although the focus is not on distributed systems, like P2P-based architectures. Their focus is on the federation of segments, resulting from failures or natural calamities, for the reestablishment of communications among segments. This is achieved through the deployment of relay nodes. In [54] two vertex distinct paths between every pair of WSNs is ensured, while the deployed relay nodes is minimized. In [53], constrained relay availability is considered. More specifically, the use of a limited number of mobile relays to provide intermittent inter-segment connectivity is considered.

The RELOAD/CoAP architecture has also been used for specific purposes. In [15, 16, 17], such architecture is used to serve network coding based constrained environments. The goal is for encoding vectors and encoded data to be stored at the P2P overlay, and a decoding service is used for packet recovery. In [16] there is also a planning of the placement of encoding nodes, while in [17] a directed acyclic graph is used to disseminate data towards the P2P overlay. Other works using RELOAD/CoAP architectures are not known in literature.

Simulators can be used for the analysis, testing and evaluation of P2P networks, before moving to a real network environment. Several simulators have been developed. Some are generic while others are protocol-specific. In general, generic P2P simulators allow modules to be customized and users to access core protocols or core application components. Examples of generic simulators include OverSim [26], PlanetSim [31] and PeerSim [37]. The GnutellaSim [23] is an example of a protocol-specific simulator for the evaluation of Gnutella systems.

2.6 Summary

This chapter presents several standards that are used for the federation of sensor networks. First, a set of CoRE related standards are presented and then RELOAD and its CoAP Usage are discussed. These allow RELOAD/CoAP architectures to be built where existing device resources and sensor data can be announced for sharing among nodes. A related work section is also presented.

CHAPTER 3

Resource Bindings in P2P Resources

3.1 Introduction

Millions of objects of all kinds are expected to be part of the IoT. In order to support such growth, open and standard-based network architecture solutions are required for interoperability between the various ecosystems. The RELOAD, together with the CoAP Usage, provides a rendezvous system for the lookup of resources and CoAP nodes storing these resources. This allows one to find which resources are served by a RELOAD node, by making a fetching using the hash of the corresponding URI.

Although RELOAD/CoAP architectures are standard-based and scalable solutions, a desirable feature when objects of all kinds are expected to be integrated into the IoT, these should also include a self-organizing way of working with resources, for efficiency and consistency purposes. More specifically, if a new P2P resource announcement includes device resource entries that are already available in the overlay network, although under a different P2P resource umbrella, then a mention to such existing P2P resource should be made, instead of populating the overlay network with duplicate content (a binding between P2P resources is required). This may be the case of P2P resources containing device resource entries managed by different proxy/RELOAD nodes. In such case, there will be duplication of device resource entries because each proxy/RELOAD node also announces its devices in a P2P resource. Such duplication can be avoided if bindings are used. However, this will require a resource binding service at the RELOAD/CoAP overlay network. This chapter discusses a resource binding model, and a heuristic algorithm for its implementation is proposed. Two different goals are discussed for such resource binding model, and their advantages and disadvantages are analysed having the heuristic results as a basis. Bindings are built having just existing P2P resources in consideration.

This chapter is organized as follows. Section 3.2 presents a resource binding service at the RELOAD/CoAP overlay network, which is required for P2P resource retrieval. Section 3.3 presents the resource binding model and then Section 3.4 presents the heuristic algorithm for its implementation. In Section 3.5 an analysis and discussion of results is presented, and finally Section 3.6 presents a summary of the chapter.

Contributions

The contributions presented in this chapter are the following:

- Development of a resource binding model for P2P resources to include bindings to other P2P resources available at the overlay network.
- Development of a heuristic algorithm for the implementation of the resource binding model.
- A resource binding service, at the RELOAD/CoAP overlay network, is proposed for operationalization.

These contributions were published in:

- L. Rodrigues, J. Guerreiro and N. Correia, "RELOAD/CoAP Architecture with Resource Aggregation/Disaggregation Service", IEEE PIMRC, IoT Workshop, September 2016, Spain.
- L. Rodrigues, J. Guerreiro and N. Correia, "RELOAD/CoAP Architecture for Federated M2M Communication", Journal of Peer-to-Peer Networking and Applications, Springer, 2019.

3.2 Resource Binding Service

3.2.1 Definitions and Assumptions

As previously mentioned, RELOAD nodes can have their device resource references in multiple P2P resources [20]. This way, device resources from multiple RELOAD nodes can be simultaneously fetched if these are under the same P2P resource. This is possible because the data model being used is a dictionary (dictionary key is the NodeID). Let us assume the following example where resources from two RELOAD nodes, with NodeIDs `9996172` and `9996173`, are available under the umbrella of P2P resource `temperature`:

```
Resource-ID = h(coap://overlay-1.com/temperature/.well-
                                                    known/)


KEY = 9996172,
VALUE = [
 </sensors/temp1>;rt="temperature";if="sensor",
 </sensors/temp2>;rt="temperature";if="sensor"
]

KEY = 9996173,
VALUE = [
 </sensors/tempA>;rt="temperature";if="sensor",
 </sensors/tempB>;rt="temperature";if="sensor"
]
```

That is, distinct nodes are making their device resources available under a single P2P resource umbrella. Note that the hash over `coap://overlay1.com/temperature/.well-known/` is just an overlay storage node locator. The nodes that have in fact temperature sensors are the ones with NodeIDs `9996172` and `9996173` (RELOAD NodeIDs are mapped to IP addresses at registration time).

The use of a dictionary data model, however, does not allow to separate device resources according to their type or some other feature. Multiple `KEY` entries with the same value are not allowed. Also, the P2P overlay network should include a self-organizing way of working with resources, for efficiency and consistency purposes, which involves providing a specific service for this purpose. More specifically, if some of the device resources included in a P2P resource announcement are already available in the overlay network, although under a different P2P resource umbrella, then a mention to such P2P resource should be made instead of duplicating device resource references at P2P resources. That is, a binding between P2P resources should be allowed. This prevents clients from having to update multiple P2P resources whenever device resource references change. Remember that federation of WSNs will exist, and announced P2P resources might include sensors from multiple entities. For a better understanding, let us assume a `heat-related-illness` resource wrapping up `temperature`, previously shown, and `co2` sensor entries:

```
Resource-ID = h(coap://overlay-1.com/heat-related-illness/
                                                .well-known/)

KEY = 9996172,
VALUE = [
 </sensors/temp1>;rt="temperature";if="sensor",
 </sensors/temp2>;rt="temperature";if="sensor",
 </sensors/co21>;rt="co2";if="sensor",
 ...
]

KEY = 9996173,
VALUE = [
 </sensors/tempA>;rt="temperature";if="sensor",
 </sensors/tempB>;rt="temperature";if="sensor"
]
```

Since `h(coap://overlay-1.com/temperature/.well-known/)`, defined above, and `h(coap://overlay-1.com/heat-related-illness/.well-known/)` have sensor entries in common, it is not efficient to announce `heat-related-illness` like this at the P2P overlay network because:

- future management (e.g., updates and removals) of information becomes harder; changing temperature sensor entries, for example, requires changing multiple P2P resources;
- any client announcing P2P resources made of public device resource links might not be aware of link changes;
- P2P resources can become quite populated with device resource entries making it difficult to visualize the different types of sensors included in

it (note that separating `temperature` and `co2` entries would require the same `KEY` value, which is not allowed).

For an efficient management of resources, the following change to `heat-related-illness` would bring benefits:

```
Resource-ID = h(coap://overlay-1.com/heat-related-illness/
                                                .well-known/)


KEY =  1116140,
VALUE = [
   </overlay-1.com/temperature>;if="virtual",
   </overlay-1.com/co2>;if="virtual"
]
```

Note that `/overlay-1.com/temperature` and `/overlay-1.com/co2` are not real devices, and for this reason can not be linked to NodeIDs `9996172` and `9996173`. These can be seen as virtual resources of RELOAD node with NodeID `1116140` that is implementing a binding service (the `if` is used to specify how to interact with the resource) and, therefore, able to manage virtual resources. Its operation is detailed next.

Naturally, both `h(coap://overlay-1.com/temperature/.well-known/)` and `h(coap:// overlay-1.com/co2/.well-known/)` must be available as resources at the P2P overlay network.

Operation of Resource Binding Service

As previously stated, resources of the same type, or with similar features, should be announced under a single P2P resource so that other resources can incorporate them. Such organization of data is expected to be very relevant in future IoT applications. The resource binding service performs as follows:

- Storage/search of CoAP resources is to be done through resource binding servers able to: *i*) determine the best redesign of virtual resource entries, upon creation of new P2P resource announcements; *ii*) ensure the fetching of virtual resource entries, so that all non-virtual device resources are provided to the clients.

- A discovery mechanism, like ReDiR, can be used to find resource binding servers, so that overload is distributed among servers. That is, scalability is ensured.

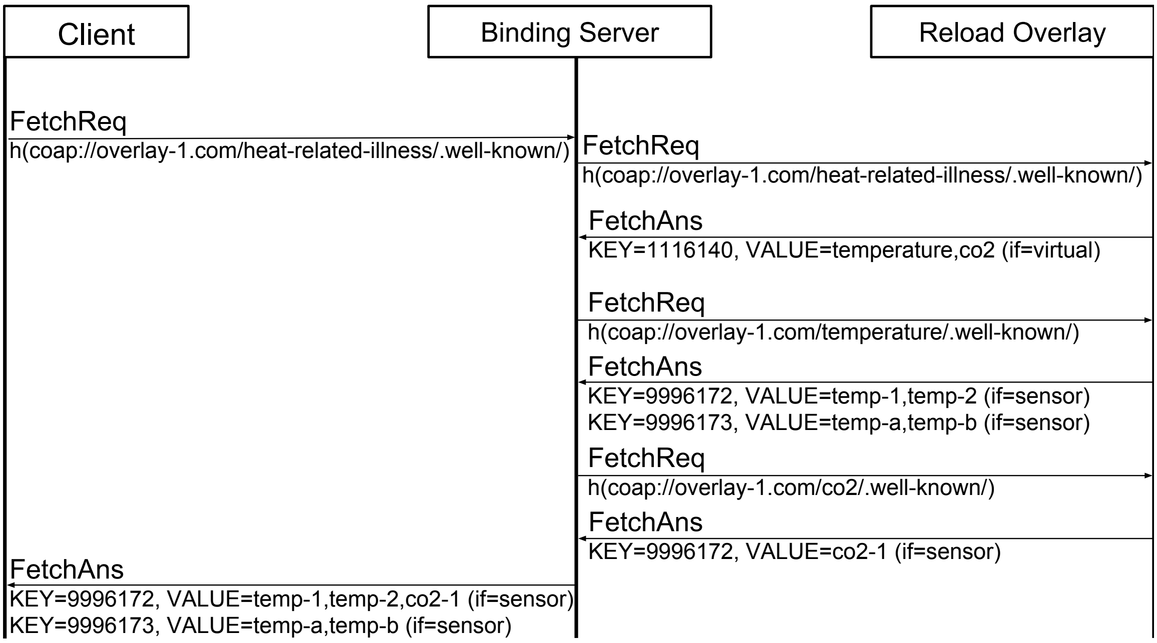

Client
Binding Server
Reload Overlay
FetchReq
h(coap://overlay-1.com/heat-related-illness/.well-known/)
FetchReq
h(coap://overlay-1.com/heat-related-illness/.well-known/)
FetchAns
KEY=1116140, VALUE=temperature,co2 (if=virtual)
FetchReq
h(coap://overlay-1.com/temperature/.well-known/)
FetchAns
KEY=9996172, VALUE=temp-1,temp-2 (if=sensor)
KEY=9996173, VALUE=temp-a,temp-b (if=sensor)
FetchReq
h(coap://overlay-1.com/co2/.well-known/)
FetchAns
KEY=9996172, VALUE=co2-1 (if=sensor)
FetchAns
KEY=9996172, VALUE=temp-1,temp-2,co2-1 (if=sensor)
KEY=9996173, VALUE=temp-a,temp-b (if=sensor)

Figure 3.1: Time diagram for use case scenario.

- The entries of virtual device resources must have as `KEY` the NodeID of a resource binding server. The entries of non virtual device resources will have as `KEY` the NodeID of the peer RELOAD node responsible for those device resources.

Figure 3.1 shows the time diagram, incorporating these features, for the previously discussed use case scenario. Note that RELOAD provides interfaces to the overlay layer, allowing a variety of overlay algorithms to be implemented (e.g., DHT). These algorithms, holding different connectivity graphs, have as major goal to allow any node in the graph to efficiently reach other nodes within a small number of hops. The fetches from the binding server to the overlay, shown in Figure 3.1, will use such interfaces. That is, the proposed resource binding service relies on existing overlay algorithms to retrieve P2P resources from the overlay.

In summary, applications defined on top of RELOAD (e.g., SIP, CoAP) can provide their own services, besides using basic RELOAD services (e.g., DHT for storage, certificate enrollment for security, NAT traversal). The binding service proposed here is an additional service for RELOAD/CoAP architectures, serving as interface between the client and the overlay layer. Such interface has two purposes:

- The redesign of P2P resources before final storage at the overlay, in order to incorporate virtual resource entries;

- The replacement of virtual entries by real device resource entries, before content delivery to the client.

The following section discusses how this service should redesign a set of P2P resources.

3.3 Theoretical Model

Let us assume a set of P2P resources, denoted by $\mathcal{R}$, available in the overlay network. A P2P resource $r \in \mathcal{R}$ includes a set of `(KEY,VALUE)` entries denoted by $\mathcal{P}_r$. Assume also that each $p_r \in \mathcal{P}_r$ includes a set of device resource entries denoted by $\mathcal{E}_{p_r}$. The job of a resource binding server is to replace device resource entries by resource bindings (references to existing P2P resources) whenever possible. The new set of entries for r is denoted by $\mathcal{P}'_r$. After replacements, the P2P resources would be

$$\mathcal{R}_E(\mathcal{R}) = \{(r, \mathcal{P}'_r) | \cup_{p'_r \in \mathcal{P}'_r} \mathcal{E}_{p'_r} = \cup_{p_r \in \mathcal{P}_r} \mathcal{E}_{p_r} \wedge \sum_{p'_r \in \mathcal{P}'_r} |\mathcal{E}_{p'_r}| = min_{p''_r \in \mathcal{P}''_r} \{|\mathcal{E}_{p''_r}|\}. \qquad (3.1)$$

This means that the content of replies to P2P resource requests should not change, after replacing entries by bindings, and that the overall number of device resource entries at the P2P overlay network should be the lowest possible. The number of device resource entries for each r will also be the lowest possible.

The binding replacement model can, however, have another goal: the minimization of the total number of fetches requested by the binding service. For example, if `temperature` P2P resource includes one or more virtual resource entries and no device resources, the `heat-related-illness` resource might reduce the number of fetches if it replaces the `temperature` resource entry by multiple entries to those virtual resources. That is, since a call to `temperature` does not bring device resource entries, and further fetches of virtual resources entries in it are required, then referring directly to the virtual resources minimizes fetches.

Let us assume a dependency graph where each node represents a resource (either a P2P resource or a device resource). A node/resource has as children nodes all resources included in it. The result will be a graph, or set of connected trees. Assume also that each parent node/resource has a weight $w(n) = \sum_{n' \in \Gamma_n} w(n') + 1$, where Γ_n are all children nodes of n. All final device resource nodes have zero weight. For the number of fetches to be minimized, the resources after replacements should be:

$$\begin{aligned} \mathcal{R}_F(\mathcal{R}) = \{(r, \mathcal{P}'_r) | \cup_{p'_r \in \mathcal{P}'_r} \mathcal{E}_{p'_r} = \cup_{p_r \in \mathcal{P}_r} \mathcal{E}_{p_r} \wedge \\ \wedge \sum_{p'_r \in \mathcal{P}'_r} \sum_{e \in \mathcal{E}_{p'_r}} w(e) = min_{p''_r \in \mathcal{P}''_r} \{ \sum_{e \in \mathcal{E}_{p''_r}} w(e)\}. \end{aligned} \qquad (3.2)$$

The $\mathcal{R}_E$ is a particular case of $\mathcal{R}_F$ were all nodes have weight equal to 1. Having these models in mind, a heuristic algorithm is discussed next to determine $\mathcal{R}_F$ (or $\mathcal{R}_E$ if nodes have weight equal to 1).

3.4 Heuristic Algorithm

For a specific P2P resource $r \in \mathcal{R}$, the previously discussed binding replacement model resembles a maximum cover problem, which is known to be NP-hard [12]. More formally, let us assume the universe of device resource entries at a P2P resource r, given by $\mathcal{E}^r = \cup_{p_r \in \mathcal{P}_r} \mathcal{E}_{p_r}$. Assume also $\mathcal{S}^r = \{r' \in \mathcal{R} \backslash \{r\} : \cup_{v_{r'} \in \mathcal{P}_{r'}} \mathcal{E}_{v_{r'}} \subset \mathcal{E}^r\}$ to be the family of P2P resources whose device resource entries form a subset of $\mathcal{E}^r$. In a maximum cover problem the goal is to select at most z of such subsets such that the maximum number of elements in $\mathcal{E}^r$ are covered. In a weighted version of the maximum cover problem, each element of $\mathcal{E}^r$ has a weight. In our P2P context, such problem would be mathematically formulated as follows:

$$\text{maximize} \sum_{\{e \in \mathcal{E}^r\}} w(e) \times y_e \tag{3.3}$$

$$\text{subject to} \sum_{\{r' \in \mathcal{S}^r\}} x_{r'} \leq k \tag{3.4}$$

$$\sum_{\{r' \in \mathcal{S}^r : e \in \cup_{v_{r'} \in \mathcal{P}_{r'}} \mathcal{E}_{v_{r'}}\}} x_{r'} \geq y_e, \forall e \in \mathcal{E}^r \tag{3.5}$$

where y_e states if e is being covered or not, and $x_{r'}$ states if P2P resource r' is selected for the cover, both binary variables. However, in the resource binding replacement problem the elements holding a weight would be the elements in $\mathcal{S}^r$, and not the elements in $\mathcal{E}^r$, if the previously discussed graph is considered. The expression at the objetive function would also change to $\sum_{\{e \in \mathcal{E}^r\}} y_e + \frac{1}{\Delta \times w(r') \times x_{r'}}$, where Δ is a big value. This way the highest covering, with lowest number of fetches would be achieved. Besides this difference, the covering must be performed for multiple P2P resources meaning that the best processing order should be found. Therefore, algorithms in the literature that solve the weighted maximum cover problem can not be applied to the binding replacement problem. These issues are addressed by the proposed heuristic, shown in Algorithm 1.

Algorithm 1: Heuristic for the replacement of resource entries.

```
Create tree graph $\tau$ as defined in Section 3.3
/* each node/resource has a list with replacements */
$\mathcal{L}_r = \{\}, \forall r \in \tau$
/* process resources with lowest weight first */
for $r \in \tau : w(r) \neq 0$ in increasing order of $w(r)$ do
    for $r' \in \tau \backslash \{r\}$ do
        /* sensor entries of $r$ */
        $\mathcal{A} = \cup_{p_r \in \mathcal{P}_r} \mathcal{E}_{p_r}$
        /* sensor entries of $r'$ */
        $\mathcal{B} = \cup_{v_{r'} \in \mathcal{P}_{r'}} \mathcal{E}_{v_{r'}}$
        /* see if replacement is valid and no superset exists in list of
        replacements */
        if $\mathcal{A} \subset \mathcal{B} \wedge \nexists l \in \mathcal{L}_{r'} : \mathcal{A} \subset l$ then
            /* find all replacements subsets of $\mathcal{A}$ */
            $\mathcal{C} = \{l \in \mathcal{L}_{r'} : l \subset \mathcal{A}\}$
            /* if more sensor entries are covered then solution is better,
            otherwise no benefit exists since weight will be higher */
            if $|\mathcal{A}| > | \cup_{c \in \mathcal{C}, p_c \in \mathcal{P}_c} \mathcal{E}_{p_c}|$ then
                /* remove replacements covered by $r$*/
                $\mathcal{L}_{r'} = \mathcal{L}_{r'} \backslash \mathcal{C}$
                /* add $r$ to list of replacements */
                $\mathcal{L}_{r'} \leftarrow r$
            end
        end
    end
end
for $r \in \tau : w(r) \neq 0$ do
    /* perform replacements */
    Replace $r$ entries covered by $\mathcal{L}_r$
    for $l \in \mathcal{L}_r$ do
        /* remove reference of $r$ in $\mathcal{L}_l$; avoids loops */
        $\mathcal{L}_l = \mathcal{L}_l \backslash \{r\}$
    end
end
```

3.5 Analysis of Results

To evaluate the theoretical models previously discussed we assume sensing as a service scenarios (smart city and eHealth) including a multi-supplier

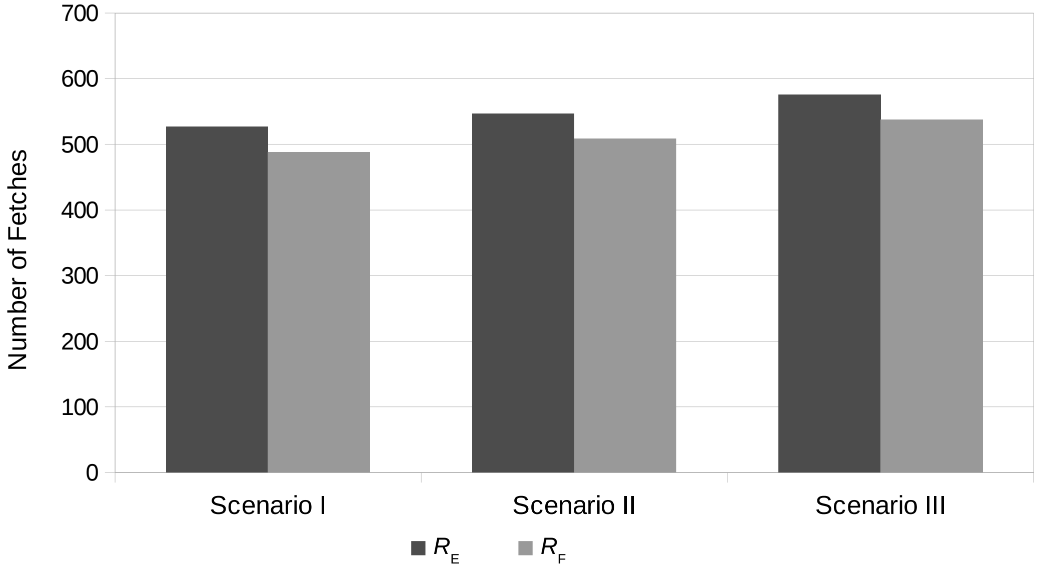


Figure 3.2: Number of fetches for smart city scenario.

deployment of sensors and multi-client access to sensor resources. This way, the benefits of IoT eco-system can reach all users. The smart city scenario is used to analyse the impact on the number of fetches, and number of sensor references at P2P resources, when the amount of available sensors grows. The eHealth scenario is used to analyse the impact on the number of fetches, and number of sensor references at P2P resources, when sensors are inserted at multiple P2P resources (data may have multiple relations). The amount of available sensors remains the same in this case.

3.5.1 Smart City Scenario

City councils are assumed to make street sensors available at the P2P overlay network. Sensors are also announced at the overlay network according to their city area (set of streets). Therefore, a specific device resource entry might be included at street, area and city P2P resources.

Three scenarios are analysed. In Scenario I a random number of sensors per street, ranging from 0 to 5, is defined. In Scenarios II and III these ranges are from 0 to 10 and from 0 to 15, respectively. The P2P overlay network has resources available for streets, areas and cities. A total of 5 cities is assumed, while for the number of streets per area, and number of areas per city, a random number between 5 and 10 is used.

The plot in Figure 3.2 compares the number of fetch operations required at

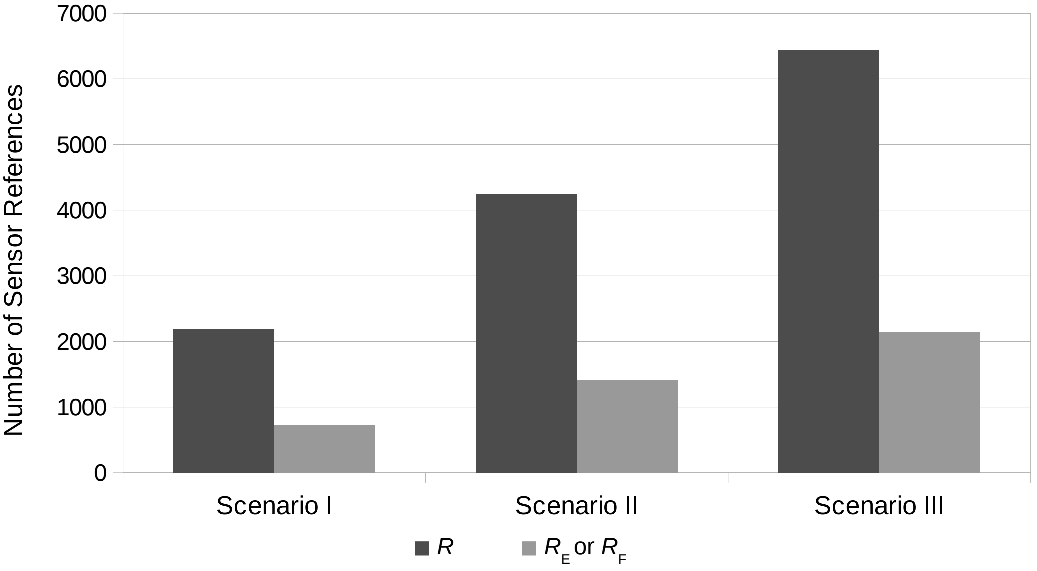


Figure 3.3: Number of sensor references for smart city scenario.

the P2P overlay network for all sensor resource entries to be retrieved. The number of fetches, at the P2P overlay network, when using $\mathcal{R}_E$ and $\mathcal{R}_F$ is shown. From such plot it is possible to state that $\mathcal{R}_F$ reduces the number of fetches required, when compared with $\mathcal{R}_E$, which will have a positive impact on client's experience. The $\frac{\mathcal{R}_F}{\mathcal{R}_E}$ ratio does not change as the number of sensors per street grows.

Regarding the total number of sensor resource entries included in the overall set of P2P resources, both $\mathcal{R}_E$ and $\mathcal{R}_F$ ensure the smallest possible number of entries. That is, a sensor resource entry is not included in more than one P2P resource, resulting in a tree-based organization of resources. The difference between $\mathcal{R}_E$ and $\mathcal{R}_F$ relies on the amount of virtual resource entries (bindings to existing P2P resources) used. The plot in Figure 3.3 shows the total number of sensor resource entries when the binding service is present ($\mathcal{R}_E$ or $\mathcal{R}_F$, as they provide the same values), and when the binding service is not present ($\mathcal{R}$). As can be seen the use of such service significantly reduces the total number of sensor resource entries, turning them unique. This facilitates future changes/updates. The $\frac{\mathcal{R}_E}{\mathcal{R}}$ ratio (or $\frac{\mathcal{R}_F}{\mathcal{R}}$) does not change with the number of sensors, meaning that the impact of using resource bindings is predictable.

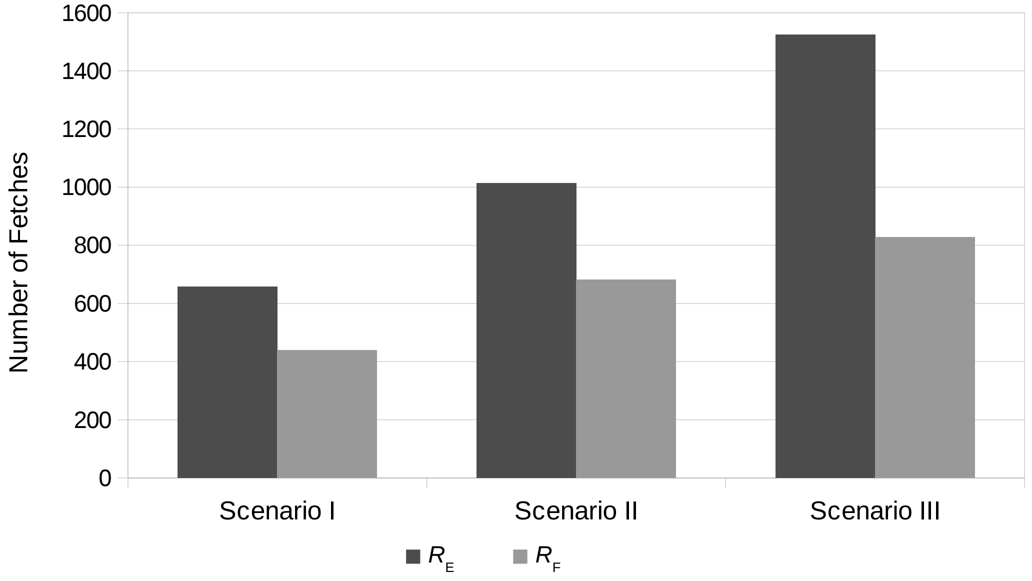


Figure 3.4: Number of fetches for eHealth scenario.

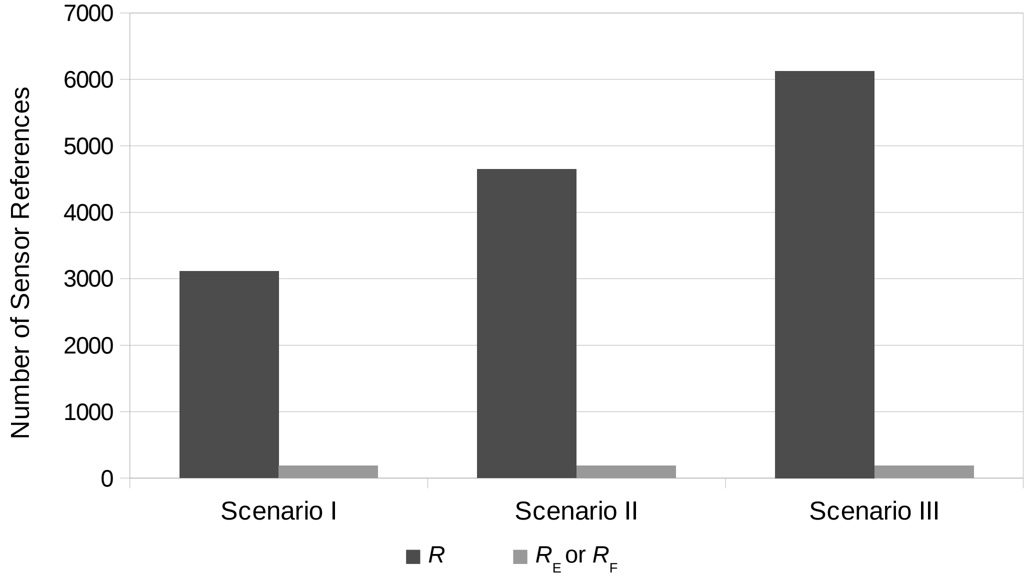


Figure 3.5: Number of sensor references for eHealth scenario.

3.5.2 eHealth Scenario

When monitoring patients, eHealth agents can make sensors/data (e.g., glucometer, blood pressure, etc) available for different kind of medical analysis. There may be interest in not only announcing P2P resources according to sensor type, group of patients under same conditions or critical level, etc, but also in inserting the same set of sensors at multiple P2P resources, as data may have multiple relations.

For simulation of an eHealth scenario, sets including a random number of sensors of the same type are generated. Each of these sets can be included in randomly selected P2P resources, meaning that P2P resources can incorporate sensors of multiple types. Each of these P2P resources can then be included in randomly selected higher level P2P resources. In all analysed scenarios, the overall number of sensor sets, first level P2P resources and second level P2P resources, is 25. The number of sensors per sensor set ranges from 5 to 10. Regarding how sensor sets are included in P2P resources, different scenarios were tested. In Scenario I each set has a 0.2 probability of being included in a P2P resource. For Scenarios II and III these probabilities are 0.4 and 0.6, respectively.

The plot in Figure 3.4 compares the number of fetch operations required at the P2P overlay network for all sensor resource entries to be retrieved. Again, plots show that $\mathcal{R}_F$ reduces the number of fetches required, when compared with $\mathcal{R}_E$, with a positive impact on client's experience. Here, however, benefits are higher than in the smart city case. Note that, contrarily to the previous smart city scenarios, sensor resource entries may be included in more than one P2P resource, resulting into an organization of resources in graph (instead of a tree). The benefit of using $\mathcal{R}_F$ grows as sensor resource entries become part of more P2P resources, as can be seen by the $\frac{\mathcal{R}_F}{\mathcal{R}_E}$ ratio decrease.

The plot in Figure 3.5 shows the total number of sensor resource entries when the binding service is used, $\mathcal{R}_E$ or $\mathcal{R}_F$, and when this service is not used, $\mathcal{R}$. In these eHealth scenarios, the total number of sensor resource entries does not change when $\mathcal{R}_E$ or $\mathcal{R}_F$ is applied because sensors are the same for Scenario I, II and III, as previously stated. As sensors are inserted in more P2P resources (relations among data increases), the number of sensor references increases for $\mathcal{R}$. This means that $\frac{\mathcal{R}_E}{\mathcal{R}}$ ratio (or $\frac{\mathcal{R}_F}{\mathcal{R}}$) decreases, allowing us to conclude that the benefits of using resource bindings is higher when there are more relations among data.

3.6 Summary

This chapter presents a resource binding service so that P2P resources can include other P2P resources available at the overlay network. That is, bindings are built having just existing P2P resources in consideration. Two models are proposed for the binding of resources, together with an algorithm for their implementation. Results show that both models significantly reduce the total number of sensor resource entries, facilitating future changes/updates. One of the models is also able to reduce the number of fetches, required at the P2P overlay network, for the retrieval of all sensor resource entries, improving the quality of client's experience. This improvement on the number of fetches increases as sensor resources become part of more P2P resources (more relations among data).

CHAPTER 4

P2P Resource Redesign

4.1 Introduction

CoAP Usage allows device resources managed by different proxies to be announced under a unique P2P resource umbrella. This can be used to group resources of the same type, or with similar characteristics, although originating from different constrained environments. For example, `h("coap://overlay-1.com/temperature/.well-known/")` can include a set of references to temperature sensors available at different nodes/environments, and `h("coap://overlay-1.com/temperature/")` would be used to fetch all related sensor data.

By the above, it is clear that P2P resources may end up including similar device resource references, or similar sensor data in the caching case. This poses consistency and efficiency problems to such distributed storage systems. More specifically, whenever a device resource reference or sensor value changes, multiple P2P resources must be updated. This becomes critical as more and more objects integrate the IoT (possible combinations of resources, for announcement purposes, will increase exponentially). Although a binding to an existing P2P resource may solve the problem some times (case considered in the previous chapter), such approach only works if P2P resources are already available, which does not always happen. That is, there may be an overlapping of entries but that overlapping part may not exist as a standalone P2P resource. Therefore, the following will be required:

- Allow bindings as P2P resource entries: If bindings to other existing P2P resource are inserted as entries in a P2P resource, then similar device resource entries in multiple P2P resources can be avoided. Any update at P2P resources would automatically have effect on the P2P resources including it (as a binding), avoiding updates to multiple P2P resources. An overlay service will be required, as explained latter.

- Create anonymous P2P resources: Advertisers of new P2P resources may wish to include part of the content of existing P2P resources. In this case, the overlay service must: *i*) detect common device resource entries (from multiple P2P resources); *ii*) place common entries inside anonymous P2P resources; *iii*) create the necessary bindings. The same can be done for P2P resources with sensor data, in order to avoid similar data elements at multiple P2P resources. This requires planning for the right set of anonymous P2P resources and bindings.

In summary, the adoption of CoAP Usage is not enough to deal with the resource consistency problem that may arise in the future. For this reason, a multi-layer fetching approach is proposed. More specifically, anonymous P2P resources are created, and content of announced P2P resources is changed in order to point to such anonymous resources leading to device resource entries of interest. This approach allows for a more efficient retrieval/storage of IoT data, while keeping information at the overlay network consistent. A mathematical optimization model is proposed for the planning of anonymous P2P resources and bindings, together with a heuristic algorithm. The following sections also discuss the required overlay service operation for this multi-layer fetching approach to work.

This chapter is organized as follows. Section 4.2 presents the binding service operating at the RELOAD/CoAP overlay network, now considering the redesign of existing resources. Section 4.3, after introducing some required definitions and assumptions, introduces the resource redesign problem and formalizes it mathematically. Section 4.4 presents an algorithmic approach to solve the resource redesign problem. In Section 4.5 an analysis and discussion of results is presented, and finally Section 4.6 presents a summary of the chapter.

Contributions

The contributions presented in this chapter are the following:

- Development of a mathematical optimization model for the planning of anonymous P2P resources and bindings.

- Development of a heuristic algorithm for solutions to be found faster.

These contributions were published in:

- L. Rodrigues, J. Guerreiro and N. Correia, "Resource Design in Federated Sensor Networks using RELOAD/CoAP Overlay Architectures", submitted to Internet of Things, Elsevier.

4.2 Resource Redesign and Binding Service Operation

4.2.1 Need for P2P Resource Redesign

One of the purposes of federating sensor networks is for data from different networks to be crossed, and for multiple clients to access multiple data. Therefore, different combinations of device resources can be made available under different P2P resource umbrellas. For example, the following `heat-related-illness` P2P resource, sharing entries with the `temperature` P2P resource previously defined, can be of interest in eHealth:

```
Resource-ID = h(coap://overlay-1.com/heat-related-illness/
                                          .well-known/core)

KEY = 9996172,
VALUE = [
 </sensors/temp-1>;rt="temperature";if="sensor",
 </sensors/temp-2>;rt="temperature";if="sensor",
 </sensors/co2-1>;rt="co2";if="sensor",
 </sensors/co2-2>;rt="co2";if="sensor"
]

KEY = 9996173,
VALUE = [
 </sensors/temp-a>;rt="temperature";if="sensor",
 </sensors/temp-b>;rt="temperature";if="sensor"
]
```

The proliferation of similar entries at multiple P2P resources should, however, be avoided. Although a binding to an existing P2P resource may solve the problem in this case, such approach only works if P2P resources are already available, which does not always happen. That is, there may be overlapping of entries but that overlapping part may not exist as a stand-alone P2P resource. To overcome this problem, a multi-layer fetching approach is proposed. More specifically, anonymous P2P resources must be created, and content of existing P2P resources must be changed in order to point to such anonymous resources that lead to device resource entries of interest. Thus, two kinds of content for P2P resources are proposed:

- *Anonymous*: Includes references to other P2P resources, each following the CoRE Link Format.
- *KeyValue*: Includes `(KEY,VALUE)` entries following the CoAP Usage format.

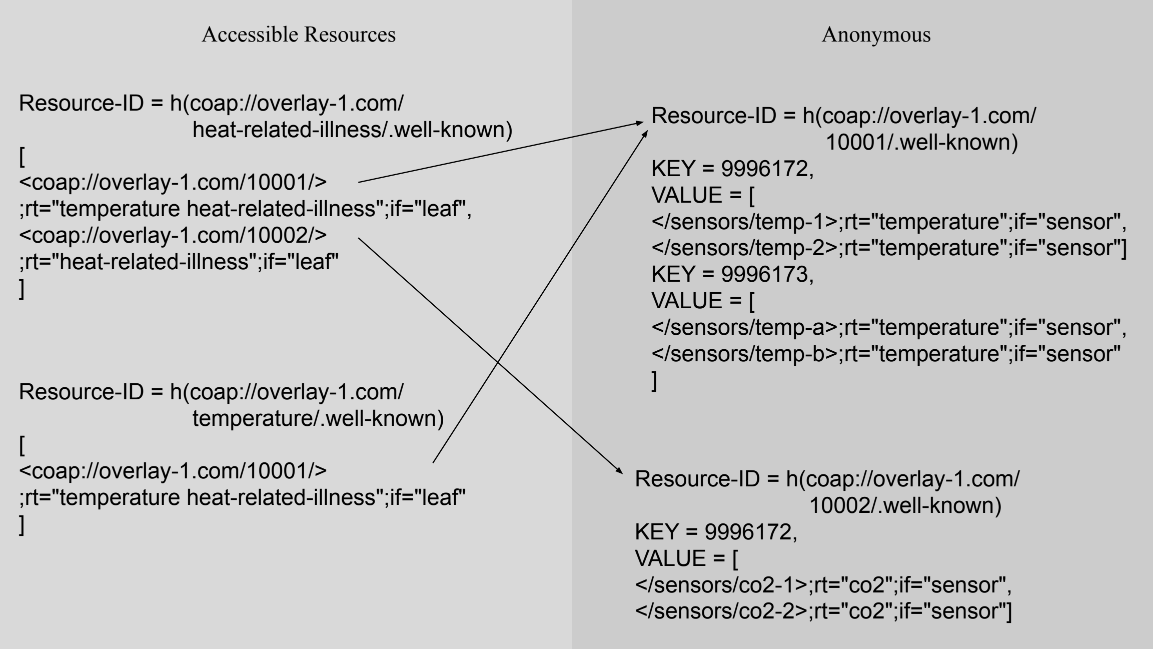
That is, the content format of P2P resources can be of one of these types. For the previous example, the content of `heat-related-illness` and `temperature` P2P resources would have the following format of Anonymous kind:

```
Resource-ID = h(coap://overlay-1.com/heat-related-illness/
                                                    .well-known)
[
    <coap://overlay-1.com/10001/>;rt="temperature
                    heat-related-illness";if="leaf",
    <coap://overlay-1.com/10002/>;rt="heat-related-il
                                        lness";if="leaf"
}
```

```
Resource-ID = h(coap://overlay-1.com/temperature/.well-
                                                     known)
[
    <coap://overlay-1.com/10001/>;rt="temperature
                     heat-related-illness";if="leaf"
]
```

While these have an Anonymous content format, the content of the new anonymous P2P resources `coap://overlay-1.com/10001/` and `coap://overlay-1.com/10002/` would follow the CoAP Usage format, storing `(KEY,VALUE)` entries with temperature and CO2 device resource links, respectively. Note that the `if` specifies how to deal with the endpoint, and “leaf” means that the content of such endpoints follows the KeyValue format (otherwise the `if="anonymous"` is used). Basically, a graph exists where P2P resources (anonymous or not) are nodes and references are links, and multiple hops exist until `(KEY,VALUE)` entries are reached. The `rt` indicates resource types for such entry, which end up being the parent P2P resources at the graph. This is illustrated in Figure 4.1. An overlay service would ensure that the client, performing a fetch to a P2P resource, receives just the `(KEY,VALUE)` entries, not being aware of existing anonymous P2P resources.

The previously discussed multi-layer fetching approach is able to avoid duplicate entries under any circumstances. Another advantage is that the task



```
Resource-ID = h(coap://overlay-1.com/
                  heat-related-illness/.well-known)
[
<coap://overlay-1.com/10001/>
;rt="temperature heat-related-illness";if="leaf",
<coap://overlay-1.com/10002/>
;rt="heat-related-illness";if="leaf"
]

Resource-ID = h(coap://overlay-1.com/
                  temperature/.well-known)
[
<coap://overlay-1.com/10001/>
;rt="temperature heat-related-illness";if="leaf"
]
```

```
Resource-ID = h(coap://overlay-1.com/
                  10001/.well-known)
KEY = 9996172,
VALUE = [
</sensors/temp-1>;rt="temperature";if="sensor",
</sensors/temp-2>;rt="temperature";if="sensor"]
KEY = 9996173,
VALUE = [
</sensors/temp-a>;rt="temperature";if="sensor",
</sensors/temp-b>;rt="temperature";if="sensor"
]

Resource-ID = h(coap://overlay-1.com/
                  10002/.well-known)
KEY = 9996172,
VALUE = [
</sensors/co2-1>;rt="co2";if="sensor",
</sensors/co2-2>;rt="co2";if="sensor"]
```

Figure 4.1: Illustration of overlay fetching graph.

of extracting anonymous P2P resource content can be distributed by several nodes implementing the service, as described next. The right set of anonymous P2P resources, their content, and redesign of P2P resources must, however, be planned. This planning problem is addressed in Section 4.3.

4.2.2 Binding Service Operation

The requirements for the overlay service include:

1. Upon client requests, P2P resource content must be extracted and delivered to the requesting node.

2. In case of content update (changes of device resource entries at P2P resources) this service should activate procedures to ensure keeping the right design of resources.

Regarding the mechanism used to discover nodes providing such service, a naive solution would be to store at the overlay, under some key k, Node-IDs of nodes providing the service. This, however, will overload the node responsible for the service identified by key k. Such node not only may end up storing a large number of Node-IDs but also must answer all service lookup requests. The ReDiR-based service discovery mechanism, discussed in [48], avoids this ensuring that the load associated with a certain service is distributed among the nodes providing the service. For this reason ReDiR is proposed, in [35],

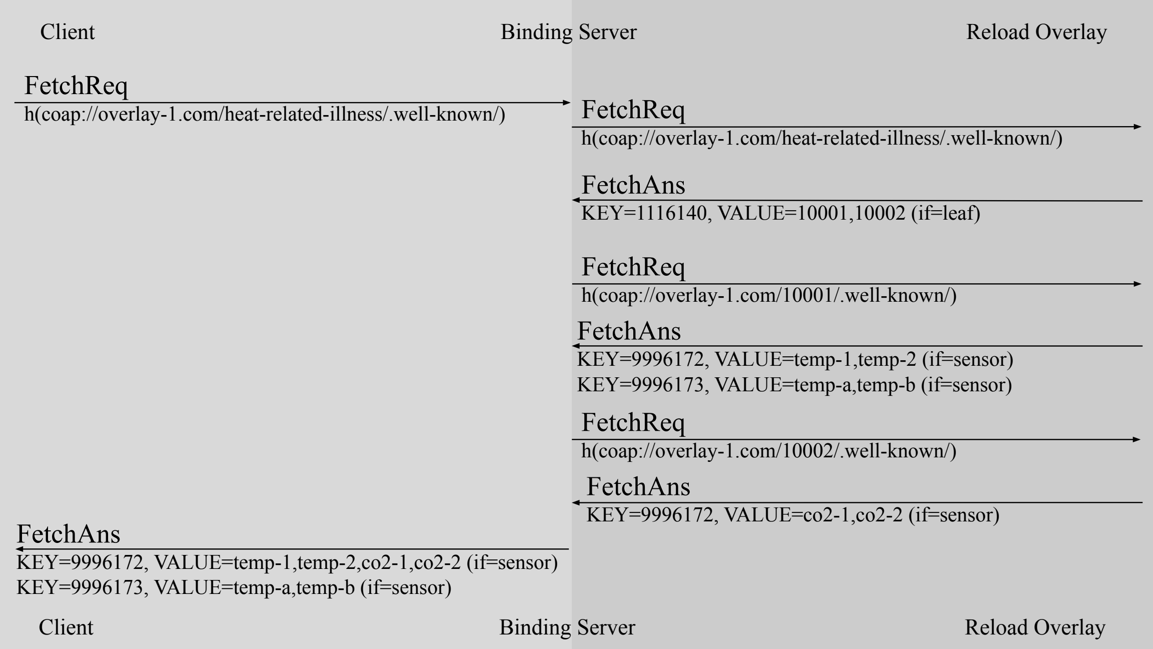

Client
Binding Server
Reload Overlay
FetchReq
h(coap://overlay-1.com/heat-related-illness/.well-known/)
FetchReq
h(coap://overlay-1.com/heat-related-illness/.well-known/)
FetchAns
KEY=1116140, VALUE=10001,10002 (if=leaf)
FetchReq
h(coap://overlay-1.com/10001/.well-known/)
FetchAns
KEY=9996172, VALUE=temp-1,temp-2 (if=sensor)
KEY=9996173, VALUE=temp-a,temp-b (if=sensor)
FetchReq
h(coap://overlay-1.com/10002/.well-known/)
FetchAns
KEY=9996172, VALUE=co2-1,co2-2 (if=sensor)
FetchAns
KEY=9996172, VALUE=temp-1,temp-2,co2-1,co2-2 (if=sensor)
KEY=9996173, VALUE=temp-a,temp-b (if=sensor)
Client
Binding Server
Reload Overlay

Figure 4.2: Illustration of overlay binding service operation.

as a generic service discovery mechanism for RELOAD overlays, ensuring scalability. Figure 4.2 illustrates the overlay service operation.

4.3 P2P Resource Redesign Problem

Multi-layer fetching in RELOAD/CoAP architectures requires the redesign of existing P2P resources, so that no duplicate device resource entries exist. Such resource redesign problem is addressed here in this section. From that point on it is possible to guarantee the non-duplication of device resource entries using procedures that deal with the dynamic arrival of new P2P resources.

4.3.1 Necessary Definitions

Definition 1 (P2P Resource). *An object, to be stored at the P2P overlay network, that can have either: i) entries, following the CoRE Link Format, that point to P2P resources; or ii) `(KEY, VALUE)` records containing entries, following the CoRE Link Format, pointing to physical device resources. That is, the content of a P2P resource can be of Anonymous or KeyValue type.*

In other words, if a P2P resource is of Anonymous type then entries are used to reach other P2P resources. If it is of KeyValue type, an entry inside a `(KEY,VALUE)` record refers to a physical device resource, under the responsability of a RELOAD/CoAP proxy node. A client may interact with such endpoint using interaction methods (e.g., GET).

Definition 2 (Anonymous P2P Resource). *A P2P resource that is created, and made available at the P2P overlay, for the purpose of avoiding similar entries inside `(KEY, VALUE)` structures. Its content can be of Anonymous or KeyValue type.*

Definition 3 (P2P-RR Problem). *Given an original set of P2P resources, $\mathcal{R}$, each $r \in \mathcal{R}$ including a set of `(KEY, VALUE)` records denoted by $\mathcal{P}_r$ and each $p_r \in \mathcal{P}_r$ including a set of device resource entries denoted by $\mathcal{E}_{p_r}$, redesign these P2P resources so that a device resource is not referenced at multiple P2P resources. Such redesign should be done while minimizing the overall number of anonymous P2P resources, as this will minimize the number of required fetches.*

The overall set of device resource entries, from all P2P resources, will be denoted by $\mathcal{E}$. A P2P resource may see its content changed from KeyValue to

Anonymous type, and anonymous P2P resources can be created to solve the P2P-RR problem. The set of anonymous P2P resources is denoted by $\mathcal{A}$, while $\bar{\mathcal{R}}$ is used to denote the redesign of $\mathcal{R}$.

4.3.2 Assumptions on Resources

Assumption 1 (Number of Entries)**.** *A device resource entry can not be included is more than one P2P resource (anonymous or not). That is, for any* $e \in \mathcal{E}$*,* $\kappa(e) = |\{u \in \bar{\mathcal{R}} \cup \mathcal{A} : e = e_{p_u}, e_{p_u} \in \mathcal{E}_{p_u}, p_u \in \mathcal{P}_u\}| = 1$*. Multiple entries pointing to an anonymous P2P resource may exist.*

Assumption 2 (Strict Coverage)**.** *P2P resource redesign must ensure that the information returned to the client (resulting from fetches to anonymous P2P resources) is not different from the original one.*

4.3.3 Mathematical Formalization of P2P-RR Problem

A mathematical model is proposed next to plan for the multi-layer organization of P2P resources.

Notation

The information given as input to P2P-RR problem is summarized as follows:

$\mathcal{R}$ Set of P2P resources to be redesigned, each denoted by $r \in \mathcal{R}$.

$\mathcal{P}_r$ Set of `(KEY,VALUE)` entries at $r \in \mathcal{R}$, each entry denoted by $p_r \in \mathcal{P}_r$.

$\mathcal{E}_{p_r}$ Set of device resource entries included in entry $p_r \in \mathcal{P}_r$ of P2P resource $r \in \mathcal{R}$, each device resource denoted by $e_{p_r} \in \mathcal{E}_{p_r}$.

$\mathcal{E}$ Overall set of device resource entries, from all P2P resources.

$\mathcal{A}$ Overall set of anonymous P2P resources that may be created, each denoted by $a \in \mathcal{A}$.

Regarding variables required to plan for the redesign of P2P resources, these are the following:

τ_a One if anonymous P2P resource $a \in \mathcal{A}$ is in use, zero otherwise.

β_e^u One if P2P resource $u \in \bar{\mathcal{R}} \cup \mathcal{A}$ includes device resource $e \in \mathcal{E}$ as an entry, zero otherwise.

$\alpha_e^{u,u'}$ One if P2P resource $u \in \bar{\mathcal{R}} \cup \mathcal{A}$ receives $e \in \mathcal{E}$ through a reference to P2P resource $u' \in \bar{\mathcal{R}} \cup \mathcal{A} \backslash \{u\}$ (u includes binding to u'), zero otherwise;

γ_e^u One if P2P resource $u \in \bar{\mathcal{R}} \cup \mathcal{A}$ is a provider of $e \in \mathcal{E}$ to others, zero otherwise;

$\xi^{u,u'}$ One if P2P resource $u \in \bar{\mathcal{R}} \cup \mathcal{A}$ is fed by P2P resource $u' \in \bar{\mathcal{R}} \cup \mathcal{A} \backslash \{u\}$, zero otherwise;

Objective and Constraints

A strict minimization of the number of anonymous P2P resources can be achieved by the following objective function, OF:

$$\text{OF: Minimize} \sum_{a \in \mathcal{A}} \tau_a. \tag{4.1}$$

The following constraints must be accomplished:

– Device resource ownership:

$$\sum_{u \in \bar{\mathcal{R}} \cup \mathcal{A}} \beta_e^u = 1, \forall e \in \mathcal{E}. \tag{4.2}$$

These constraints avoid device resource entries at multiple P2P resources.

– Filling P2P resources:

$$\sum_{u \in \bar{\mathcal{R}} \cup \mathcal{A} \backslash \{r\}} \alpha_e^{r,u} + \beta_e^r = \begin{cases} 1, & \text{if } e \in \mathcal{E}_{p_r}, p_r \in \mathcal{P}_r \\ 0, & \text{otherwise} \end{cases}, \forall r \in \bar{\mathcal{R}}, \forall e \in \mathcal{E}, \tag{4.3}$$

$$\sum_{u'' \in \{\bar{\mathcal{R}} \cup \mathcal{A}\} \backslash \{u,u'\}} \alpha_e^{u',u''} + \beta_e^{u'} \geq \alpha_e^{u,u'}, \forall u, u' \in \bar{\mathcal{R}} \cup \mathcal{A} : u \neq u', \forall e \in \mathcal{E}. \tag{4.4}$$

Constraints (4.3) ensure that for some required device resource, redesigned P2P resources will either include an entry or a binding to another P2P resource able to provide it. The constraints also ensure strict coverage.

Constraints (4.4) state that if P2P resource u receives e through a binding to u', then the second must either include a reference to such device resource or a binding to a virtual resource providing it. This is basically a flow conservation law.

– Similar feeding:

$$\gamma_e^u \geq \alpha_e^{u',u}, \forall u, u' \in \bar{\mathcal{R}} \cup \mathcal{A}, \forall e \in \mathcal{E}, \tag{4.5}$$

$$\gamma_e^u \geq \beta_e^u, \forall u \in \bar{\mathcal{R}} \cup \mathcal{A}, \forall e \in \mathcal{E}, \tag{4.6}$$

$$\xi^{u,u'} \geq \alpha_e^{u,u'}, \forall u, u' \in \bar{\mathcal{R}} \cup \mathcal{A}, \forall e \in \mathcal{E}, \tag{4.7}$$

$$\alpha_e^{u,u'} \geq \gamma_e^{u'} + \xi^{u,u'} - 1, \forall u, u' \in \bar{\mathcal{R}} \cup \mathcal{A} : u \neq u', \forall e \in \mathcal{E}. \tag{4.8}$$

$$\alpha_e^{u'',u} \geq \alpha_e^{u,u'} + \xi^{u'',u} - 1, \forall u, u', u'' \in \bar{\mathcal{R}} \cup \mathcal{A}, \forall e \in \mathcal{E}. \tag{4.9}$$

These constraints ensure that a P2P resource provides all its resources to all P2P resources having a binding to it (similar universal feeding).

– Using just anonymous resources to be created:

$$\sum_{u \in \{\bar{\mathcal{R}} \cup \mathcal{A}\} \backslash a} \sum_{e \in \mathcal{E}} (\alpha_e^{u,a} + \alpha_e^{a,u}) \leq \tau_a \times \Delta, \forall a \in \mathcal{A}, \tag{4.10}$$

$$\sum_{u \in \{\bar{\mathcal{R}} \cup \mathcal{A}\} \backslash a} (\xi^{u,a} + \xi^{a,u}) \leq \tau_a \times \Delta, \forall a \in \mathcal{A}, \tag{4.11}$$

$$\sum_{e \in \mathcal{E}} (\beta_e^a + \gamma_e^a) \leq \tau_a \times \Delta, \forall a \in \mathcal{A}, \tag{4.12}$$

where Δ is a big value. These constraints state that α, β, ξ and γ variables of unused anonymous resources can not be filled.

– Single content type (Anonynoums or KeyValue):

$$\sum_{e\in\mathcal{E}} \beta_e^u \leq (1-\xi^{u,u'}) \times \Delta, \forall u,u' \in \bar{\mathcal{R}} \cup \mathcal{A} : u \neq u' \tag{4.13}$$

These ensure that an Anonymous content type will not have device resource entries.

– Non-negativity assignment to variables:

$$\tau_a, \alpha_e^{u,u'}, \beta_e^u, \gamma_e^u, \xi^{u,u'} \in [0,1]. \tag{4.14}$$

The CPLEX[1] optimizer was used to solve this problem, and the solution found will be the optimal solution for the P2P-RR problem under consideration. However, this is an *Integer Linear Programming* (ILP) problem and these are generally NP-hard. Therefore, CPLEX will take a long time for large instances of the problem. For this reason a heuristic algorithm is proposed in the following section.

[1] IBM ILOG CPLEX Optimizer.

Table 4.1: Adopted parameter values.

Parameter	Value
Avg number of P2P resources	5, 10 or 15
Sensor pool size factor, k	1, 2 or 3
Upper bound on P2P resource sensor entries, M	$0.25 \times \lvert\mathcal{E}\rvert$ or $0.5 \times \lvert\mathcal{E}\rvert$

4.4 Algorithmic Approach

4.4.1 Assumptions

Let us assume a compatibility graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ where $\mathcal{N}$ denotes P2P resources and $l = (n, n') \in \mathcal{L}$ iff $\mathcal{E}(n) \cap \mathcal{E}(n') \neq \emptyset$, where $\mathcal{E}(n) = \bigcup\limits_{p_u \in \mathcal{P}_u} \mathcal{E}_{p_u}$ and $\mathcal{E}(n') = \bigcup\limits_{p_{u'} \in \mathcal{P}_{u'}} \mathcal{E}_{p_{u'}}$.

Let $\mathcal{N}_n$ denote the neighbours of node $n \in \mathcal{N}$, $\mathcal{N}_n = \{n' \in \mathcal{N} : (n, n') \in \mathcal{L}\}$. Given a clique $\mathcal{C} \subset \mathcal{G}$, let us define $\mathcal{X}(\mathcal{C})$ as the set of nodes that can expand the clique using, if necessary, a swap operation. That is, $\mathcal{X}(\mathcal{C}) = \{n \in \mathcal{N} : |\mathcal{C} \backslash \mathcal{N}_n| \in \{0, 1\}\}$, where $|\mathcal{C} \backslash \mathcal{N}_n| = 0$ means that the clique is expanded by direct inclusion of node n, and $|\mathcal{C} \backslash \mathcal{N}_n| = 1$ means that the clique must perform first a swap of one of its nodes (the one not connected to n) with n.

The overall ideia of the proposed heuristic is to expand, at every round, a clique initially including just the $l = (n, n')$ with highest weight, using the neighbouring approach previously mentioned. The weight of an l is given by $|\mathcal{E}(n) \cap \mathcal{E}(n')|$, which is basically the number of device resource entries that P2P resources n and n' have in common. This allows the highest weight clique to be found, for an anonymous P2P resource to be built. The pseudocode of the heuristic algorithm is shown in Algorithm 2. Clique expansion is also applied in [28] but the weighted graph and claims for direct inclusion and swaps are different, as the context of application is different.

4.4.2 Algorithm Details

The heuristic approach in Algorithm 2 tries to increase the size of cliques around compatibility graph links with highest weight, meaning that the largest number of P2P resources with similar device resource entries is being searched. Swap operations can be performed, if productive. Therefore, although this is a greedy approach, a non-myopic policy is being used, which can greatly improve the success of the algorithm. Algorithm 2 includes two steps:

1. An initialization step, at Lines 1-8, that basically creates the compatib-

ility graph;

2. A resource redesign step, at Lines 9-41, that tries to expand cliques. The initial clique includes just the highest weight link (see Lines 10-12) and then all possible expansions of the clique are stored until no more expansion is possible (Lines 13-39). Lines 16-20 evaluate expansions with no swaps, Lines 21-32 evaluate expansions involving a swap operation, and Lines 34-38 choose the best expansion of the current clique. When no more clique expansions can not performed, an anonymous P2P resource is built with the entries that are common to the elements (P2P resources) of que clique.

4.5 Analysis of Results

4.5.1 Simulation Setup

To evaluate the impact of redesigning original P2P resources, replacing them by KeyValue and Anonymous P2P resources, random scenarios were generated based on a pool of sensors that populate the `(KEY,VALUE)` entries of the original set of P2P resources, $\mathcal{R}$. Both the proposed mathematical optimization model and the heuristic approach are evaluated. These random scenarios were created as follows:

- Scenarios A, B, C and D have on average 5, 10, 15 and 20 original P2P resources, respectively;
- The size of the pool of sensors, $|\mathcal{E}|$, is $No.\ P2P\ resources \times k$, where k is a pool size increase factor.
- A P2P resource is populated with m sensor entries, where $1 \leq m \leq M$. Two cases were considered for M: *i)* $0.25 \times |\mathcal{E}|$, called sparse case; *ii)* $0.5 \times |\mathcal{E}|$, called dense case. The m value is randomly generated using a uniform distribution, and the m sensor entries are adjacent elements picked from the pool of sensors, starting from a randomly generated point of the pool.

Table 4.1 summarizes the adopted parameter values.

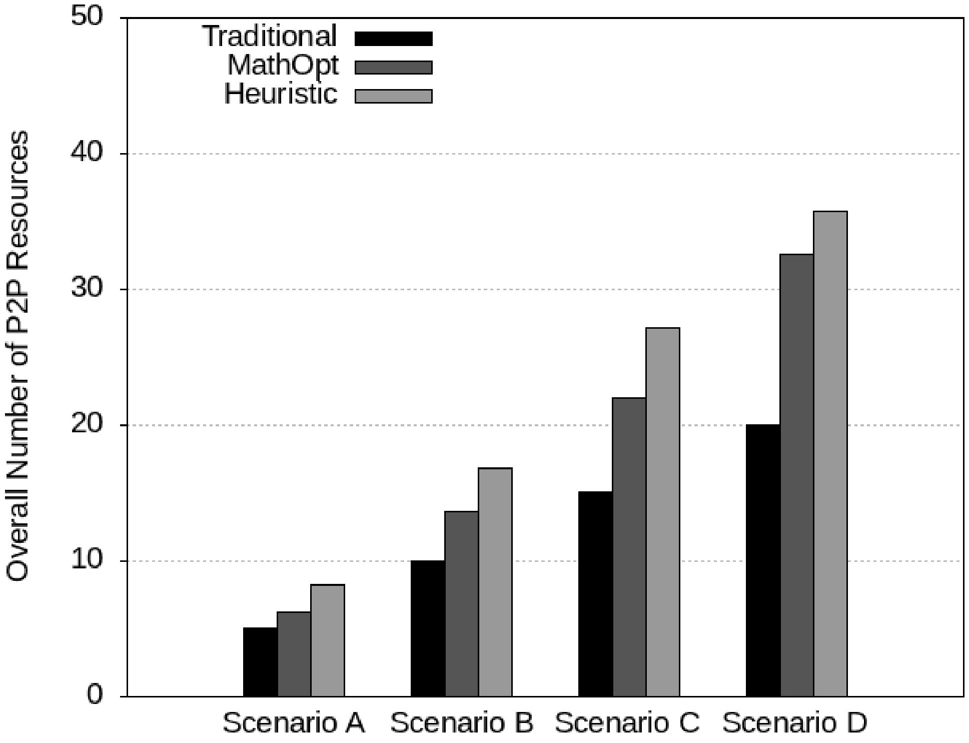
4.5.2 Evaluation

The following plots show the average of 5 tests performed for each scenario/case.

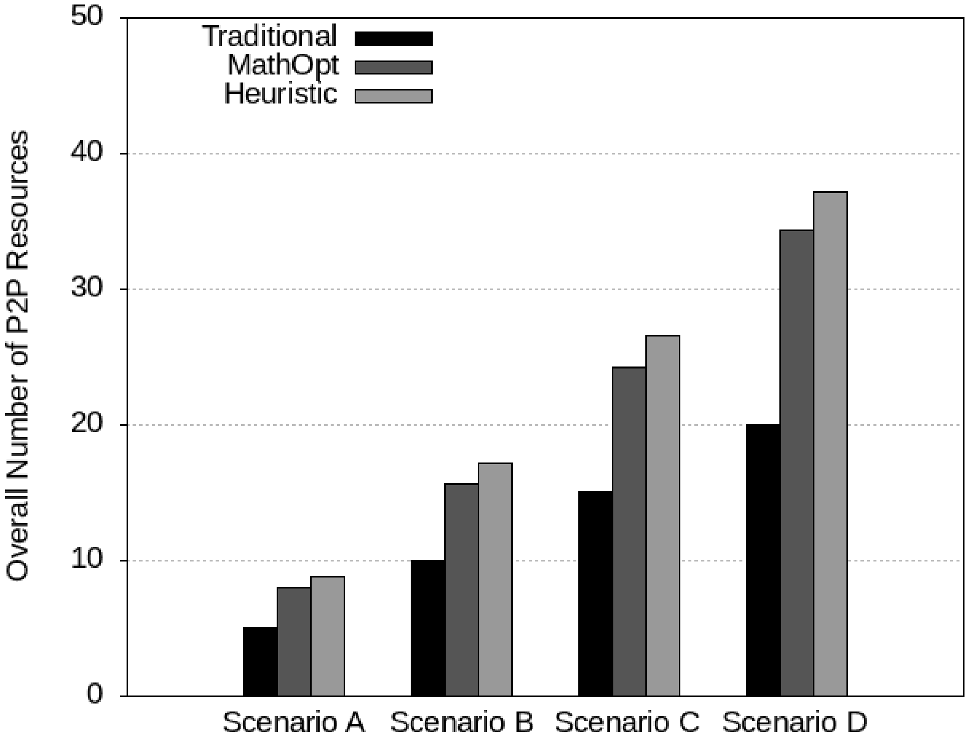
P2P Resources

The plots in Figures 4.3-4.5 show the overall number of P2P resources resulting from the redesign of original P2P resources, for the multi-layer fetching approach to be implemented. Both the sparse and dense scenarios, for different sensor pool sizes, are shown.

Results show that as k (sensor pool size factor) increases, which means increasing the probability of P2P resources having different device resource entries, the ratio between the number of P2P resources generated by the multi-layer approach and the traditional one increases due to an increase in the number of anonymous P2P resources, as expected. This increase is

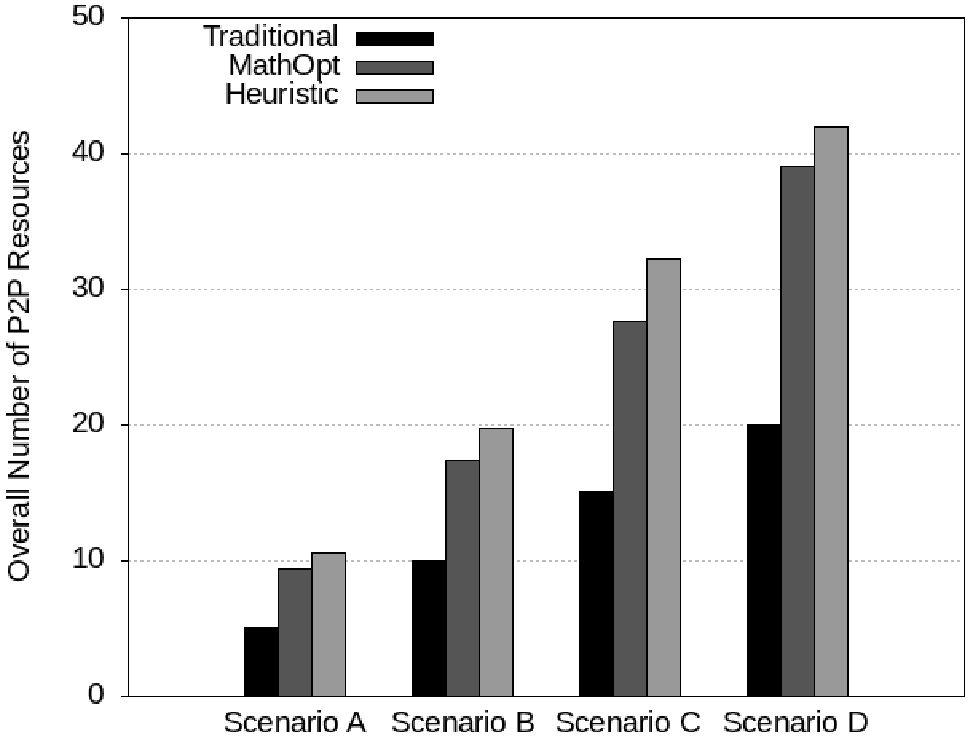


(a) Sensor pool size factor, sparse case.

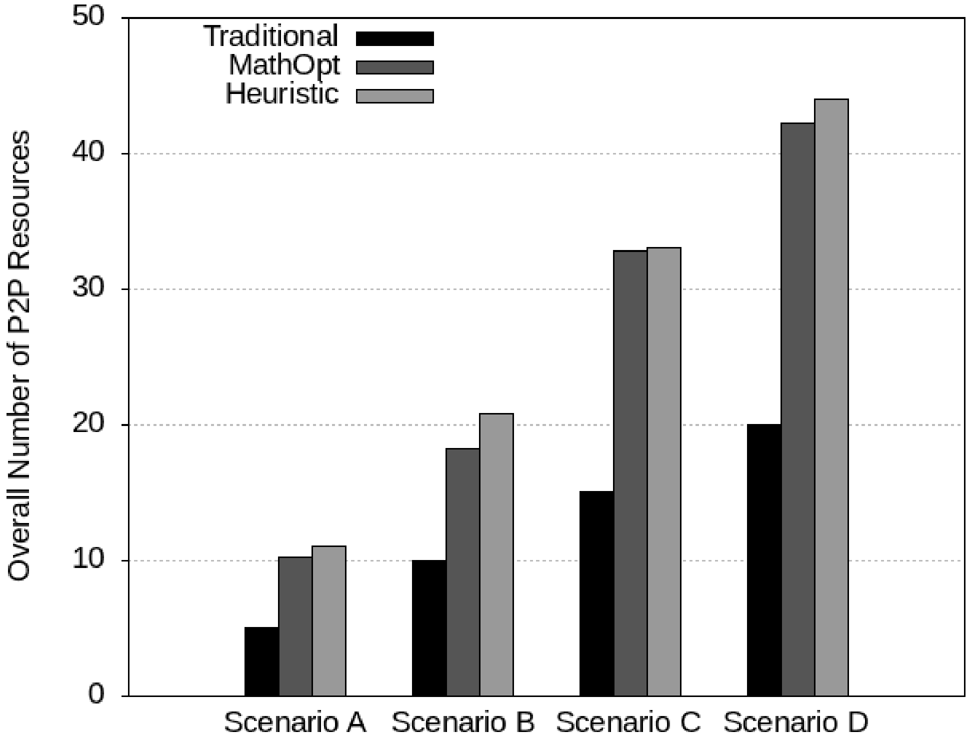


(b) Sensor pool size factor, dense case.

Figure 4.3: Average number of P2P resources for the traditional, heuristic and optimal approaches, $k = 1$.

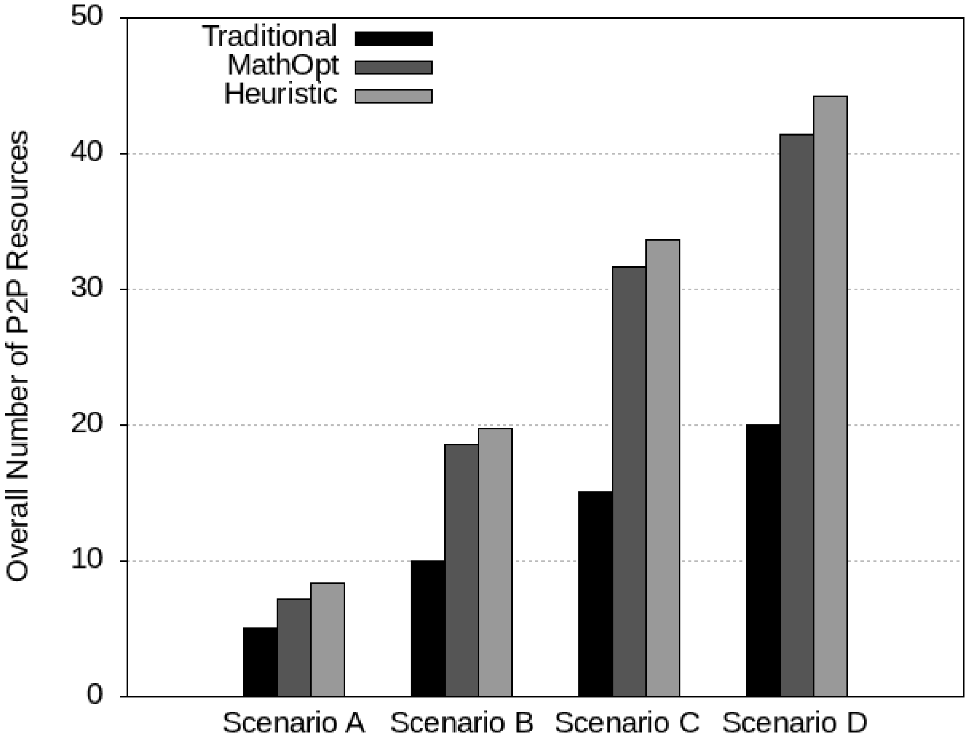


(a) Sensor pool size factor, sparse case.

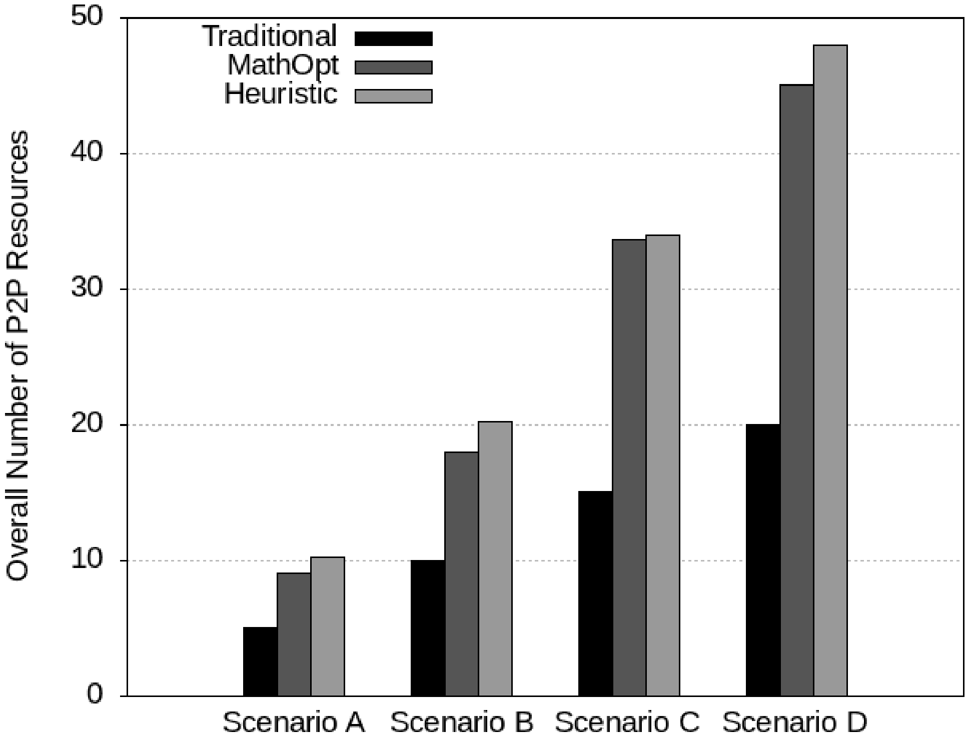


(b) Sensor pool size factor, dense case.

Figure 4.4: Average number of P2P resources for the traditional, heuristic and optimal approaches, $k = 2$.



(a) Sensor pool size factor, sparse case.



(b) Sensor pool size factor, dense case.

Figure 4.5: Average number of P2P resources for the traditional, heuristic and optimal approaches, $k = 3$.

slightly higher in the dense scenario because the population of entries per P2P resource is higher. Despite this increase, the heuristic approach is able to keep results very close to the optimal. High quality solutions can be obtained meaning that the heuristic is scalable from this point of view. This happens for both sparse and dense scenarios and different k factors, so it seems that the performance of the heuristic is invariable for different populations of sensors in pools and P2P resources.

Fetches

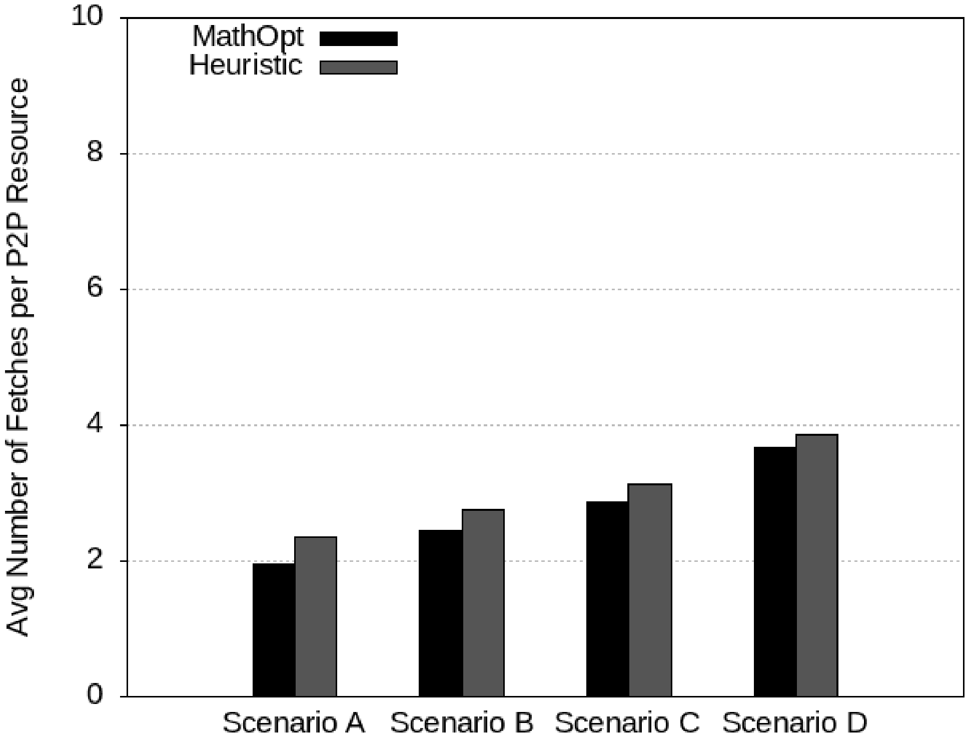
The plots in Figure 4.6-4.8 show the average number of fetches per P2P resource for the optimal and heuristic approaches applied to the proposed multilayer scheme. The traditional one would get the P2P resource content using a single fetch operation because no anonymous P2P resources need to be retrieved, to build the content of original P2P resources. Both the sparse and dense scenarios, for different sensor pool sizes, are shown.

Results confirm that the heuristic approach is able to keep the number of fetches very close to the optimal, which of course is related with its performance in terms of anonymous P2P resources created. The number of fetches increases as k (sensor pool size factor) increases, as expected, and this is more accentuated in dense scenarios. There are oscilations in the heuristic vs optimal number of fetches but this is related with a greater or lesser retrieval need of anonymous P2P resources, by non anonymous (original) P2P resources.

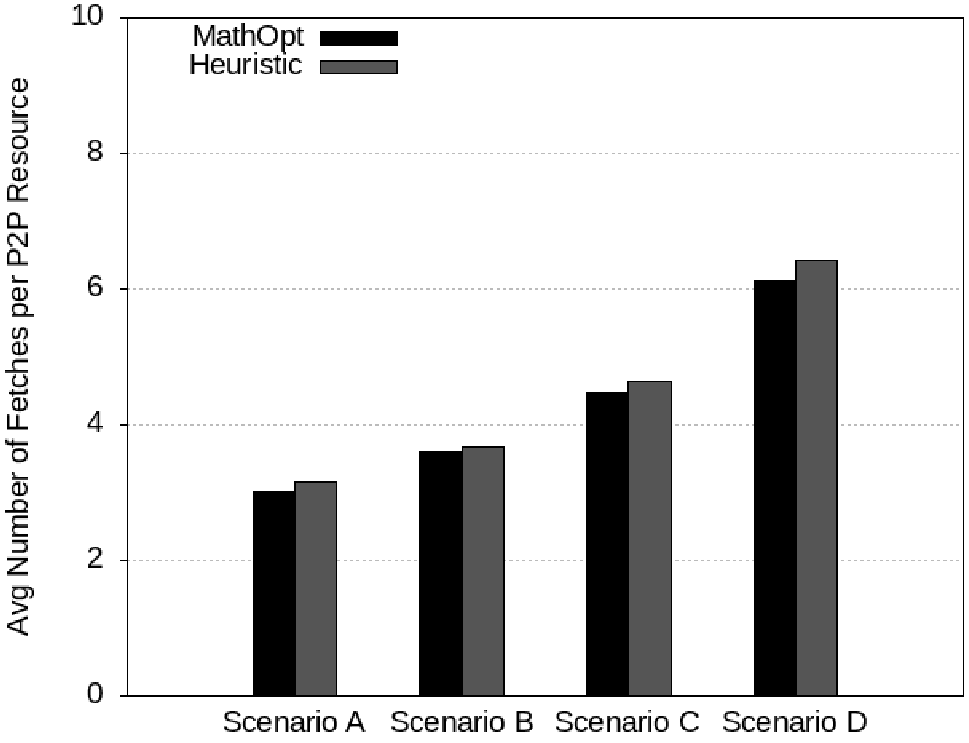
There is a slight difference between sparse and dense scenarios in Figures 4.3-4.5 and 4.6-4.8. In the dense scenarios the number of fetches is relatively high, and increases with k, while in the sparse ones the number of fetches stabilises. This is related with the fact that anonymous P2P resouces are serving different client requests (original P2P resource contents). Thus, although there is a need to create anonymous P2P resources, these will not be very requested in sparse scenarios. The anonymous are more requested in dense scenarios. Thus, for a certain density level scalability is ensured after some point (k value).

Heuristic Behaviour

The plots in Figure 4.9-4.11 show the average number of expansion operations, direct and swaps, performed by the heuristic. The heuristic seems to make very little use of swap operations, the direct ones being much more productive. This allows us to conclude that reducing complexity, by eliminating

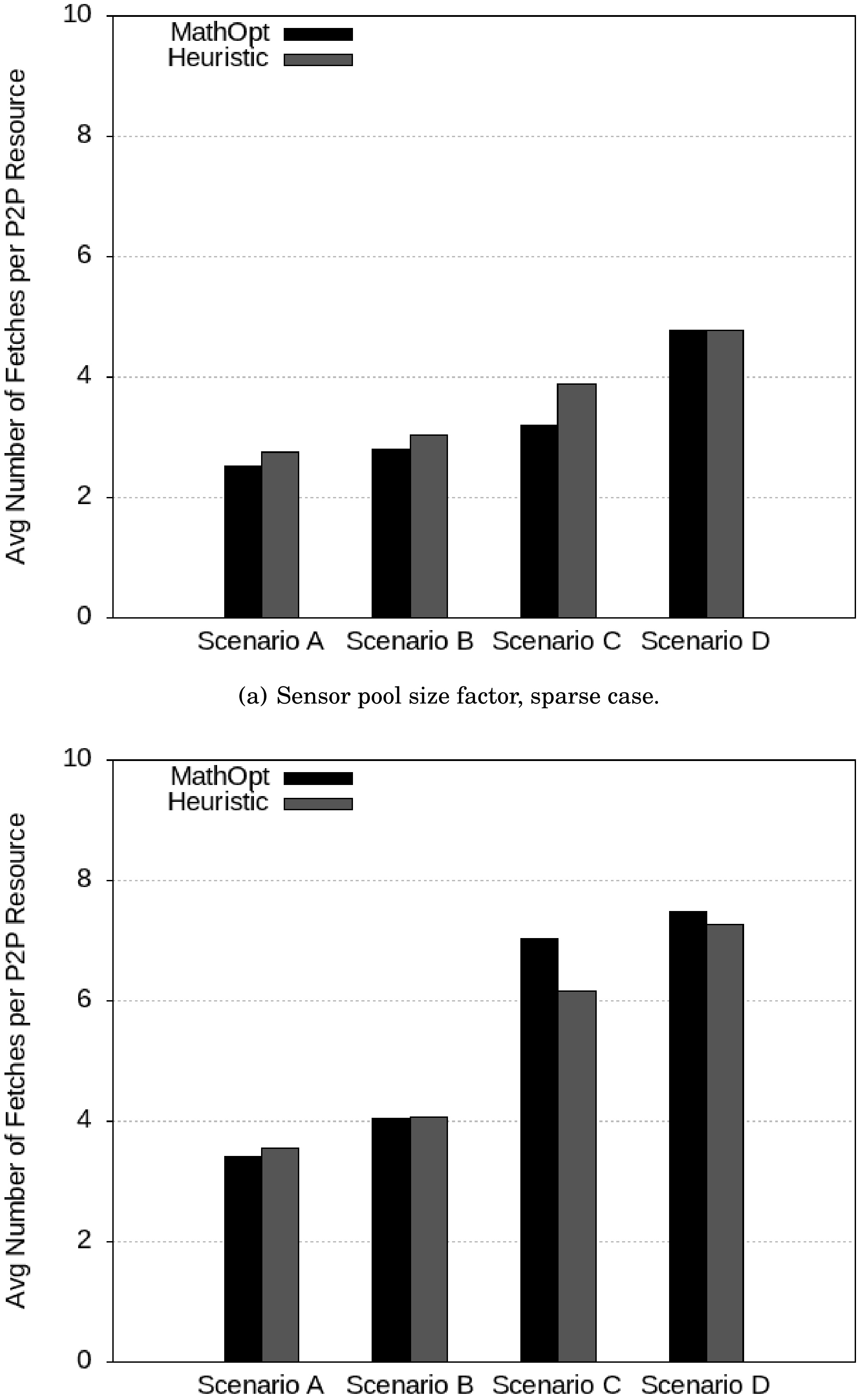


(a) Sensor pool size factor, sparse case.



(b) Sensor pool size factor, dense case.

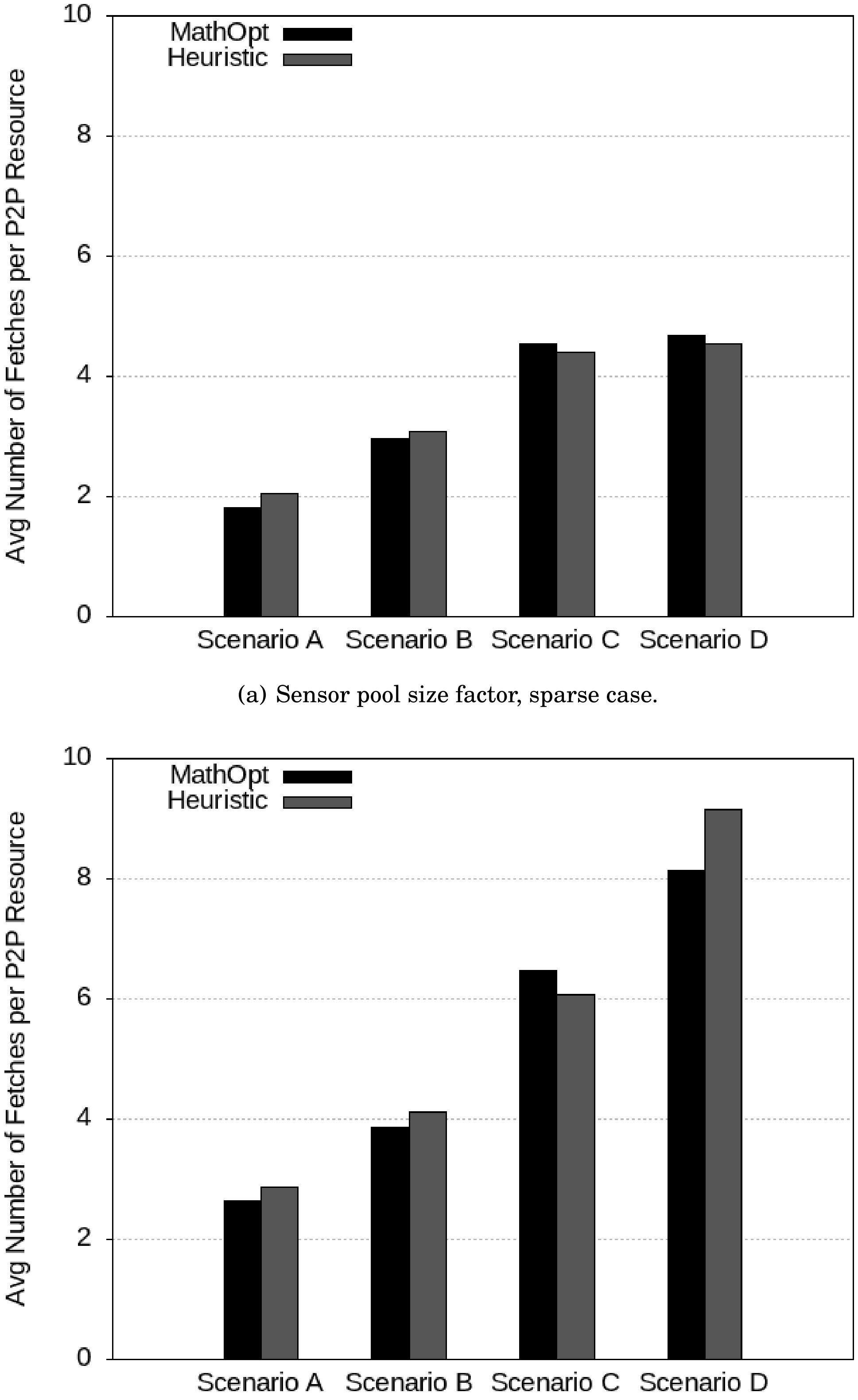
Figure 4.6: Average number of fetch operations for the heuristic and optimal approaches, $k = 1$.



(a) Sensor pool size factor, sparse case.

(b) Sensor pool size factor, dense case.

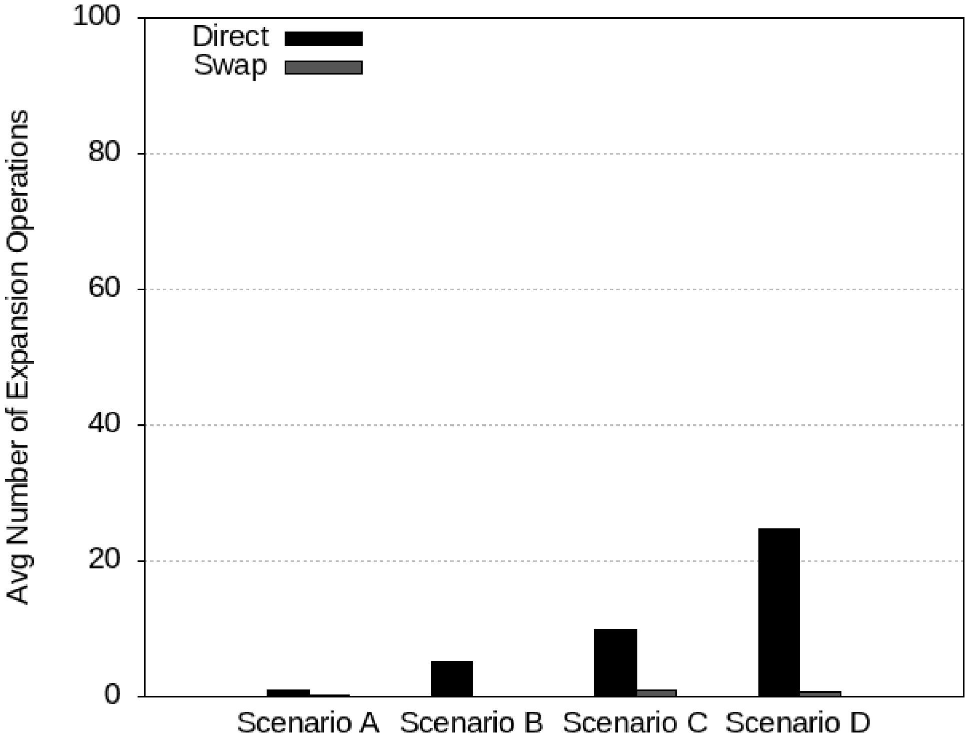
Figure 4.7: Average number of fetch operations for the heuristic and optimal approaches, $k = 2$.



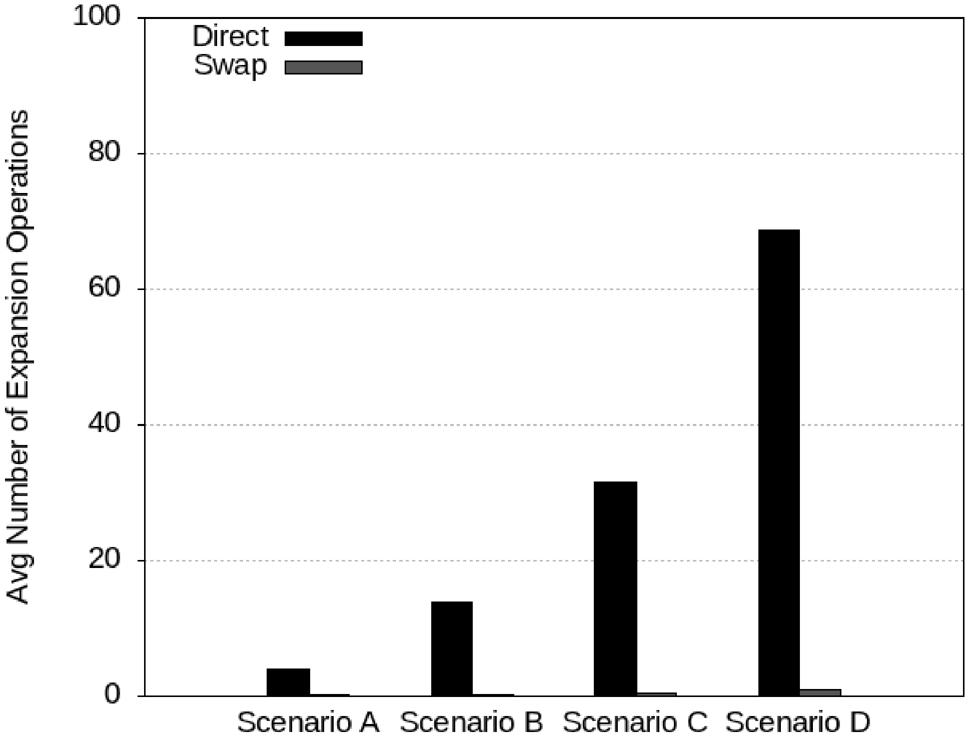
(a) Sensor pool size factor, sparse case.

(b) Sensor pool size factor, dense case.

Figure 4.8: Average number of fetch operations for the heuristic and optimal approaches, $k = 3$.

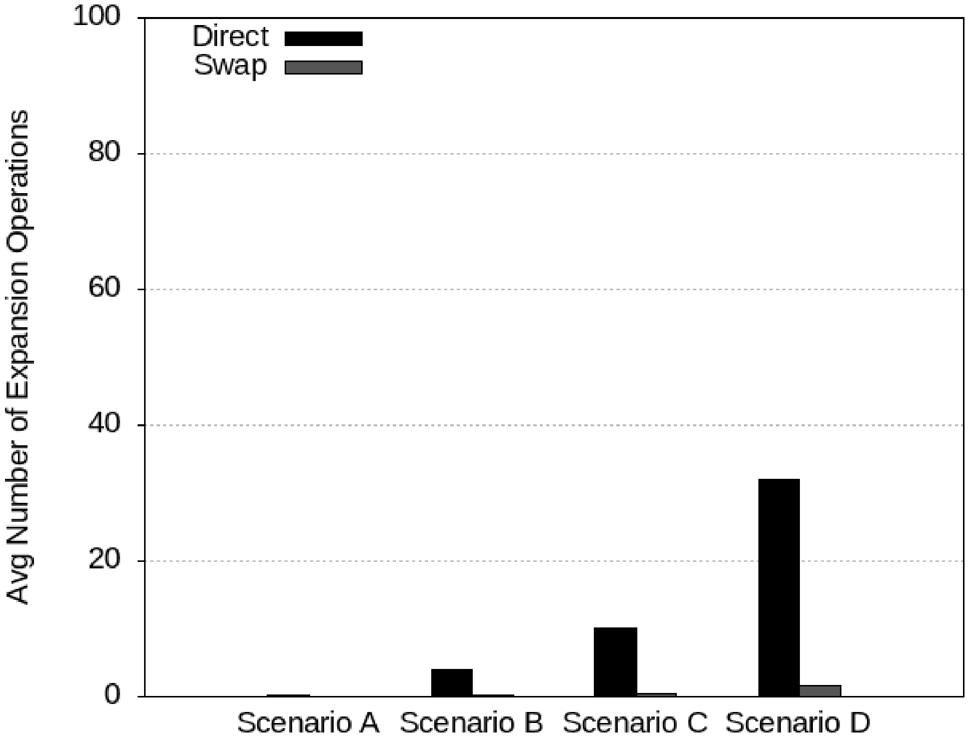


(a) Sensor pool size factor, sparse case.

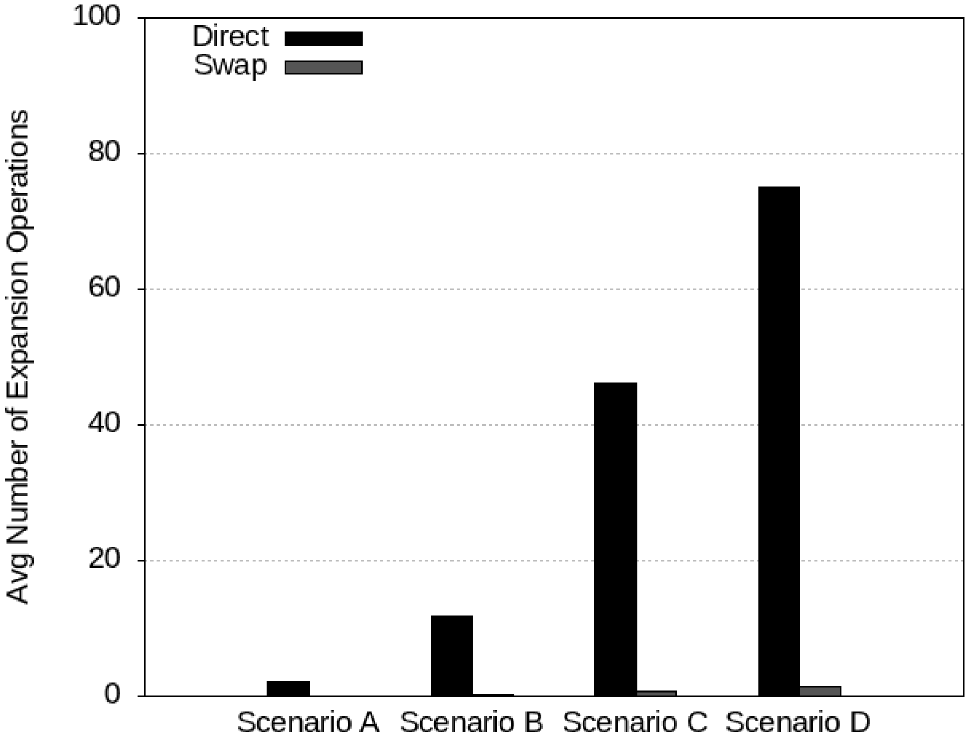


(b) Sensor pool size factor, dense case.

Figure 4.9: Average number of direct and swap expansion operations performed by the heuristic, $k = 1$.

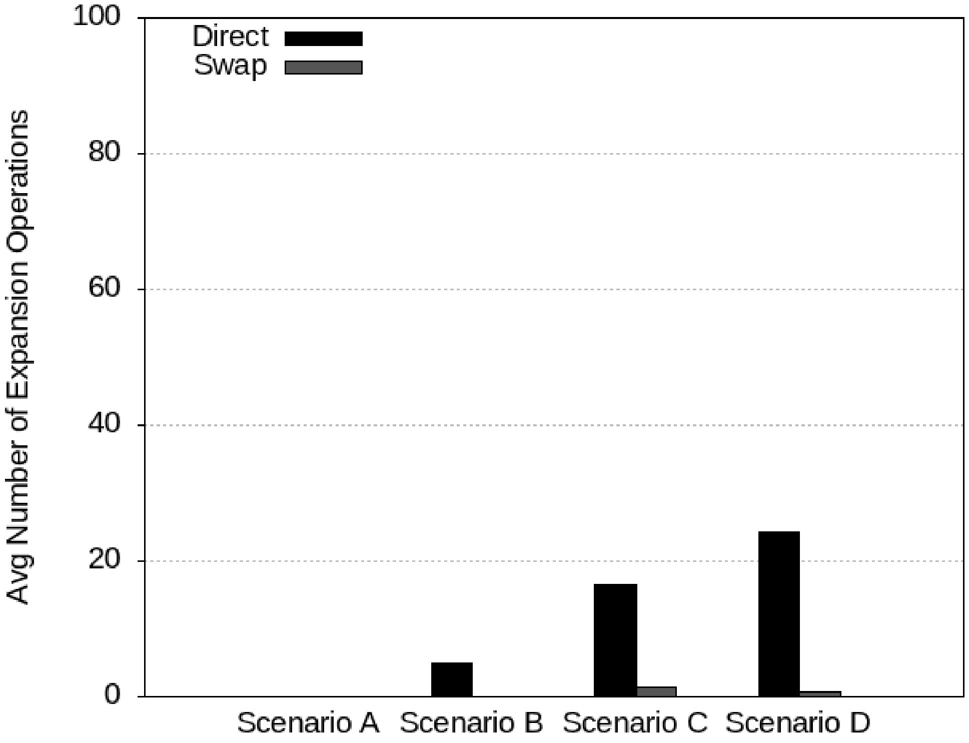


(a) Sensor pool size factor, sparse case.

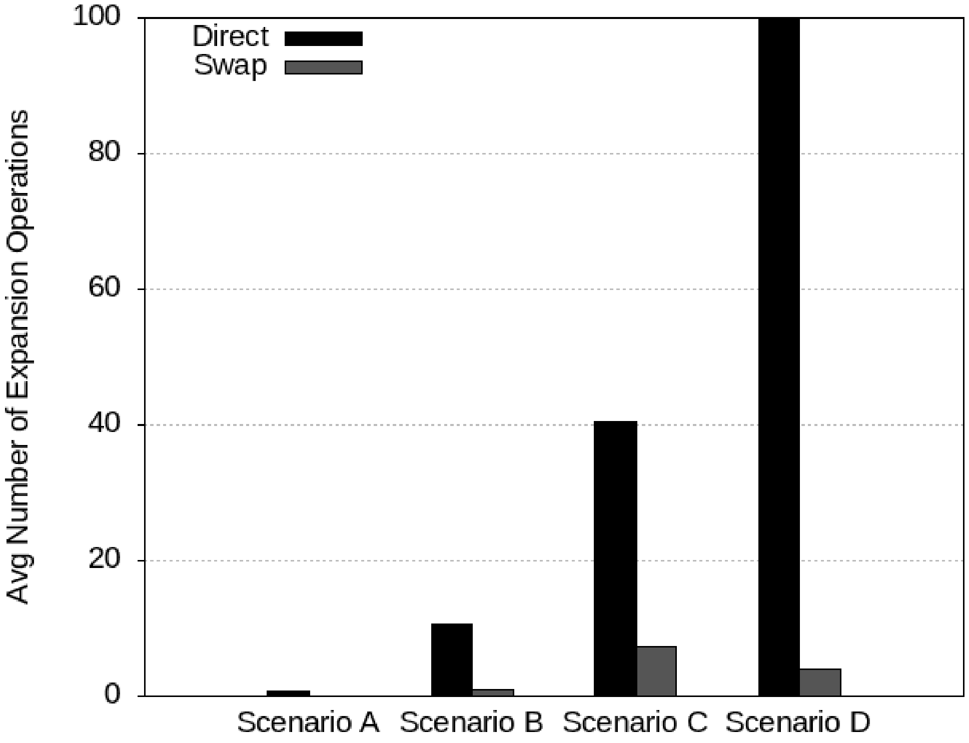


(b) Sensor pool size factor, dense case.

Figure 4.10: Average number of direct and swap expansion operations performed by the heuristic, $k = 2$.



(a) Sensor pool size factor, sparse case.



(b) Sensor pool size factor, dense case.

Figure 4.11: Average number of direct and swap expansion operations performed by the heuristic, $k = 3$.

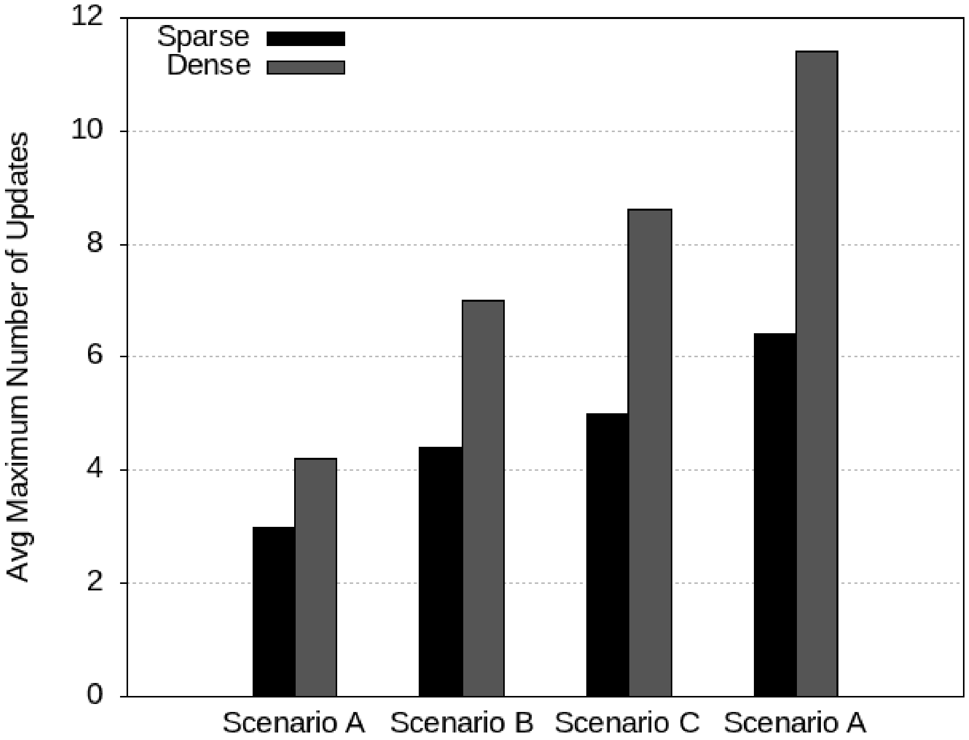


Figure 4.12: Average maximum number of updates in the traditional approach, for sparse and dense scenarios, $k = 1$.

swaps in the heuristic, will have very little impact. The benefit of swap operations in clique expansions is highly dependent on underlying claims, related with weights. Ultimately, the highest efficiency of swaps is reached when the focus is to increase the cliques under no conditions.

The low number of expansion operations in sparse scenarios is related with the low connectivity of compatibility graphs, which reduces both direct and swap operations. Increasing k also does not necessarily lead to an increase in the number of expansion operations. In fact, from $k = 2$ to $k = 3$ it decreases because the probability of similar device resource entries decreases, and an increase in compatibility graph size leads to a decrease in connectivity.

Maximum Number of Updates in Traditional Approach

The plots in Figure 4.12-4.14 show the average maximum number of updates when the traditional approach is used. For the multi-layer approach (Math-Opt and Heuristic) there will be a single update. Both the sparse and dense scenarios, for different sensor pool sizes, are shown. Such maximum number of updates is basically the highest number of similar device resource entries in the original P2P resources and, therefore, the maximum number of P2P resources that would have to be updated in case of device resource

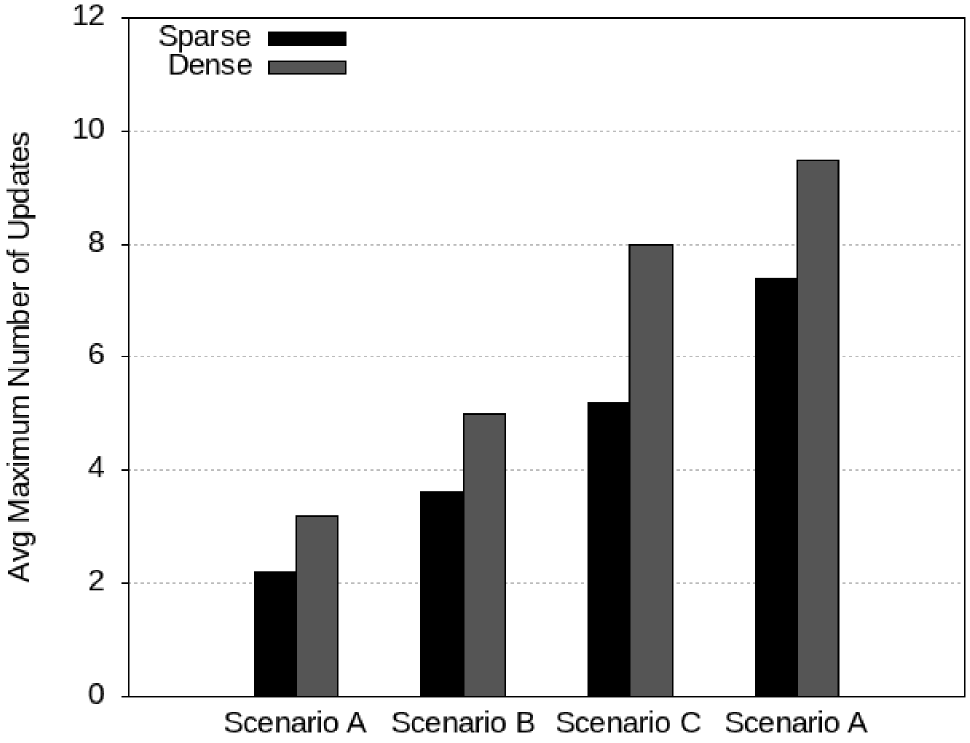


Figure 4.13: Average maximum number of updates in the traditional approach, for sparse and dense scenarios, $k = 2$.

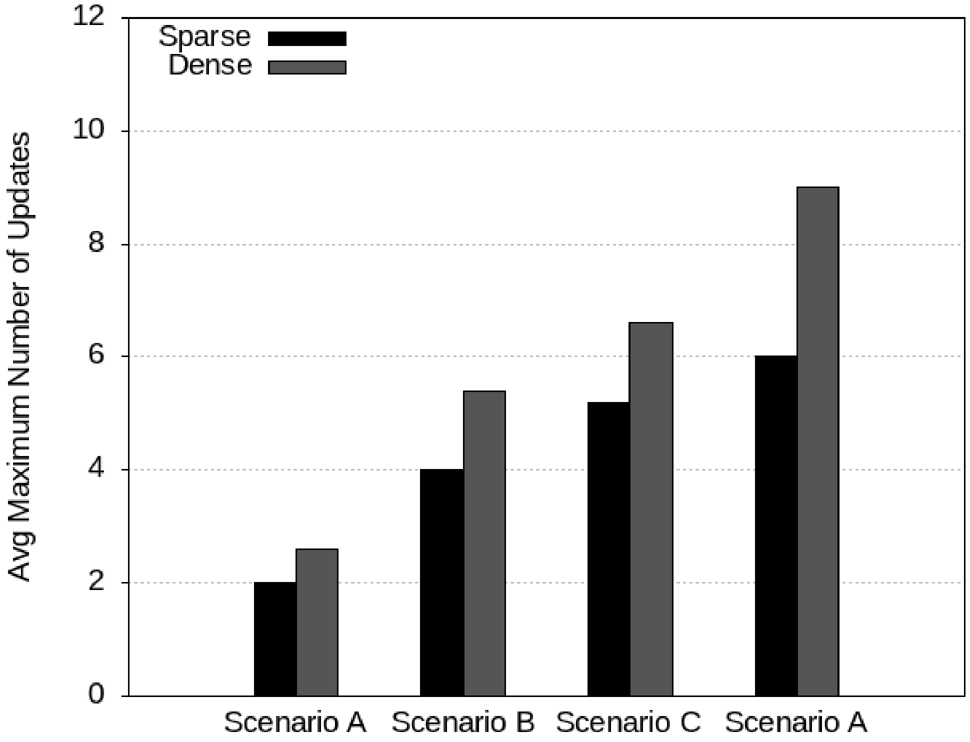


Figure 4.14: Average maximum number of updates in the traditional approach, for sparse and dense scenarios, $k = 3$.

change/removal.

Results show that dense scenarios generate more updates than sparse scenarios, as expected. When the sensor pool size factor increases, the average maximum number of updates decreases because the population/diversity of sensors increases, reducing the probability of having similar entries.

4.5.3 Final Remarks

Both traditional and multi-layer fetching schemes can be used in RELOAD/CoAP architectures. The first has the advantage of requiring a single fetch for the retrieval of P2P resources, but requires many updates to keep the information consistent. The second requires building a layer of anonymous P2P resources, to avoid having duplicate entries in different P2P resources requested by clients. Of course, consistency is kept at the expense of additional fetches (of anonymous P2P resources) for the original content of P2P resources to be rebuilt, before delivering it to the client. The client is, therefore, not aware of such anonymous P2P resources. In scenarios where device entries do not change, a traditional approach would fit. However, future scenarios will have to deal with the dynamic removal, change and insertion of device resource entries in P2P resources, and in this case the multi-layer scheme is better. Of course, all P2P resources having those entries could be removed and reinserted into the P2P overlay after changes. However, this can bring instability because many entries can change at multiple places simultaneously, and there will be different combinations of sensors under different P2P resource umbrellas. Under such dynamic scenario, the use of anonymous resources can better ensure consistency of device resource entries.

4.6 Summary

In this chapter a multi-layer fetching approach is proposed that uses anonymous P2P resources to keep information at the overlay network consistent, allowing for an efficient storage/retrieval of IoT data. A mathematical optimization model is proposed for the planning of additional anonymous P2P resources, and necessary bindings, that are necessary to ensure consistency. A heuristic algorithm is also proposed. Results show that the heuristic is quite scalable, providing high quality solutions independently of the size of the problem. Such feature allows us to conclude that this solution is suitable for RELOAD/CoAP overlay networks of any size.

Algorithm 2: Pseudocode of heuristic algorithm.

```
/* STEP: Initialization */
for each $n \in \mathcal{N}$ do
    for each $n' \in \mathcal{N} : n' \neq n$ do
        if $\mathcal{E}(n) \cap \mathcal{E}(n') \neq \emptyset$ then
            $\mathcal{L} \leftarrow (n, n')$
        end
    end
end
/* STEP: Resource Redesign */
while $(l = \textsc{HighestWeightLink}(\mathcal{L})) \neq NULL$ do
    $\mathcal{I} = \mathcal{E}(n) \cap \mathcal{E}(n')$, where $l = (n, n')$
    $\mathcal{C} = \{l\}$ /* initial clique */
    repeat
        $\mathcal{Q} = \emptyset$ /* list of possible expansions */
        for each $n \in \mathcal{X}(\mathcal{C})$ do
            if $|\mathcal{C} \backslash \mathcal{N}_n| = 0$ then
                if $\mathcal{I} \cap \mathcal{E}(n) \neq \emptyset$ then
                    $\mathcal{Q} \leftarrow \{n, weight = |\mathcal{I} \cap \mathcal{E}(n)|\}$
                end
            end
            if $|\mathcal{C} \backslash \mathcal{N}_n| = 1$ then
                $m = \{n' \in \mathcal{N}_n : (n, n') \notin \mathcal{L}\}$
                $\mathcal{C}' = \mathcal{C} \backslash \{m\} \cup \{n\}$
                $\mathcal{I}'$=$\textsc{UpdateInter}(\mathcal{C}')$
                for each $n' \in \mathcal{X}(\mathcal{C}')$ do
                    if $|\mathcal{C}' \backslash \mathcal{N}_{n'}| = 0$ then
                        if $\mathcal{I}' \cap \mathcal{E}(n') \neq \emptyset$ then
                            $\mathcal{Q} \leftarrow \{n', weight = |\mathcal{I}' \cap \mathcal{E}(n')|\}$
                        end
                    end
                end
            end
        end
        /* pick highest weight $\mathcal{Q}$ element */
        $n^* = \textsc{BestCliqueExpansion}(\mathcal{Q})$
        /* expand $\mathcal{C}$ and update $\mathcal{I}$ */
        $\mathcal{C}$=$\textsc{Expand}(\mathcal{C}$,$n^*)$
        $\mathcal{I}$=$\textsc{UpdateInter}(\mathcal{C})$
    until $\mathcal{Q} = \emptyset$ ;
    Build anonymous P2P resource with $\mathcal{I}$ and update $\mathcal{G}$.
end
```

CHAPTER 5

Procedures for On Demand P2P Resource Redesign

5.1 Introduction

The adoption of CoAP Usage is not enough to deal with the resource consistency problem that may arise in future scenarios. For this reason, a multi-layer fetching approach is proposed in the previous chapter. The idea is for anonymous P2P resources to be created, and content of announced P2P resources to be changed in order to point to such anonymous resources. While the previous chapter shows how to redesign a set of original P2P resources, for such multi-layer organization to be implemented, new P2P resources will arrive in the future or existing ones will be removed/modified. This requires procedures for information to be kept consistent at the overlay network. That is, no redesign from scratch is required and on demand procedures can be implemented to keep information consistent. This chapter presents procedures to face on demand P2P resource creation, removal and update.

This chapter is organized as follows. In Section 5.2 some assumptions are introduced and a discussion on the operation of the binding service at the RELOAD/CoAP overlay network, now considering the on demand redesign of existing resources, is presented. Section 5.3 presents the resource redesign procedures. In Section 5.4 an analysis and discussion of results is presented, and finally Section 5.5 presents a summary of the chapter.

Contributions

The contributions presented in this chapter are the following:

- Procedures are proposed to keep the population of P2P anonymous resources, and any reference to them, updated over time as new P2P resources are announced or existing ones are updated/removed.

5.1 Introduction

These contributions were published in:

- L. Rodrigues, J. Guerreiro and N. Correia, “Resource Redesign in RELOAD/CoAP Overlays for the Federation of Sensor Networks”, 9th EAI International Conference on Broadband Communications, Networks, and Systems (Broadnets), September 2018, Portugal.

5.2 Assumptions and Operation

Let us assume a set of original RELOAD/CoAP P2P resources, $\mathcal{R}$, where each $r \in \mathcal{R}$ includes a set of `(KEY,VALUE)` entries denoted by $\mathcal{P}_r$, where each $p_r \in \mathcal{P}_r$ includes a set of device resource entries denoted by $\mathcal{E}_{p_r}$. Let us assume also that P2P resources in $\mathcal{R}$ were redesigned according to the previously mentioned two-layer overlay approach, giving rise to $\mathcal{R}'$. That is, its content has changed so that references to P2P anonymous resources are used to avoid duplicates. The set of created P2P anonymous resources is denoted by $\mathcal{A}$. Such redesign should be done having the following assumptions as a basis.

Assumption 3 (P2P Resource Content)**.** *The content of a P2P resource* $r \in \mathcal{R}' \cup \mathcal{A}$ *will be of type* $T(r)$*, where* $T(r) \in \{\text{“}Anonymous\text{”}, \text{“}KeyValue\text{”}\}$*. More specifically, if* $r \in \mathcal{R}'$ *then* $T(r) = \{Anonymous\}$*, meaning that an entry, denoted by* $p_r \in \mathcal{P}_r$*, will be a reference to a P2P anonymous resource following the CoRE Link Format. If* $r \in \mathcal{A}$*, then* $T(r) = \{Anonymous\}$ *if* r *is not a leaf and* $T(r) = \{KeyValue\}$ *if* r *is a leaf.*

This means that a resource in $\mathcal{R}'$ will always have links to P2P anonymous resources, while P2P anonymous resources in $\mathcal{A}$ will have either links to leaf P2P resources with `(KEY,VALUE)` entries or links to other P2P anonymous resources.

Assumption 4 (Number of Entries)**.** *A device resource entry can not be included in more than one P2P resource. That is, for any* $e \in \mathcal{E}_{p_r}$*,* $p_r \in \mathcal{P}_r$ *and* $r \in \mathcal{R}$*, the following must hold:* $|\{a \in \mathcal{A} : e_{p_a} = e, e_{p_a} \in \mathcal{E}_{p_a}, p_a \in \mathcal{P}_a\}| = 1$*. A P2P anonymous resource entry can be included in multiple P2P resources.*

Assumption 5 (Strict Coverage)**.** *P2P resources must be redesigned while not changing the content to be returned. That is,* $\bigcup_{p_r \in \mathcal{P}_r} \mathcal{E}_{p_r} = \bigcup_{p_{r'} \in \mathcal{P}_{r'}} \mathcal{E}_{p_{r'}}$*,* $\forall r \in \mathcal{R}$*,* $\forall r' \in \mathcal{R}'$*.*

Upon a P2P resource fetch, an overlay service[1] would have to fetch the referenced P2P anonymous resources recursively, so that device resource entries are reached and returned to the client following the CoAP Usage format. This information is the one used by clients to perform AppAttachs.

To keep assumptions valid, P2P resources (anonymous included) will have to be updated when P2P resources are created/modified/removed. A set of

[1] A service discovery mechanism like ReDiR can be used to distribute load among RELOAD/CoAP nodes able to provide such service, ensuring scalability.

merge/split procedures, for this purpose, are discussed next. These procedures assume the existence of a P2P resource per proxy (e.g., `coap://overlay-1.com/KEY=9996172`) containing references to the P2P anonymous resources including proxy's `(KEY,VALUE)` entries. These are referred to as proxy P2P resources.

5.3 Resource Redesign Procedures

5.3.1 P2P Resource Creation

When a new P2P resource r is to be created, initially with `(KEY,VALUE)` entries in its content, the procedure CREATE shown below must be performed. At line 3, the P2P anonymous resources, whose entries are fully included in r, are obtained. A call to these should replace corresponding entries in r. Line 7 fetches P2P anonymous resources sharing content with r. These should be analyzed by SPLIT procedure. At this procedure, the splitting is performed at lines 2-4. This is recursive so that parent nodes are analyzed for splitting too (call at line 10).

Algorithm 3: CREATE(r).

```
1  for $p_r \in \mathcal{P}_r$ do
2      Extract $p_r$'s KEY and fetch corresponding proxy P2P resources, $r'$
3      $\mathcal{I} = \{p_{r'} \in \mathcal{P}_{r'} : p_{r'} \subseteq p_r\}$
4      for $p_i \in \mathcal{I}$ do
5          Replace $p_i \cap p_r$ content in $r$ by reference $p_i$
6      end
7      $\mathcal{I}' = \{p_{r'} \in \mathcal{P}_{r'} \backslash \mathcal{I} : p_{r'} \cap p_r \neq \emptyset\}$
8      SPLIT($\mathcal{I}', p_r$)
9      for $p_i \in \mathcal{I}'$ do
10         Replace $p_i \cap p_r$ content in $r$ by reference $p_i$
11     end
12     Create P2P anonymous resource if intact $p_r$ content exists, and
       replace it by reference.
13 end
```

Algorithm 4: SPLIT($\mathcal{I}', p_r$).

```
for $p_i \in \mathcal{I}'$ do
    Create resource with content $p_i \cap p_r$
    Create resource with content $p_r \backslash \{p_i \cap p_r\}$
    Update references at resources in $rt(p_i)$
end
if no changes exist then
    Return
end
for $p_i \in \mathcal{I}'$ do
    SPLIT$(rt(p_i), p_r)$
    Delete $p_i$
end
```

5.3.2 P2P Resource Removal

When a P2P resource $r \in \mathcal{R}'$ is to be removed, the procedure REMOVE shown below must be performed. Lines 2-5 will remove references to the resource being deleted. The MERGE procedure analyses common content, for possible joins. At this procedure, the $\mathbb{P}()$ is the powerset. Merges will be performed at children nodes recursively. It is assumed that the corresponding rt information is updated accordingly.

Algorithm 5: REMOVE(r).

```
1 for $p_r \in \mathcal{P}_r$ do
2     for $i \in rt(p_r)$ do
3         Fetch $i$
4         Remove $r$ from $rt(\{p_i \in \mathcal{P}_i : p_i = p_r\})$
5     end
6     MERGE($rt(p_r)$)
7 end
8 Delete $r$
```

Algorithm 6: MERGE($resources$).

```
$\mathcal{U} = \bigcup_{i \in resources} \mathcal{P}_i$
$\mathcal{I} = \arg\max_{\mathcal{S} \in \mathbb{P}(\mathcal{U})}(|\{i \in \mathcal{S} : rt(i) = \bigcap_{j \in \mathcal{S}} rt(j)\}|)$
if $|\mathcal{I}| = 1$ then
    Replace content $\mathcal{I}$, at every $r' \in resources$, by $i \in \mathcal{I}$
    Delete $i \in \mathcal{I}$
    MERGE($resources$)
end
else
    if $|\mathcal{I}| > 1$ then
        Create new resource $r$ with content of every $i \in \mathcal{I}$
        Replace content $\mathcal{I}$, at every $r' \in resources$, by $r$
        Delete every $i \in \mathcal{I}$
        if $r$ is not of leaf then
            MERGE($r$)
        end
    end
end
```

5.3.3 P2P Resource Update

It is assumed that a P2P resource removal is performed, followed by a P2P resource creation.

Table 5.1: Scenarios under test.

Scenario	Public P2P resources	Anonymous resources per level	Refs for Anonymous content	Refs for KeyValue content
A	[1-10]	[5-10]	[1-5]	[1-5]
B	[1-20]	[10-20]	[1-10]	[1-10]
C	[1-30]	[15-30]	[1-15]	[1-15]
D	[1-40]	[20-40]	[1-20]	[1-20]
E	[1-50]	[25-50]	[1-25]	[1-25]
F	[1-60]	[30-60]	[1-30]	[1-30]

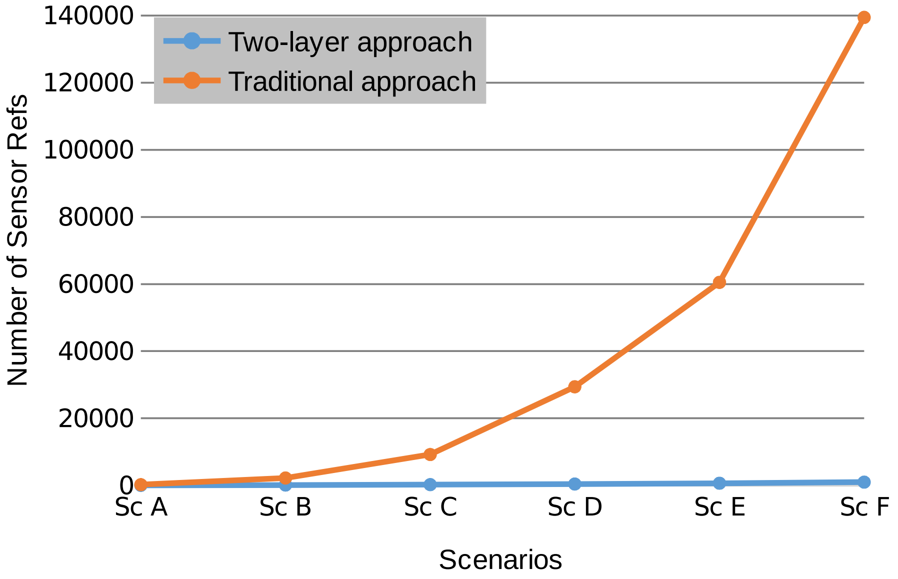


Figure 5.1: Number of sensor references.

5.4 Analysis of Results

To evaluate the advantages of having P2P anonymous resources, different scenarios were tested. In such scenarios there are public P2P resources (URIs known to the public), each having a set of P2P anonymous resource entries (i.e., of Anonymous content type). The P2P anonymous resources being referenced can then be of Anonymous or KeyValue content type. In the first case another calling level is being built, which may reference any existing P2P anonymous resources. A maximum of two levels exists. Table 5.1 shows the ranges used to define the number of P2P resources or resource/sensor entries (a random number is picked to run a test). The following plots show the average of 20 tests performed for each scenario.

As shown in Figures 5.1 and 5.2, if P2P anonymous resources are not created then the total number of sensor entries will grow exponentially due to

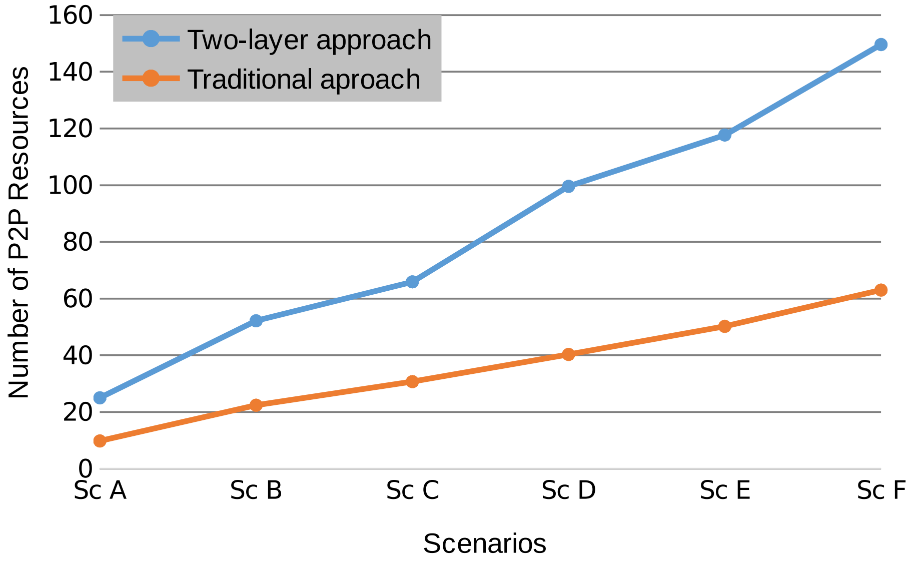


Figure 5.2: Number of P2P resources.

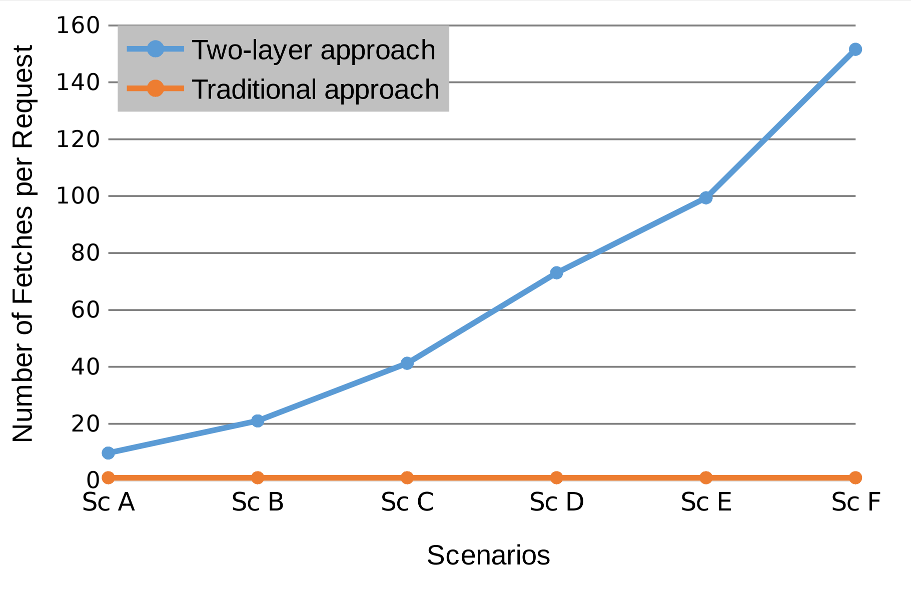


Figure 5.3: Number of fetches per public P2P resource.

duplicate entries. The proposed solution is able to keep a single reference to sensor entries, resulting into very low values for the total number of entries. Duplicate entries are avoided, however, at the expense of additional fetches. Figure 5.3 shows the number of required fetches per public P2P resource. Note, however, that fetching in parallel will reduce latency (i.e., for two levels the overall delay converges to the time of two fetches in series).

5.5 Summary

Resource redesign procedures were proposed to keep P2P resources in RELOAD/CoAP architectures updated over time, while ensuring that sensor resource entries are unique. This ensures data consistency, better coordination of cooperating systems, and timeliness of notifications. Results show that these procedures are able to keep P2P resources consistent and sensor resource entries remain unique.

CHAPTER 6

Conclusions and Future Work

6.1 Conclusions

CoAP will be the application layer protocol to be adopted by IoT devices, and in order to be able to federate CoAP based constrained networks that are geographically dispersed, a CoAP Usage was recently proposed for the RELOAD protocol. The RELOAD is a P2P protocol that ensures an abstract storage and messaging service based on a set of cooperating peers that form a P2P overlay network for this purpose. Therefore, RELOAD/CoAP architectures will allow constrained networks to be federated for wide-area geographical coverage. In this case, proxy nodes of constrained environments form a P2P overlay to announce device resources or sensor data. Although this is a standard-based solution, consistency problems may arise because P2P resources (data objects stored at the overlay network) may end up including similar device resource entries. This is so because device resource entries, or sensor data, can be announced under different P2P resource umbrellas, meaning that any update to them will require changing multiple P2P resources. This means that mechanisms to ensure the consistency of P2P resources end up being required.

In this dissertation, a resource binding model is proposed for P2P resources to be able to include bindings to other P2P resources available in the overlay network. Besides this binding model, a multi-layer model for P2P resources to be able to keep information at the overlay network consistent, avoiding duplicates, is proposed together with a RELOAD/CoAP overlay network service for its operationalization.

Besides presenting the previously mentioned binding and multi-layer models, optimization procedures and heuristic algorithms are discussed and their performance is evaluated. Results show that heuristics are scalable, providing high quality solutions independently of the size of the problem. Such feature allows us to conclude that these solutions, for the organization P2P resources content, are suitable for RELOAD/CoAP overlay networks of any size. These features end up facilitating the emergence of a larger number of sensor network federations. The availability of different aggregates of IoT resources, using these distributed architectures, will in turn allow the emergence of applications with an innovative nature and based on the integration of IoT resources.

6.2 Future Work

P2P systems offer an efficient way to share various resources and access diverse services over the Internet, and for this reason RELOAD/CoAP architectures provide an effective way of federating constrained networks. To further improve these systems, an effective selection of peers to store P2P resources (original and/or anonymous) can be provided. More specifically, storage peers can be selected according to the relationship between P2P resources, which basically depends on bindings, using dynamic approaches like reinforcement learning. Besides such optimization, these dynamic approaches can be used to select which peers should provide the proposed RELOAD/CoAP overlay network service. These kind of procedures will be explored in a near future.

Bibliography

[1] MQTT V3.1.1 Protocol Specification. Tech. rep., 2014.

[2] MQTT V5.0 Protocol Specification. Tech. rep., 2019.

[3] A. LUDOVICI, E. G. X. G., AND AUGE, A. C. Adding QoS Support for Timeliness to the Observe Extension of CoAP. In *IEEE Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)* (2012).

[4] ALANAZI, F., AND YEFERNY, T. Reinforcement Learning Based Query Routing Approach for P2P Systems. *Future Internet 11*, 253 (2019).

[5] ALSULAMI, M. M., AND AKKARI, N. The Role of 5G Wireless Networks in the Internet-of-Things (IoT). In *International Conference on Computer Applications and Information Security (ICCAIS)* (2018).

[6] AMINA BOUBENDIR, E. A. 5G Edge Resource Federation: Dynamic and Cross-domain Network Slice Deployment. In *IEEE Conference on Network Softwarization and Workshops (NetSoft)* (2018).

[7] AMOL BHAGAT, R. C., AND DONGRE, K. Content-based File Sharing in Peer-to-peer Networks Using Threshold. In *International Conference on Communication, Computing and Virtualization (ICCCV)* (2016).

[8] ATHANASIOS KARAPANTELAKIS, E. A. DevOps for IoT Applications Using Cellular Networks and Cloud. In *International Conference on Future Internet of Things and Cloud (FiCloud)* (2016).

[9] BERNSTEIN, D., AND VIJ, D. Intercloud federation using via semantic resource federation API and dynamic SDN provisioning. In *International Conference and Workshop on the Network of the Future (NOF)* (2014).

[10] BRAHIM DJELLABI, M. Y., AND AMAD, M. Effective peer-to-peer design for supporting range query in Internet of Things applications. *Computer Communications, Elsevier 150* (2020).

[11] CHONG ZHANG, WEIDONG XIAO, D. T., AND TANG, J. P2P-based Multi-dimensional Indexing Methods: A Survey. *Journal of Systems and Software, Elsevier 84*, 12 (2011), 2348–2362.

[12] COHEN, R., AND KATZIR, L. The Generalized Maximum Coverage Problem. *Information Processing Letters 108*, 1 (2008), 15–22.

[13] DA XU, E. A. Peer-to-Peer Multi-Energy and Communication Resource Trading for Interconnected Microgrids. *IEEE Transactions on Industrial Informatics (Early Access)* (2020).

[14] DI ZHANG, E. A. One Integrated Energy Efficiency Proposal for 5G IoT Communications. *IEEE Internet of Things Journal 3*, 6 (2016), 1346–1354.

[15] E. AL-HAWRI, N. C., AND BARRADAS, A. RELOAD/CoAP P2P Overlays for Network Coding Based Constrained Networks. In *IFIP Advances in Information and Communication Technology* (2017).

[16] E. AL-HAWRI, N. C., AND BARRADAS, A. Design of Network Coding Based Reliable Sensor Networks. *AdHoc Networks, Elsevier 91* (2019).

[17] E. AL-HAWRI, N. C., AND BARRADAS, A. DAG Coder for Reliability in Wireless Sensor Networks. *IEEE Access 8* (2020), 21886–21896.

[18] ET AL, C. J. REsource LOcation And Discovery (RELOAD) Base Protocol. RFC 6940, Internet Engineering Task Force (IETF), 2014.

[19] ET AL, C. J. A SIP Usage for REsource LOcation And Discovery (RELOAD). RFC 7904, Internet Engineering Task Force (IETF), 2016.

[20] ET AL, J. J. A Constrained Application Protocol (CoAP) Usage for REsource LOcation And Discovery (RELOAD). RFC 7650, Internet Engineering Task Force (IETF), 2015.

[21] ET AL, Z. S. The Constrained Application Protocol (CoAP). RFC 7252, Internet Engineering Task Force (IETF), 2014.

[22] FABRIZIO MESSINA, G. P., AND SANTORO, C. Decentralised Resource Finding and Allocation in Cloud Federations. In *International Conference on Intelligent Networking and Collaborative Systems* (2014).

[23] (GDF), T. G. D. F. The Annotated Gnutella Protocol Specification v0.4. Tech. rep., The Gnutella Developer Forum (GDF).

[24] GIANLUCA ZANGARA, E. A. A Cloud Federation Architecture. In *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)* (2015).

[25] HARTKE, K. Observing Resources in the Constrained Application Protocol (CoAP). RFC 7641, Internet Engineering Task Force (IETF), 2015.

[26] I. BAUMGART, B. H., AND KRAUSE, S. OverSim: A Flexible Overlay Network Simulation Framework. In *IEEE Global Internet Symposium* (2007).

[27] IOAN PETRI, ATEYAH ALZAHRANI, J. R., AND REZGUI, Y. Federating Smart Cluster Energy Grids for Peer-to-Peer Energy Sharing and Trading. *IEEE Access 8* (2020), 2169–3536.

[28] J. GUERREIRO, L. R., AND CORREIA, N. Allocation of Resources in SAaaS Clouds Managing Thing Mashups. *IEEE Transactions on Network and Service 17*, 3 (2020), 1597–1609.

[29] JORGE BARANDA HORTIGÜELA, E. A. Realizing the Network Service Federation Vision: Enabling Automated Multidomain Orchestration of Network Services. *IEEE Vehicular Technology Magazine 15*, 2 (2020), 48–57.

[30] JOUNI MÄENPÄÄ, J. J. B., AND LORETO, S. Using RELOAD and CoAP for Wide Area Sensor and Actuator Networking. *EURASIP Journal on Wireless Communications and Networking*, 121 (2012).

[31] J.P. AHULLÓ, P. L. PlanetSim: An Extensible Framework for Overlay Network and Services Simulations. In *International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (Simutools)* (2008).

[32] KAIS MEKKI, EDDY BAJIC, F. C., AND MEYER, F. Overview of Cellular LPWAN Technologies for IoT Deployment: Sigfox, LoRaWAN, and NB-IoT. In *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom)* (2018).

[33] KANG, M., AND KHASHNOBISH, A. A Peer-to-Peer Federated Authentication System. In *International Conference on Information Technology: New Generations* (2009).

[34] LAPACZ, R., AND PIETRZAK, B. Networking Solutions in the Federation of Clouds. In *International Conference on Network and Service Management (CNSM)* (2013).

[35] MAENPAA, J., AND CAMARILLO, G. Service Discovery Usage for REsource LOcation And Discovery (RELOAD). RFC 7374, Internet Engineering Task Force (IETF), 2014.

[36] MARIO MEIRELES TEIXEIRA, F. A. O. M., AND PINTO, A. V. A P2P Network for Multimedia Content Sharing Using Android-Based Mobile Devices. In *Workshop on Communication in Critical Embedded Systems* (2017).

[37] MONTRESOR, A., AND JELASITY, M. Peersim: A Scalable P2P Simulator. In *IEEE Ninth International Conference on Peer-to-Peer Computing* (2009).

[38] N. CORREIA, D. S., AND SCHÜTZ, G. Dynamic Aggregation and Scheduling in CoAP/Observe Based Wireless Sensor Networks. *IEEE Internet of Things Journal 3*, 6 (2016), 923–936.

[39] N. KUSHALNAGAR, G. M., AND SCHUMACHER, C. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. RFC 4919, Internet Engineering Task Force (IETF), 2007.

[40] NOTTINGHAM, M. Web Linking. RFC 5988, Internet Engineering Task Force (IETF), 2010.

[41] NOTTINGHAM, M., AND HAMMER-LAHAV, E. Defining Well-Known Uniform Resource Identifiers (URIs). RFC 5785, Internet Engineering Task Force (IETF), 2010.

[42] PARUL SHARMA, A. B., AND KAUSHAL, R. Performance analysis of BitTorrent protocol. In *The National Conference on Communications (NCC), India* (2013).

[43] PRASAD, A. R., AND BUFORD, J. F. *Future Internet Services and Service Architectures*. River Publishers Series in Communications, 2011.

[44] R. MAHY, P. M., AND ROSENBERG, J. Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN). RFC 5766, Internet Engineering Task Force (IETF), 2010.

[45] ROSENBERG, J. Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols. RFC 5245, Internet Engineering Task Force (IETF), 2010.

[46] RYAN RANDY SURYONO, B. P., AND BUDI, I. Peer to Peer (P2P) Lending Problems and Potential Solutions: A Systematic Literature Review. In *The Fifth Information Systems International Conference* (2019).

[47] SANTOS, J. Scalable Design of SDN Controllers for Optical Networks using Federation-based Architectures. In *European Conference on Networks and Optical Communications (NOC)* (2016).

[48] SEAN RHEA, E. A. OpenDHT: A Public DHT Service and Its Uses. In *ACM SIGCOMM* (2005).

[49] SHELBY, Z. Constrained RESTful Environments (CoRE) Link Format. RFC 6690, Internet Engineering Task Force (IETF), 2012.

[50] SHIVANGI SURATI, D. C. J., AND GARG, S. A Survey of Simulators for P2P Overlay Networks with a Case Study of the P2P Tree Overlay using an Event-Driven Simulator. *Engineering Science and Technology, an International Journal, Elsevier 20*, 2 (2017), 705–720.

[51] SHIVANGI VASHI, E. A. Internet of Things (IoT): A vision, architectural elements, and security issues. In *International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)* (2017).

[52] SON N. HAN, QUYET H. CAO, B. A., AND CRESPI, N. Design, implementation, and evaluation of 6LoWPAN for home and building automation in the Internet of Things. In *IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)* (2015).

[53] SOOKYOUNG LEE, MOHAMED YOUNIS, B. A., AND LEE, M. Load and Resource Aware Federation of Disjoint Sensor Network Segments. In *IEEE International Conference on Communications (ICC)* (2017).

[54] SOOKYOUNG LEE, M. Y., AND LEE, M. Optimized Bi-Connected Federation of Multiple Sensor Network Segments. *Ad Hoc Networks, Elsevier 38* (2016), 1–18.

[55] SYE LOONG KEOH, S. S. K., AND TSCHOFENIG, H. Securing the Internet of Things: A Standardization Perspective. *IEEE Internet of Things Journal 1*, 3 (2014), 265–275.

[56] Tiago Gomes, e. a. A 6LoWPAN Accelerator for Internet of Things Endpoint Devices. *IEEE Internet of Things Journal 5*, 1 (2018).

[57] Z. Shelby, e. a. CoRE Resource Directory, draft-ietf-core-resource-directory-24. draft, Internet Engineering Task Force (IETF), 2020.

List of Publications

Journals:

1. L. Rodrigues, J. Guerreiro and N. Correia, “Resource Design in Federated Sensor Networks using RELOAD/CoAP Overlay Architectures”, submitted to Internet of Things, Elsevier.
2. L. Rodrigues, J. Guerreiro and N. Correia, “RELOAD/CoAP architecture for the federation of wireless sensor networks”, Journal of Peer-to-Peer Networking and Applications, Springer, 2019.

Conferences:

1. L. Rodrigues, J. Guerreiro and N. Correia, “Resource Redesign in RELOAD/CoAP Overlays for the Federation of Sensor Networks”, 9th EAI International Conference on Broadband Communications, Networks, and Systems (Broadnets), September 2018, Portugal.
2. L. Rodrigues, J. Guerreiro and N. Correia, “RELOAD/CoAP Architecture with Resource Aggregation/Disaggregation Service”, IEEE PIMRC, IoT Workshop, September 2016, Spain.