CAEsAR: Making the RPL Routing Protocol Context-Aware

Andras Kalmar and Rolland Vida

***Abstract*—Due to the continuous development in hardware-, radio-, and sensor technologies, and the efforts of standardization organizations, the Internet of Things is not just a vision anymore, but it slowly becomes a part of our everyday life. The number of deployed sensors and actuators in our environment is increasing day-by-day transforming the physical world into an intelligent environment enabling context-aware services. To fully support this transformation we need to adapt the basic principles of communication. We do not want to know the IP addresses of individual sensor for example, we would rather like to query them based on their context. Also, we are often interested in the information itself, no matter which device provides it.**

In this paper we extend our formerly proposed addressing scheme for RPL networks (CAEsAR) to make it even more efficient. CAEsARv2 uses RPL trees and aggregates context information in Bloom-filters (BF) or bit vectors along the tree. With this addressing scheme the RPL protocol itself is enhanced to support context-based multicast, service-discovery and data-centric communication. Compared to our original proposal, in CAEsARv2 we get shorter update messages, as a result of assigning distinct data structures (Bloom filters or bit vectors) to each of the context parameters. We also show that by storing IP addresses also in Bloom filters, similarly to other context parameters, routing entries become shorter and evenly distributed among the nodes. Through simulations we demonstrate that the efficiency of Bloom-filter and bit vector aggregation in CAEsARv2 is not affected significantly by the radio ranges of the nodes in the network. Finally, through experimental results we show that, in case of correlation between geographical proximity and measured values, CAEsARv2 can adapt more efficiently to context changes than the centralized publish/subscribe messaging systems.

***Index Terms*—Internet of Things, RPL routing protocol, context-awareness, Bloom filters, multicast, service-discovery, data-centric communication.**

I. INTRODUCTION

THE Internet of Things has huge potential and countless opportunities, as it can revolutionize almost every aspect of our life. However, there are still some technical-, business-, and policy challenges that must be tackled before these systems can widely spread. Focusing on the technical aspects, most of the IoT devices will be resource-constrained, with limited memory, battery, and processing capabilities, and they will communicate mostly through wireless channels that are noisy. Our task is thus to provide communication standards and protocols that can operate efficiently even under these constraints.

The IETF has already standardized protocols like 6loWPAN [1] and RPL [2] enabling IoT devices with scarce resources to get an IPv6 address and connect to the Internet. However, in order to utilize the full potential of the future IoT infrastructure and to provide personalized, location-aware, and more generally context-aware services we need additional communication features. First, we need a *service discovery* mechanism [3], as we need to know what devices are available in a certain area and what is their current context (e.g., what services they are able to provide, what is their battery status, their geographic position, what operating system are they running, etc.) . Also, we might want to communicate with a set of IoT devices that share the same context - *context-based multicast* (e.g., we would like to communicate with all the smoke detectors in a given area that have a battery level above 50%). Lastly, IoT applications will be rather data-centric than message centric (i.e., they care about the data itself, and not about how and from whom it is being delivered).

This paper presents thus an extension of our formerly proposed Context-Aware Addressing and Routing scheme for RPL networks (a.k.a., CAEsAR) [4] [5], to efficiently support these above features: service discovery, context-based multicast and data-centric communication We denote this updated version by CAEsARv2.

Regarding service discovery in the IoT domain, traditional solutions [6] use a centralized registry, with which every device, offering any kind of service, communicates individually. This means that these devices have to send their registration, status update and keep-alive messages, possibly through multiple hops, to the registry. Compared to this, as we explain it later, CAEsARv2 uses Bloom filters (BFs) and bit vectors (BVs) to represent the current context of the devices, including their offered services. These BFs and BVs are aggregated along the RPL tree to which all these devices are attached. Changes in the context information of a device (e.g., changes in its position, battery status, measured value, etc.) may initiate update messages in the network, similarly to the updates sent to the centralized registry in traditional solutions. However, these updates could die out rapidly due to the BF and BV aggregation process (as explained later). Thus, the signaling burden in CAEsARv2 is much lower.

Regarding context-based multicast, in theory we might map application-layer context (subscriber) groups to network-layer multicast groups [7]. However, traditional IP multicast does not scale well with lots of small groups, since the multicast addresses cannot be aggregated, so a separate routing entry should be stored for each group. Nevertheless, with context-based group addressing we get exactly in this situation: we should maintain as many multicast groups as the number of all possible permutations of the defined parameters (e.g., we should build and maintain a separate multicast group and tree

A. Kalmar, R. Vida are with the Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Magyar Tudosok krt. 2., Budapest 1117, Hungary (email: kalmar,vida@tmit.bme.hu., url: http://www.tmit.bme.hu)

for all the smoke detectors on the third floor with battery above 50 %, one for those on the second floor and battery above 70%, one for the motion sensors on the ground floor that detected any movement in the last 5 minutes, and so on. Defining separate groups for these endless possible cases is clearly unmanageable for the resource-constrained IoT domain. With CAEsARv2 we provide a solution for that as well, as no individual multicast trees have to be maintained. Nodes with the same subset of context information, which would be members of a specific multicast group, will be reached easily via an efficient BF- and BV-based routing scheme over the RPL tree (as explained later).

Finally, data-centric communication in an IoT domain could be handled via traditional pub/sub systems. However, we think that the parties interested in some specific IoT data will typically not be other IoT devices from the same domain, but more likely applications that run on remote nodes, connected to this IoT domain through the traditional Internet. In this case the centralized and distributed pub/sub systems [8] [7] are identical in the sense that every report message from an IoT publisher has to be sent at least until the RPL root. In CAEsARv2 however these report messages may die out because of the already mentioned BF- and BV-aggregation process, representing thus a much smaller signaling burden.

The contributions of this paper are the followings:

- We introduce new design steps that decrease the needed memory and the length of the update messages for CAEsARv2. We make suggestions to separate the different parameters and to store them in distinct data structures. We examine what data structure (bit vector or Bloom filter) is worth to be assigned to each parameter, based on its type and its value range.
- We propose to store IP addresses in Bloom filters as well, similarly to other context parameters. We show that by doing so the routing entries become shorter and are distributed evenly among the RPL nodes.
- We validate through experiments that CAEsARv2 can adapt to context changes more efficiently than the centralized publish/subscribe messaging systems if there is a correlation between geographical proximity and measured values. In our former work [5] we only ran simulations targeting this aspect.
- Through simulations we prove that if the fore mentioned correlation exists, the efficiency of Bloom filter and bit vector aggregation in CAEsARv2 is not affected significantly by the radio ranges in the network, as opposed to the centralized solutions where shorter radio ranges means longer routes.

The remainder of this paper is organized as follows: in section II we review the related work; in section III we introduce the required background on the RPL protocol and the operation of CAEsAR. In section IV we introduce several design steps to improve CAEsAR, while in section V we evaluate the extended CAEsARv2 framework through simulations and experiments. Finally, in section VI we conclude our work.

II. RELATED WORK

AS mentioned before, we propose in this paper an extended framework for context-based addressing and routing, in order to efficiently support group addressing, service-discovery and data-centric communication in IoT networks. In this section we briefly introduce the general concepts behind the above services, and compare them with CAEsARv2, which is based on a different basic principle. In the followings we assume that the RPL routing protocol is used in the IoT domain.

A. Service discovery

If we have a networking infrastructure in place, enabling nodes to communicate with each other, then it is a reasonable assumption that individual nodes do not have to perform all the possible tasks and do not have to store all the possible data by themselves. Instead, they can rely on the services provided by other nodes. Not all the devices have to have thus a temperature sensor, or a GPS module, it is enough if they know how to query another device which can provide that service to them. However, in order to do so, they need first a way to discover the services that are available to them in a given moment, at a given location, through the devices that are in their radio range for example.

There are two main types of service-discovery protocols [6]: distributed and centralized. In a centralized approach one or more central registry maintain a list of services provided by the devices in the network. Any application or user that wants to use a service in the network has to turn one of these directories as an intermediary. If a change occurs in that service, updates should be sent to the registry. On the other hand, if another node want to discover the available services, it will query directly the registry. Obviously, the centralized registry might be a single point of failure, so alternative solutions either distribute the load of the registry into several sub-registries, building a hierarchical registry structure, or replicate the entire registry in several nodes. Both of these solutions have their own drawbacks. In the distributed case devices interact with each other directly to discover services without any coordinator entity. It can be done by using broadcast or multicast, it follows that this kind of solutions generate much more message overhead in the discovery phase, therefore in the typical resource-constrained LLN environment it is not a viable solution.

There were recently some registry-based service discovery solutions proposed specifically for IoT networks (e.g., TRENDY [9]). As opposed to these, our CAEsARv2 framework enables IoT nodes to store their context parameters, including their proposed services (e.g., the possession of a GPS module or the availability of moisture readings), in Bloom filters and bit vectors (as explained later). If another IoT device wants to find a specific service available in the area, its query will be forwarded rapidly, through a contextaware routing scheme, to nodes providing the desired service. No central registry needs to be used, and the signaling burden of maintaining aggregated context information will be much lower than the burden of regularly updating a central service registry.

B. Context-based group addressing

There will be several application scenarios in future IoT networks when user applications will need to communicate with a specific set of IoT devices, that have in common one or a set of context parameters (e.g., let's imagine a smart building scenario, where the user wants to close all windows in a given room, wants to turn off all the lights on a specific corridor, or wants to know the locations of the dustbins that are full).

There are two main approaches to support IP based group communication in the IoT domain. The first one is to maintain a registry in a device that has relatively more resources compared to other IoT devices (this is typically the RPL root, or a node situated in the wired part of the Internet). Every device that wants to join a multicast group registers itself in this registry. If someone wants to send a message to the group, it sends it to the registry, which then forwards it to the group members. This forwarding can be done either by sending a copy of the message to group member individually, or by using the so called explicit multicast [10], where the unicast IP address of every group member is added to the IP header of the message, before proceeding with traditional unicast routing. Supporting group communication by sending messages individually is a very inefficient solution, especially in the IoT domain with scarce resources. On the other hand, increasing the length of the messages by adding several destination IP addresses to the header is also problematic, as longer messages require more energy to be sent, and the chance of interferences or collisions is higher. Moreover, to keep up to date this central registry we mentioned before, IoT nodes would be required to send periodic keep-alive messages, which again consumes energy. In both of the above cases multicast group addresses are not needed.

The other possibility would be use traditional IP multicast [11] [12], routing packets along a multicast tree. Such an approach could be very beneficial for the resource-constrained IoT devices, since in this way messages are only duplicated where it is needed, at the branching points of the tree. However, while traditional multicast could be very efficient with a few but large groups, it does not scale well with lots of small groups, as multicast addresses cannot be aggregated. In practice we have to maintain as many spanning trees as many groups we want to handle. Thus, such a solution is not viable for our case, since we want to support a virtually "endless" number of groups, corresponding to all possible permutations of all possible context parameters.

However, if we consider IP addresses to be context parameters themselves, as explained later, and store them in Bloom filters as well, as all other parameters in the proposed CAEsAR framework, then traditional IP multicast and explicit multicast can be supported efficiently in the extended RPL domain, both regarding memory usage, message lengths and signaling overhead. No central registry is needed in this case.

C. Data-centric communication

In several IoT scenarios it can happen that different applications are interested in the same type of data, but for different purposes. In such cases it is very inefficient for the resource-constrained IoT devices to maintain several connections with these applications and send the same data several times. These situations can be overcome by using the data-centric communication paradigm in which we query the network for some specific content, no matter which device provides it. [13]. In this way the applications can focus on the data itself, rather than the process of getting it.

Publish/Subscribe messaging systems [8] are based on this networking principle and are considered to be potentially one of the best data collection protocols for IoT. Subscribers register their interest in specific information, the publishers provide such information, and the pub/sub system takes care of the information exchange. A pub/sub system can have centralized or distributed architecture [8]. In the latter case smart communication primitives (e.g. multicast) are used to ensure data exchange between the interacting parties. This typically puts a heavier burden on the participating nodes - compared to the centralized approach - since managing those primitives requires more processing power and/or more memory.

In the centralized approach an intermediary broker is used. The broker coordinates subscriptions, i.e., it ensures that data is collected from the publishers and is sent to the subscribers. Every time a change happens in a publisher's data, it has to publish it again through the broker. In a multi-hop network this means that a publish message has to be sent to the broker, possibly through multiple hops. MQTT-SN [14] is one such centralized publish-subscribe scheme. It is an extended version of the traditional MQTT protocol, and is designed to be as close as possible, in terms of operation, to the traditional solution, while being optimized for resource-constrained environments (the SN in its acronym comes from Sensor Networks).

Our proposed CAEsARv2 framework also can be considered as a centralized pub/sub system, since the measured values of the different environmental parameters are represented in an aggregated way at the RPL root in Bloom filters or in bit vectors (see section IV.). That means the RPL root has to manage pub/sub topics (e.g., the moisture readings in the given RPL domain) and subscriptions. Anytime a change happens in an aggregate BF or BV - because of the update messages - the root has to report this to the subscribers. In order to do that it needs to list the inserted elements in the newly created aggregate data structure. That means it has to query all the possible values of the actual parameter on the BF or on the BV. Since the RPL root has more resources, and querying a BF or a BV is a very light-weight process, this can be done easily even when the actual parameter is densely quantized.

If the users are interested in a more specific data, e.g., the temperature readings in a given room, then they can query all the temperature sensors in that room (through the very fast context-aware routing process in the CAEsARv2 framework). Alternatively, a special RPL objective function (OF) (explained later) could be used, which allows only those devices to connect to the RPL DODAG which are situated in that room.

III. CAEsAR IN A NUTSHELL

IN this section we briefly introduce first the RPL routing protocol on which CAEsAR is built, and then the CAEsAR addressing scheme itself. We also introduce the different types of context-parameters.

A. The RPL routing protocol

The RPL routing protocol [2] was designed especially for low power and lossy networks (LLN), and was standardized by the IETF in March 2012. It is a distance vector routing protocol that builds up a Destination Oriented Directed Acyclic Graph (DODAG) that has a single root. This root is the connection point, a gateway to other networks. A so called Objective Function (OF) defines the DODAG formation process, based on different metrics and constrains that have to be taken into account. There could be several OFs active in the same IoT domain; we call them RPL instances.

The RPL protocol defines new ICMPv6 control messages, such as DIO (DODAG Information Object), DIS (DODAG Information Solicitation) and DAO (DODAG Destination Advertisement Object). The DIOs carry information about the RPL instance and its configuration parameters; therefore, they are used for building up and maintaining the topology of the DODAG. A node can solicit DIO messages from other nodes by sending a DIS message. Finally, the DAOs are used to build up and maintain downward routes in the DODAG. Nodes inside the DODAG can operate either in storing-mode or non-storing mode. In non-storing mode, nodes do not store any routing entry; messages sent by any IoT device to any other Iot device are forwarded up to the root, and then directed downwards again to the destination, via source routing. In storing mode, the DAO messages sent by child nodes to their parents generate routing entries which can be used later to route packets inside the domain.

B. CAEsAR

In the CAEsAR [5] framework the context parameter values of the IoT nodes are hashed into Bloom filters (BF) [15] and aggregated upwards along the DODAG.

A Bloom filter is a space-efficient probabilistic data structure for representing a set of elements [16]. It is a bit array with a predefined length and hash functions. Each hash function hashes an element to a bit position. Inserting an element is done by hashing it with all the hash functions and setting to "1" all the resulting bit positions. Checking the membership of an element in a BF is achieved by the same hashing method.

False positives may occurs in BFs, as all the bit positions that correspond to a specific element may be set to "1" - by other elements - , even if this element does not belong to the set. This probabilistic nature is the price of space efficiency. If we choose the parameters properly, this probability can be kept reasonably low, so the space savings are often worth this tradeoff.

The question is thus how to store efficiently the context of an individual node, or the aggregated context of an entire sub-graph, in a BF. Context parameters can have continuous and discrete value ranges. (The measured temperature is an example for the former one, while the type of an IoT device is an example for the latter one.) For parameters with continuous value ranges a limited number of discrete intervals have to be set, to quantize their values. As a result, we can hash every context parameter easily into a BF. It is assumed that every node in the RPL tree uses a BF with predefined length and structure, and a set of predefined hash functions, specified and propagated along the tree by the root node. The RPL root has to specify these parameters according to the possible maximum number of context-parameters in the network and the maximum acceptable false positive rate in the aggregate BF(s) stored at the root node.

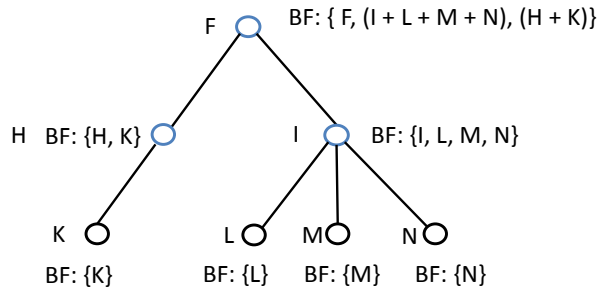


Fig. 1: Aggregating BFs - and the corresponding context parameter values - in a sub-graph of the DODAG

Bloom filters - with the same size and same hash functions - can be aggregated by performing a bitwise OR operation on them, which is a lightweight operation and suits well the resource-constrained IoT devices. In CAEsAR every node has to store a BF for representing its own context, and as many other BFs as many children it has, as it can be seen in Fig. 1. Every node in the RPL instance aggregates its stored BFs and sends this aggregate BF to its parent node, which stores this as the BF assigned to that particular child. If the topology of the DODAG is reconfigured, it may initiate new BF aggregation messages. Similarly, if a context parameter of a node changes (e.g., the measured temperature value), it has to re-create its own BF, aggregate all of its stored BFs, and if this currently created aggregate BF is different from the former one, this has to be sent to its parent node. The handling of the stored BFs in the RPL network is discussed in a more detailed way in our former paper [5].

The aggregate BFs can be used for two different purposes. On the one hand the aggregate BFs stored at the root represent the values of the context parameters that are available currently in the RPL domain. On the other hand the aggregate BFs inside the network can be used for context-based routing. This means that we can send a message to a device that has a specific context, without knowing its address. As each node inside the RPL tree aggregates in a separate BF the context parameters of each of its "child subtrees", it can be checked very rapidly if the desired context corresponds to any node included in a child subtree, or not. Then, the message will be forwarded only to the subtree(s) where a match was found. This process is then repeated until the message reaches one or more devices with the desired context. With this kind of routing we can support

service-discovery and context-based group communication in the RPL domain.

IV. CAEsAR 2.0

In this section we introduce several new design steps in order to extend CAEsAR and make it more efficient. First we make a suggestion to separate the different types of parameters and store them in distinct data structures. In this way, if the value of a parameter changes, only its assigned data structure needs to be propagated upwards in the RPL tree, not the entire context; thus, the update messages become shorter. We also make a suggestion to store IP addresses in Bloom filters as a context parameter, and examine its possible advantages.

A. Separation of static and dynamic context parameters

In the previous paragraph we differentiated context parameters based on whether their value ranges are continuous or discrete. Another way to differentiate them is based on whether they are static or dynamic. Static parameters do not change in time (e.g., the type of the device), while dynamic parameters do (e.g., the current temperature reading). In our original CAEsAR proposal every parameter was hashed to one "standard" BF and this BF was aggregated upward along the DODAG. This means that anytime a change happened in a context parameter of a device, it had to hash all of its context parameters into a new BF, calculate the new aggregated BF from all the BFs it stores (its own, and that of its child subtrees), and then initiate the sending of a BF update message upward along the DODAG, if needed. However, it can be more efficient if we could handle the different types of parameters in different ways. We suggest thus to separate the static parameters from every dynamic parameter, by assigning them individual BFs. . The length of the given BF should be set according to the expected number of elements in the BF and the maximum false positive rate at the root after the aggregation process. For a given false positive probability p and the number of inserted elements n the required number of bits m for a BF is [17]:

$$m = \frac{-n * ln(p)}{(ln(2))^2} \tag{1}$$

However, even though Bloom filters are a very space-efficient way of storing context information, the false positives that are due to their probabilistic nature are sometimes not acceptable. Another way to store context information would be to use bit vectors (BVs). Every bit position in a BV represents a context category. For example, if a node has a solar panel attached, the corresponding bit is set to 1; if not, it is set to 0. On the other hand, if the state of the battery is quantized into three intervals (low, middle, high), then three bit positions in the vector are allocated to represent the node's battery state, and only one of them could be set to 1. Using BVs is obviously more resource-hungry than using BFs, but there are no false positives. The problem is thus to decide when to use BFs and when to use BVs.

The number of bits m needed in a BV to store a given context parameter is equal to the number of elements in the complete value set of this parameter, n_{max}. This means the BF is more efficient than a BV if $n_{max} > \frac{-n_{inserted} * ln(p)}{(ln(2))^2}$. Let us examine what does this mean from our point of view.

Regarding the static parameters, assume that k nodes will join the RPL DODAG (that can be the maximum number of routing entries in the root).

- Let's call context parameters of *type 1* those that correspond to mutually exclusive choices, from which only one can be valid for a device at a given moment (e.g., the type of object). Regarding the values of such a parameter, we can say that at most k different values will be present in the network, out of the j possible values that form the complete value set of that parameter.
- Let's call context parameters of *type 2* those parameters that correspond to multiple non-exclusive choices. (e.g., what kind of sensors has a given device). This means that all the values of such a parameter can be represented in the network, and this is independent from the number of the currently connected devices to the RPL tree, since even only one device can possess all the possible values of such a parameter. Let's denote the number of all possible values of a given parameter as l.

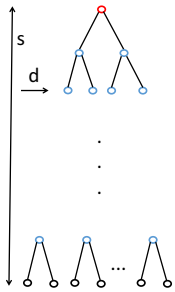
If we want to represent these parameters in a bit vector (BV), we need to set its length to: $m = l + j$, while for a BF it is: $m = \frac{-(l+k) * ln(p)}{(ln(2))^2}$. In a BV every additional element (i.e., a new context parameter, or a new value of an already included parameter) adds one more bit position to the length. In a BF every "inserted element" means $\frac{-ln(p)}{(ln(2))^2}$ additional bit positions. Therefore, as a novelty of the CAEsARv2 framework, we propose to store type 1 parameters in BF if $j > \frac{-k * ln(p)}{(ln(2))^2}$, and use a bit vector otherwise. We will store type 2 parameters always in BVs.

Dynamic parameters can be considered type 1 parameters. However, for a given parameter its possible values depend on its quantization density. Let's denote by q_i the quantization density of parameter i For example, if we represent battery state in three intervals - low, middle, high - then $q_i = 3$; if we quantize the temperature by Celsius degrees and readings can be values between 0 and 100, then $q_i = 100$). We propose to store the values of parameter i in a BF if: $q_i > \frac{-k * ln(p)}{(ln(2))^2}$.

To summarize our reasoning, we proposed to separate the static parameters on one hand, and every dynamic parameter on the other hand, where separation means to store them in distinct data structures. We gave conditions to decide which data structure should be used in which case, in order to decrease the needed memory and message length in the network. With this separation, when a dynamic parameter changes, only its assigned data structure needs to be propagated upward along the DODAG.

B. Handling IP addresses as context parameters

In this section we examine why could it be worth to handle IP addresses as context parameters as well, and why should we store them also in BFs. We examine the scaling of routing entries, the compressing of DAO messages, and the way this solution can be used for traditional- and explicit multicast. The BFs might include false positives, which in case of



f: number of children
s: depth of the tree (maximum hop number)
d: current hierarchy level

Number of routing entries	RPL	CAEsARv2
In a given node	$\sum_{i=1}^{s-d} f^i$	f, if $s \neq d$
In the whole network	$\sum_{d=0}^{s-1} f^d \sum_{i=1}^{s-d} f^i$	$\sum_{d=0}^{s-1} f^{d+1}$

Fig. 2: Scaling of routing entries

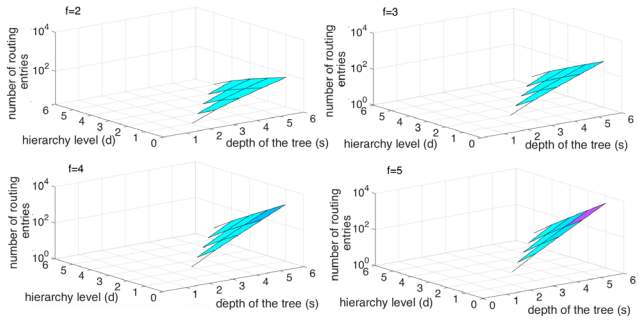


Fig. 3: Scaling of routing entries in a specific node in the RPL DODAG as the function of its hierarchy level d and the depth of the tree s, in case of an increasing number of children f and a full tree

stored IP addresses means that a packet is possibly forwarded downwards the DODAG - unnecessarily - even though the node with the required IP address is not located in that part of the tree. To decrease its possibility, we have to choose the BF parameters properly; however, if false positive appears, we can handle it by a so called false positive recovery mechanism used in ORPL [18], which is an opportunistic extension of the traditional RPL protocol. Now let's see the advantages of the IP addresses being handled as context parameters.

1) Scaling of routing entries: We can provide formulas to calculate the number of routing entries in a regular RPL DODAG for individual nodes at different hierarchy levels, as well as aggregate numbers for the whole network (fig 2). In this case we assume of course that nodes in the RPL DODAG are in storing mode, so they consume memory for these entries

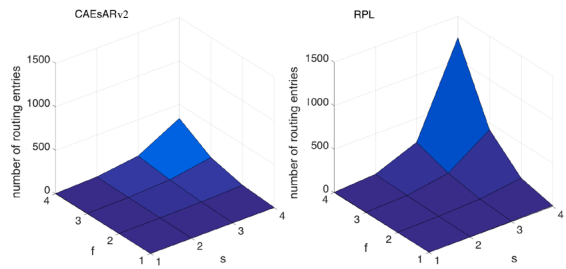


Fig. 4: Scaling of routing entries in the whole network as a function of depth of the tree "s" and the number of children "f"

number of routing entries	RPL	CAEsARv2
in a given node	$f * \frac{f^{s-d}-1}{f-1}$ $s-d$, if $f=1$	f, if $s \neq d$
in the whole network	$\frac{f^{s+1}*(s*f-s-1)+f}{(f-1)^2}$ $\frac{s^2+s}{2}$, if f=1	$\frac{f^{s+1}-f}{f-1}$ $s-1$, if f=1

TABLE I: Number of routing entries with closed formulas

but spare a lot of routing messages that are needed in case of the non-storing mode.

We can see that in RPL, as we increase the number of hierarchy levels, the number of routing entries in every node - except the ones at the bottom level - increases exponentially. This is due to the fact that in traditional RPL routing in the DODAG is not done based on the IP addresses of the nodes, but based on the routing tables built by DAO messages. Thus, separate entries should be stored in those tables for each node that has sent a DAO message; the hierarchical nature of the IP address space cannot be used to aggregate routing entries.

As opposed to this, in CAEsARv2, if IP addresses are stored in BFs as well, the same nodes have to store only f routing entries, that is the number of children they have in the DODAG. This is because all the IP addresses from the entire subtree are aggregated in the same BF at the parent node.

If we sum up these routing entries for every hierarchy level, then we get the total number of routing entries in the network for RPL and for CAEsARv2. Using the formulas on summing up geometric series we can get closed formulas:

- For RPL it is: $f * \frac{f^{s-d}-1}{f-1}$ ($s-d$,if f=1) for a specific node and $\frac{f^{s+1}*(s*f-s-1)+f}{(f-1)^2}$ ($\frac{s^2+s}{2}$, if f=1) for the whole network.
- For CAEsARv2 it is: f if $s \neq d$ for a specific node and $\frac{f^{s+1}-f}{f-1}$ ($s-1$,if f=1) for the whole network

These results are presented in figures 3 and 4. Please note that the z axis in fig. 3 is exponential. In fig. 4 we can see how the total number of entries in the network changes in function of the structure of the DODAG, in RPL and in CAEsARv2 respectively. For every parameter setting there will be fewer entries in the network if we use CAEsARv2.

The point here is that the total number of routing entries in the whole network scales better in CAEsARv2, and these entries are distributed evenly among the nodes (i.e., the nodes that are closer to the root - and have lots of nodes in their sub-DODAG - do not have to store much more entries than the nodes further away from the root. We should note however that depending on the used BF length, a routing entry in CAEsARv2 can be larger than a traditional routing entry.

2) Compressing DAO messages: Since in the other parts of the Internet traditional routing entries are used, we need the IP addresses from the RPL domain to be stored in their traditional format at the root, even if we store them in BFs in the RPL nodes. Therefore, we suggest that when a node joins for the first time an RPL DODAG, it should send a traditional DAO message that has to be propagated upward

until the root as it is. (The intermediate nodes insert this IP address in their proper BF.) From here on, this IP address can be propagated upward to the traditional Internet. As we saw in the former paragraph, storing IP addresses in BFs - similarly to context-parameters - can be beneficial in terms of the number of routing entries in the nodes and their even distribution in the network. Nevertheless, an IP address can be considered as a static parameter, we propose thus to assign a separate BF for it, since in this way we can support efficient explicit multicast (described later). This BF can be also used to compress DAO messages, since when a node has to send routing information about several routes, it can happen easily that sending its aggregate BF to its parent node is more efficient - regarding the message length - than sending several IP addresses in one or several DAO messages. (However, if the traditional DAO message is shorter, it also can be sent since the receiver nodes only have to insert them in the proper BFs.) To illustrate this, let us take a look at fig. 5. The false positive probability parameter p ($p \approx (1 - e^{(\frac{-k*n}{m})})^k$) [19]) can be seen in this figure as a function of the inserted elements n and the number of used hash functions k for a 32 bytes and a 64 bytes long BF(m). This means that if 32 byte long BFs are used in a network, then if we want to send routing information about more than two IP addresses, it is more efficient - regarding the message length - to send the corresponding BF.

3) Traditional- and explicit IP multicast: As we already mentioned, there are many cases when we would like to communicate not just with a single IoT device, but with a group of such devices that share some common context information for example. In RPL networks we can use the created DODAG to support multicast communication using the Stateless Multicast RPL Forwarding scheme (SMRF) [20] for example. In this solution the multicast addresses are advertised in a very similar way to unicast addresses; the only difference between them is that a multicast address can be assigned to several children in the routing table. If we store the IP addresses in BFs, then maintaining several multicast groups in the RPL domain does not mean additional routing entries in the nodes, as the multicast addresses corresponding to those groups can also be aggregated in the BF, together with the unicast addresses of a sub-tree. Explicit multicast [21] [22] was proposed as a solution to support network-layer group communication when there is no multicast support from the network service provider. It can be used when a source wants to send a message to several nodes, and their IP addresses are known in advance. This can be done by appending the individual unicast addresses to the IP header of the packet one-by-one. By doing so, we can achieve similar operation as for traditional IP multicast: the packet is duplicated only where it is needed. However, in LLNs, where the packet sizes are limited, this is not an efficient solution, as the header with all the included unicast addresses will be too large compared to the size of the payload itself. Nevertheless, if we use BFs to store IP addresses in the RPL domain, we can support more efficiently the explicit multicast operation in terms of message length. We only have to hash all the destination addresses into a single BF and append them to the message. The receiving

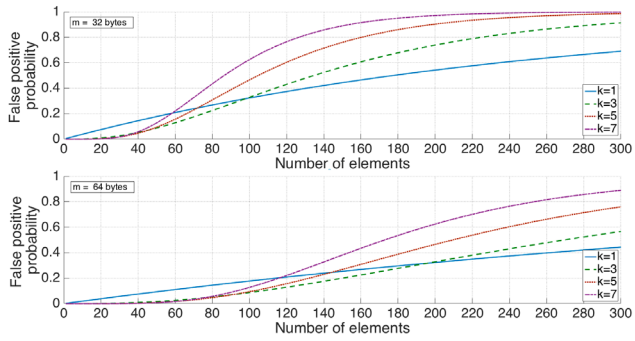


Fig. 5: The false positive probability in a BF as function of the number of inserted elements, the number of hashes k

nodes then compare this received BF with their stored BFs, and if any of them has an intersection, then this message should be forwarded to the proper sub-tree; if the BF contains the node's own address, than this message is intended to this node as well.

V. SIMULATION AND EXPERIMENTAL RESULTS

IN this section we analyze the efficiency of the proposed CAEsARv2 framework from several aspects, comparing it to traditional centralized approaches. Our previous paper [5] included already some simulation analysis, here we do not repeat, but extend those results and provide new insights.

First, we examine how CAEsARv2 is affected by the changes of the node radio ranges, and as a result the changes of the average hop numbers in the IoT domain. This is important because in the traditional centralized solutions if the average hop number increases in the network that means the messages between the nodes and the central registry (be that used for service discovery, data-centric communication or context-based multicast) have to travel over longer routes. Therefore, more messages have to be sent for instance to report a context change. As a second point, we have also examined how efficiently can CAEsARv2 adapt to context changes in real world circumstances, and not in a simulated environment. In order to do this, we ran experiments on the IoT Lab testbed [23].

A. Effect of the radio ranges on the efficiency of aggregation

In several cases there is a correlation between geographic proximity and the measured values of context parameters (e.g., temperature, light conditions, etc.). Therefore, there is a good chance in CAEsARv2 that if such a parameter changes, the corresponding BF or BV update messages coming from nearby IoT nodes will be aggregated. We already examined this phenomenon in our former paper [5]. However, it is an interesting question to see how is the aggregation efficiency affected by the node radio ranges, and as a consequence, by the hop numbers in the IoT domain. We examined this phenomenon through simulations, and not through experiments, since in this way we can ensure that the underlying correlation between the measured values and geographic proximity was the same in every case (and in every simulation).

In the simulations we used the so called "room heating up scenario" in the Cooja network simulator [24]. The setup

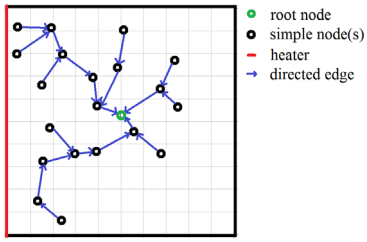


Fig. 6: An example setup for the room heating up simulation

is shown in fig. 6. One of the walls represented the body of the heater (x=0, y=0...100). We have put the RPL root node in the middle of the room and added nodes inside the room with random positions. The nodes have organized themselves into an RPL DODAG. We modelled that the room is heated by 10 degrees Celsius in every simulation. The temperature was quantized by 1 Celsius degrees, so any time the measured temperature value of a node changed to another Celsius degree value it sent an update message. At the beginning of this process the temperature was constant in every position of the room; after the heating was turned on, the temperature started to increase in different ways in the different locations, according to the proximity to the heater. Linear heating characteristics were used.

We ran simulations in the following way: we started with a simulation that contained 5 nodes in addition to the root, and ran it 5 times, with different radio ranges (tx_power=10, 15, 20, 25, 30). Then, we added randomly 5 more nodes to the simulation setup, paying attention to the fact that the newly added nodes should be able to connect to the DODAG even if the lowest radio ranges are used. With this new setup we also ran 5 different simulations with 5 different tx_power parameter settings. We continued this process until we reached 50 nodes in the simulations. (fig. 7) We considered this as being one iteration process, and we ran three such iterations.

We measured how the average hop numbers and the number of sent messages changed with the different parameter settings. The results can be seen in figures 8, 9 and 10 for one iteration. Every point in the figures represents the result of one simulation. (The tendencies were very similar in the other two iterations as well.)

Regarding the average hop numbers (fig. 8), we can see that as we decreased the tx_power parameter in the simulations, the

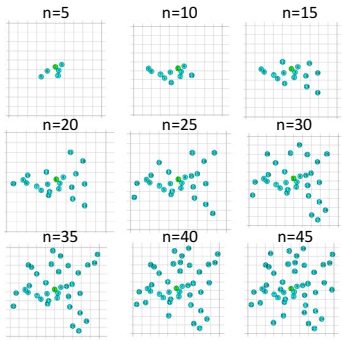


Fig. 7: The used topologies for one simulation iteration

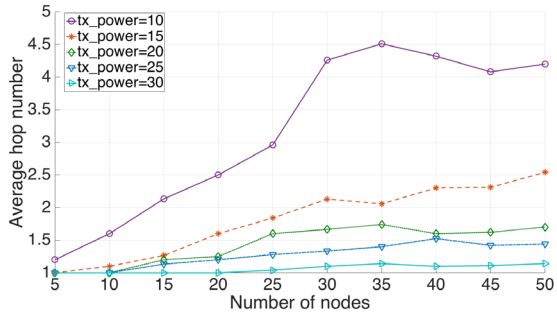


Fig. 8: The average hop number as a function of tx_power and number of nodes parameters

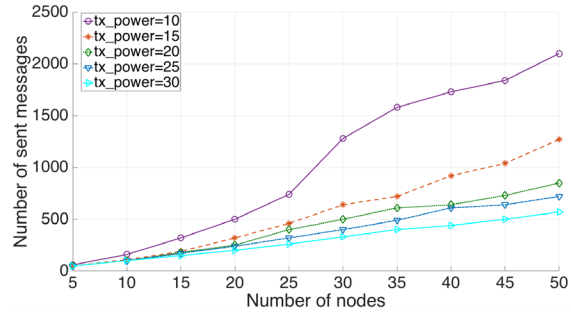


Fig. 9: The number of sent messages in the centralized approach as a function of tx_power and number of nodes parameters

hop numbers increased. Moreover, it seems that this increase is getting "leap-like" as we increase the number of nodes (especially for the lowest value of the *tx_power* parameter). This can be caused by the fact that with a larger node number it is more likely that more nodes get further away from the root, positioned in the middle of the area; with the decreasing radio ranges, they could not connect to the root with direct, short routes, but only along longer, roundabout routes. Also as the node number further increased the average hop number slightly decreased since some of the newly added nodes could be better (closer to the root) RPL parents for some of the nodes.

We can see in fig. 9 how many messages have to be sent if a centralized registry is used to maintain the state of the RPL nodes. We assumed here that this registry is co-located with the RPL root. If that registry is outside the IoT domain, then the route between the RPL root and the registry is constant,

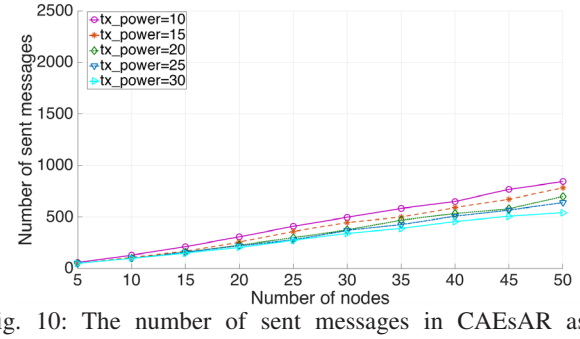


Fig. 10: The number of sent messages in CAEsAR as a function of tx_power and number of nodes parameters

and it is not affected by the node radio ranges or the hop numbers inside the IoT domain. Thus, it is interesting to see only what happens inside the domain. In the figure we can see that, as we heated up the room by 10 degrees Celsius, the number of messages was equal to 10 times the average hop number, multiplied by the node number. As this number depends linearly on the average hop number, we can also see here the "leap-like" increase with short radio ranges and with large node numbers.

Regarding the BF and BV update messages in CAEsARv2 (fig. 10), we can see that the number of sent messages depends mostly on the number of nodes in the simulations and not - or at least much less - on the average hop numbers. This means that CAEsARv2 is not affected very much by the longer routes in the RPL domain. Longer routes could appear nut just because of shorter radio ranges, but as a consequence of noisy communication channels as well, if the Minimum Rank with Hysteresis Objective Function (MRHOF) [25] is used to build the DODAG. This objective function optimizes routes according to the so called expected number of transmissions (ETX) [26].

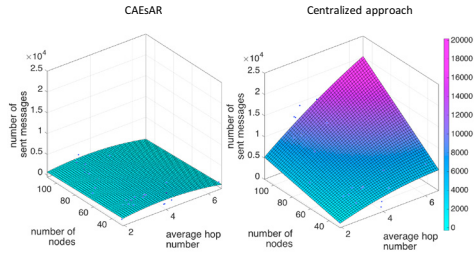


Fig. 11: Results of IoTlab temperature experiments

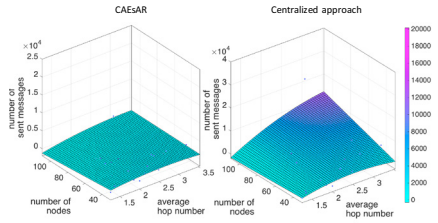


Fig. 12: Results of IoTlab light experiments

B. Experimental results

In order to validate how CAEsARv2 can adapt to context changes in real world circumstances, we implemented it for the IoTlab [23] version of ContikiOS [27], one of the most deployed operating systems for the IoT. IoT-LAB is a large scale IoT testbed in France with over 2700 wireless sensor nodes at six different sites. Nodes are either fixed or mobile and can be allocated in various topologies throughout all sites.

We had run 24 hour long experiments at two different sites of IoTlab: Lille and Grenoble. Regarding the Lille experiments, we chose a random number of nodes with random locations for every experiment. We chose one node as being the RPL root, and the RPL DODAG was built up from that node. After the DODAG built up phase has finished, the nodes started to measure periodically the light conditions. We quantized the measured values by 300 luxes. In the experiments the measured light conditions typically were between 0 and 4500 luxes. We hashed the categories into BFs and these BFs were aggregated along the DODAG. We measured the sent messages for CAEsARv2 and for the traditional centralized data-centric communication approaches (e.g., MQTT). We have also measured the number of category changes during the experiments. The results can be seen in fig. 12 and in fig. 13. We used polynomial surface fitting with degree 21, in order to make the figures more illustrative. The fitted surfaces are relatively plain (the points fit on them with little error), and show well the dependence of the signaling overhead on the average hop number and the number of nodes and also the dependence of the number of category changes on the average hop number and the number of nodes in the experiments. We chose the Lille site for this type of experiments since the testbed is surrounded there by windows; therefore, during the day we expected the light conditions to change a lot.

At the Grenoble testbed we did similar experiments; the only difference was that the nodes measured temperature periodically and we quantized the measured values by 1 degree Celsius. We chose this testbed for this case since the distances between nodes were larger, and it was expected thus that the measured temperature values will differ more. The results can be seen in fig. 11 and in fig. 14 with the similar surface fitting.

We ran approximately 60 experiments for both cases. As we see from the results, CAEsARv2 can utilize the correlation between the geographical proximity and the measured values in both cases, the aggregation was thus efficient. In several cases the number of sent messages was lower than the actual number of context category changes, and we can say that in general these two numbers were close to each other. To explain this, let us imagine a situation in which a device just sensed a category change. It hashes the new context category into the proper BF or BV, and aggregates all the data structures that are stored by this node and are assigned to that specific parameter. If the resulted aggregate BF or BV is the same as the former one that has been previously sent to the parent node, the node does not need to send it again. As opposed to this, obviously, in the centralized approach every context category change must be reported to the central registry.

VI. CONCLUSION

IN this paper we proposed CAEsARv2, an extension of our formerly proposed context-aware addressing and routing scheme for RPL networks. A major change compared to the original version was the separation of the different context parameters and the assignment of different data structures to them. We showed what are the benefits of storing IP addresses in Bloom filters, similarly to other context parameters. Through simulations we also proved that efficiency of Bloom filter aggregation in CAEsARv2 is not affected significantly by the radio ranges in the network. We have also validated through experiments that CAEsARv2 can adapt to context

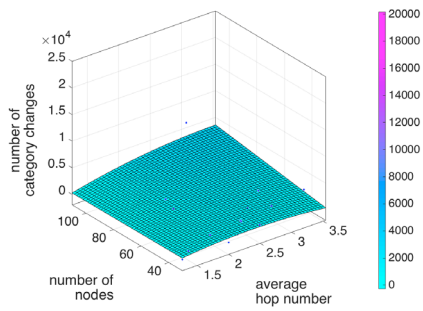


Fig. 13: The category changes in the different light measuring experiments taken at the IoTlab Lille testbed

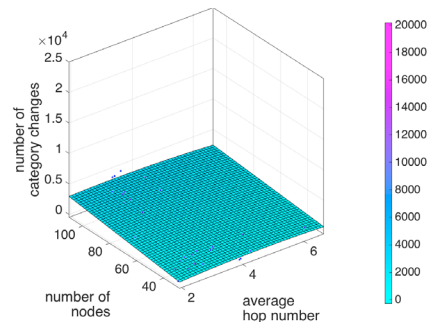


Fig. 14: The category changes in the different temperature measuring experiments taken at the IoTlab Grenoble testbed

changes more efficiently than the centralized publish/subscribe messaging systems if there is a correlation between geographical proximity and measured values.

ACKNOWLEDGMENT

The authors would like to thank Simon Duquennoy for his assistance with the Contiki implementation of the proposed method and his comments on the manuscript. We would also like to thank Chayan Sharkar for his comments that greatly improved the manuscript. We are also immensely grateful to the FIT IoTLAB staff for making it possible to run experiments on the FIT IoTLAB testbeds.

REFERENCES

[1] Z. Shelby and C. Bormann, *6LoWPAN: The wireless embedded Internet*. John Wiley & Sons, 2011, vol. 43.

[2] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550 (Proposed Standard), Mar. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6550.txt

[3] B. C. Villaverde, R. D. P. Alberola, A. J. Jara, S. Fedor, S. K. Das, and D. Pesch, "Service discovery protocols for constrained machine-to-machine communications," *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 41–60, 2014.

[4] A. Kalmar, R. Vida, and M. Maliosz, "Context-aware Addressing in the Internet of Things using Bloom Filters," in *Cognitive Infocommunications (CogInfoCom), 2013 IEEE 4th International Conference on*. IEEE, 2013, pp. 487–492.

[5] ——, "Caesar: A context-aware addressing and routing scheme for rpl networks," in *Communications (ICC), 2015 IEEE International Conference on*. IEEE, 2015, pp. 635–641.

[6] B. C. Villaverde, R. D. P. Alberola, A. J. Jara, S. Fedor, S. K. Das, and D. Pesch, "Service discovery protocols for constrained machine-to-machine communications," *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 41–60, 2014.

[7] S. Akkermans, R. Bachiller, N. Matthys, W. Joosen, D. Hughes *et al.*, "Towards efficient publish-subscribe middleware in the iot with ipv6 multicast," in *Communications (ICC), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–6.

[8] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys (CSUR)*, vol. 35, no. 2, pp. 114–131, 2003.

[9] T. A. Butt, I. Phillips, L. Guan, and G. Oikonomou, "Trendy: An adaptive and context-aware service discovery protocol for 6lowpans," in *Proceedings of the third international workshop on the web of things*. ACM, 2012, p. 2.

[10] R. Boivie, N. Feldman, Y. Imai, W. Livens, and D. Ooms, "Explicit multicast (xcast) concepts and options," Tech. Rep., 2007.

[11] R. Vida and L. Costa, "Multicast listener discovery version 2 (mldv2) for ipv6," Tech. Rep., 2004.

[12] G. Oikonomou and I. Phillips, "Stateless multicast forwarding with rpl in 6lowpan sensor networks," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*. IEEE, 2012, pp. 272–277.

[13] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. ACM, 1999, pp. 263–270.

[14] A. Stanford-Clark and H. L. Truong, "Mqtt for sensor networks (mqtt-s) protocol specification," *International Business Machines Corporation version*, vol. 1, 2008.

[15] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[16] ——, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[17] D. Starobinski, A. Trachtenberg, and S. Agarwal, "Efficient pda synchronization," *Mobile Computing, IEEE Transactions on*, vol. 2, no. 1, pp. 40–51, 2003.

[18] S. Duquennoy, O. Landsiedel, and T. Voigt, "Let the tree : Scalable opportunistic routing with orpl," in *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2013, p. 2.

[19] A. Broder and M. Mitzenmacher, "Network applications of filters: A survey," *Internet mathematics*, vol. 1, no. 4, pp. 485–509, 2004.

[20] G. Oikonomou, I. Phillips, and T. Tryfonas, "Ipv6 multicast forwarding in rpl-based wireless sensor networks," *Wireless personal communications*, vol. 73, no. 3, pp. 1089–1116, 2013.

[21] H. W. Holbrook and D. R. Cheriton, "Ip multicast channels: Express support for large-scale single-source applications," in *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 4. ACM, 1999, pp. 65–78.

[22] R. Boivie, N. Feldman, Y. Imai, W. Livens, and D. Ooms, "Explicit multicast (xcast) concepts and options," Tech. Rep., 2007.

[23] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele *et al.*, "Fit iot-lab: A large scale open experimental iot testbed," in *Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on*. IEEE, 2015, pp. 459–464.

[24] J. Eriksson, F. Österlind, N. Finne, N. Tsiftes, A. Dunkels, T. Voigt, R. Sauter, and P. J. Marrón, "Cooja/mspsim: interoperability testing for wireless sensor networks," in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009, p. 27.

[25] O.Gnawali, P. Levis, "The Minimum Rank with Hysteresis Objective Function," RFC 6719 (Proposed Standard), Sep. 2012. [Online]. Available: https://tools.ietf.org/rfc/rfc6719.txt

[26] D. S. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, no. 4, pp. 419–434, 2005.

[27] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*. IEEE, 2004, pp. 455–462.

Andras Kalmar currently is a visiting researcher at RISE SICS in Stockholm in the Network Embedded Systems Group (NES). He received his M.Sc. in Electrical Engineering at the Budapest University of Technology and Economics (BME) in 2012. Besides his traditional PhD education he is also part of the doctoral program of EIT Digital, in which the doctoral candidates are offered the opportunity to acquire a mindset for Innovation and Entrepreneurship (I&E). His main research areas are group communication and service-discovery aspects of IoT. He also has papers on how to realize context-aware services in IoT networks using machine learning methods.

Rolland Vida Rolland Vida is an Associate Professor at Budapest University of Technology and Economics since 2007. He obtained his MSc in Computer Science at Babes Bolyai University Cluj Napoca, Romania, in 1997, and his PhD in Computer Networks at Université Pierre et Marie Curie, Paris, France in 2003. He was a Bolyai Research Fellow of the Hungarian Academy of Sciences between 2007 and 2010. His research interests are in wireless sensor networks, vehicular networks, smart cities, peer-to-peer networks and multicast communications. He has published more than 80 papers in scientific journals and conference proceedings, for which he has more than 1300 citations. Rolland Vida is involved in different administrative committees of the IEEE Communications Society and the IEEE Sensors Council. He is member of the Steering Committee of the IEEE Internet of Things Journal.