Using RELOAD and CoAP for Wide Area Machine-to-Machine Communication

Jouni Mäenpää and Jaime Jiménez Bolonio
Ericsson Finland
{jouni.maenpaa, jaime.jimenez}@ericsson.com

Abstract—In this paper, we propose a new architecture for wide area Machine-to-Machine (M2M) networks. The architecture is based on combining two protocols that are being specified by the Internet Engineering Task Force (IETF): REsource LOcation And Discovery (RELOAD) and Constrained Application Protocol (CoAP). To integrate CoAP and RELOAD, we introduce a CoAP application usage for RELOAD. The architecture provides a non-infrastructure based peer-to-peer rendezvous service for CoAP nodes in Wireless Sensor Networks (WSNs). Our architecture also enables a peer-to-peer federation of geographically distributed WSNs. This is supported by proxy nodes that are part of the WSN but also connect to a RELOAD overlay network via cellular Internet access. Other features of the architecture include integration to web, self-organization, scalability, and robustness. We evaluate the proposed architecture through simulations and compare its performance to a traditional client/server architecture.

I. INTRODUCTION

The rapid increase in the number of IP-enabled embedded devices is giving rise to the Internet of Things. The vision behind the Internet of Things is that everyday objects (e.g., sensors, actuators, consumer electronics, and industrial devices) become interconnected, IP-addressable, and an integral part of the services in Internet. According to some estimates, the number of connected devices grows to more than 50 billion in the next ten years [1]. Thus, there is a need for network architectures that can support the exponential growth in the number of devices. The solutions are likely to benefit from being open and standards-based to support interoperability between various ecosystems and vertical industries.

One main factor enabling the Internet of Things is the cost of wireless modules. Technological advances, broad deployment and economies of scale of 3GPP and 3GPP2 standards make them affordable and attractive for many applications and competitive compared to other technologies, especially when there is a need for wide-area connectivity. It is foreseen that the geographical coverage and flexibility of cellular mobile connections will make them the dominant technology for connecting things to the Internet [2].

Although cellular technologies are attractive for wide-area M2M communication, constrained Wireless Sensor Networks (WSNs) such as Low-Power wireless Area Networks (LoWPANs) are better served by short-range radio technologies like IEEE 802.15.4, which specifies the physical and media access control layer for low-rate LoWPANs.

In this paper, we propose a new architecture for device-to-device, that is, Machine-to-Machine (M2M) communication. The architecture provides an infrastructureless federation of geographically distributed WSN islands. Nodes equipped with both cellular and WSN radio interfaces sit at the edge of the WSNs and participate in a Peer-to-Peer (P2P) overlay network that provides a common namespace, rendezvous, and other services. The solution is based on open Internet standards. The key enabling protocols are CoAP and RELOAD that are currently being specified by the Internet Engineering Task Force (IETF).

The paper is structured as follows. Section I introduces the main technologies that our architecture utilizes. Section II presents related work. Section III lists the requirements of the use cases for which our architecture is targeted. Section IV describes the architecture. Section V describes the setup of the simulations that we used to evaluate the architecture. Section VI presents the results of the simulations. Section VII concludes the paper.

A. Constrained Application Protocol (CoAP)

CoAP [3] is a specialized web transfer protocol. It realizes the Representational State Transfer (REST) architecture for the most constrained nodes. CoAP can be used not only between nodes on the same constrained network but also between constrained devices and nodes on the Internet. The latter is possible since CoAP can be translated to HTTP for integration with the web. CoAP can also be used between devices in different constrained networks interconnected by an internet, which makes it very suitable for our architecture that federates separate constrained networks. Application areas of CoAP include different forms of M2M communication. CoAP provides a request/response interaction model between application endpoints, supports built-in resource discovery, and includes key web concepts such as URIs and content-types. CoAP uses unreliable datagram-oriented transport (i.e., UDP). CoAP meets the specialized requirements of constrained environments such as low overhead, simplicity, and ability to deal with sleeping nodes. The main example of operating environments CoAP targets is 6LoWPANs (IPv6 over LoW Power wireless Area Networks). However, CoAP also operates over traditional IP networks. CoAP has been extended to enable clients to establish observation relationships between themselves and resources [4]. CoAP targets constrained networks such as IPv6 over LoW Power wireless Area Networks (6LoWPANs).
B. IPv6 over LoW Power wireless Area Networks (6LoWPAN)

A low-power wireless personal area network (LoWPAN) is a simple low-cost communication network that allows wireless connectivity in applications with limited power and relaxed throughput requirements. Devices in a LoWPAN conform to the IEEE 802.15.4 standard. IEEE 802.15.4 devices typically have short range, low bit rate, low power, and low cost. Further, the devices are limited in their computational power, memory, and/or energy availability. The IPv6 over LoW Power wireless Area Networks (6LoWPAN) working group of the IETF has defined encapsulation and header compression mechanisms that allow IPv6 packets to be sent and received over LoWPANs [5]. 6LoWPAN networks often have high packet error rates and a typical throughput of tens of kilobits per second.

C. REsource LOcation And Discovery (RELOAD)

REsource LOcation And Discovery (RELOAD) [6] is a P2P signaling protocol that is being specified by the Peer-to-Peer Session Initiation Protocol (P2PSIP) working group of the IETF. RELOAD provides a generic, self-organizing P2P overlay network service. Nodes can use the overlay to route messages to other nodes and to store and retrieve data. RELOAD uses the Chord Distributed Hash Table (DHT) algorithm [7] as the default algorithm to organize the overlay. For Network Address Translator (NAT) traversal, RELOAD uses Interactive Connectivity Establishment (ICE) [8]. RELOAD supports two types of nodes: peers and clients. Clients are nodes that run the DHT algorithm and route messages and store data on behalf of other nodes. Clients are nodes that do not run the DHT algorithm and do not provide message routing and storage services. Instead, they use the services of the overlay by connecting to a peer. New applications can utilize RELOAD by defining new usages. Our architecture makes use of a new CoAP usage for RELOAD. This usage allows a RELOAD overlay to be used as a distributed rendezvous, storage, and NAT traversal service for CoAP endpoints.

D. Interactive Connectivity Establishment (ICE)

As discussed above, RELOAD uses ICE as a technique for NAT traversal. ICE uses the Session Traversal Utilities for NAT (STUN) [9] protocol and its extension, Traversal Using Relays around NAT (TURN) [10]. STUN is used by a host to determine the IP address and port allocated to it by a NAT, to test connectivity between two hosts, and as a keep-alive protocol to maintain NAT bindings. TURN is used in situations when two hosts are unable to communicate without the help of a relay. TURN allows a host to control the relay and to exchange packets with its peers using the relay.

III. REQUIREMENTS

In this section, we will describe the requirements of the use cases for which our architecture is targeted. First, the devices interconnected by the architecture require wide-area geographical coverage. For this, they use cellular mobile connections. Examples of such devices include sensors and actuators equipped with 2G or 3G modules or gateway nodes with dual interfaces: a short-range WSN radio interface and a 2G or 3G interface. The devices are deployed in the wide area; distances between the devices are on the order of hundreds of meters or kilometers.

The systems require self-x properties, that is, they need to be self-configuring, self-organizing, self-optimizing, self-adjusting, and self-reliant. As an example, the system may use data from sensors to trigger decisions in actuators in an autonomous manner without the involvement of central application servers. New devices can be added to the system in a plug-and-play (zero configuration) fashion. The system should also have low capital expenditures (that is, low investment in central servers and data centers) and operating expenditures (e.g., low involvement from maintenance personnel).

The system should scale from simple use cases to use cases in which data from the devices is real-time and the volume of it high. Further, there may be a high number of devices. Thus, the system should be scalable; even if a large number of new nodes are added, this should not require investment in new capacity in central servers or data centers. Since cellular networks may assign private IPv4 addresses to the devices [17], the system should support NAT traversal in a scalable manner. Due to the above-mentioned factors, the system benefits from a great degree of decentralization to reduce the load of central components such as servers and data centers.

The devices interconnected by the architecture are heterogeneous. Some of the devices are very constrained sensors utilizing WSN radio technologies, whereas other devices use also cellular technologies and have at least a moderate amount of CPU and RAM. There are also nodes having both a WSN radio interface and a cellular interface.

All the devices, even the most constrained sensors have IP connectivity, which is provided by technologies such as 6LoWPAN. However, non-IP based WSNs (e.g., ZigBee) can join the
system through gateway nodes. Finally, the resources hosted on the devices should be accessible from web applications.

IV. ARCHITECTURE

Our wide-area M2M communication architecture that integrates the CoAP and RELOAD protocols is illustrated in Figure 1 and Figure 2. In Figure 1, all of the nodes (i.e., sensors and actuators) are geographically distributed and thus the system uses only cellular technologies for communication. We call such nodes using only cellular access Wide area Nodes (WNs). All WNs act as peers in a RELOAD overlay network. In Figure 2, there are three types of nodes: WNs, Proxy Nodes (PNs), and Local area Nodes (LNs). LNs are part of a WSN, which may be for instance a 6LoWPAN or a ZigBee network. LNs connect to the RELOAD overlay network through a PN, which is a special node having two radio interfaces: a cellular interface and a IEEE 802.15.4 radio interface. Other types of nodes in Figure 1 and Figure 2 include Gateway Nodes (GWs), and Monitoring and Controlling Nodes (MCNs). We will go through the different types of nodes in more detail in the subsections below.

A. Local Nodes

Local Nodes (LNs) are constrained devices such as sensors and actuators with limited resources (CPU, RAM, battery, etc.). LNs are part of a 6LoWPAN and thus have IPv6 connectivity. The LNs act directly as CoAP endpoints. They host one or more resources that need to be discoverable in the RELOAD overlay. However, since LNs are constrained, and since the 6LoWPAN application throughput may be low and payload size small, LNs are assumed not to be capable of acting as RELOAD clients. In other words, the LNs cannot use RELOAD directly to register their resources in the overlay. Instead, the PN takes care of registering the resources hosted by the LNs in the RELOAD overlay. All the application-level signaling between the LNs and PNs utilizes CoAP.

B. Non-IP LNs

Also non-IP connected sensors and actuators can become addressable in the RELOAD overlay network. In this scenario, the PN acts in a similar manner as in the case with IPv6-enabled LNs. The difference is of course that the signaling on the WSN side does not utilize CoAP/UDP/IP. In this scenario, the PN, having discovered LNs in the WSN, assigns each LN a RELOAD node-ID, and for each resource hosted by the LN, a CoAP URI and a RELOAD resource-ID. The LN maintains a mapping between the WSN-specific IDs of the LNs and the RELOAD node-IDs and between the WSN-specific resource IDs and CoAP URIs. When receiving CoAP methods addressed to the LNs, the PN performs L1-L3 translation from CoAP/UDP/IP to the WSN-specific protocols.

C. Wide Area Nodes

Wide area Nodes (WNs) are devices using a cellular technology such as 2G or 3G modules to connect to the Internet. WNs do not belong to a 6LoWPAN and do not have a WSN radio (e.g., IEEE 802.15.4) interface. WNs participate as peers in the RELOAD overlay network. Since WNs act as RELOAD peers, they run the DHT algorithm and provide message routing and storage services to other peers in the overlay. Like LNs, also the WNs act as CoAP endpoints and host resources that can be accessed using CoAP.

D. Proxy Nodes

Proxy Nodes (PNs) are located at the edge of a WSN. In the rest of this subsection, we will assume that the WSN is a 6LoWPAN, although non-IP LNs can be supported also as described in Section IV-B. They PNs act as gateways between the 6LoWPAN and the Internet. Each 6LoWPAN is assumed to have its own domain in CoAP URIs to distinguish it from other 6LoWPANs. PNs use IEEE 802.15.4 on the 6LowPAN side and connect to the Internet using a cellular radio interface. Like WNs, also PNs act as RELOAD peers and thus participate in the RELOAD overlay. The difference between WNs and PNs is that since PNs have an interface towards a 6LoWPAN, they can connect LNs to the RELOAD overlay network. Since a PN sits at the boundary of the 6LoWPAN and the Internet, it acts as a 6LoWPAN Border Router (6LBR) [18]. The PNs
also act as CoAP endpoints and as CoAP proxies for CoAP nodes in the 6LoWPAN. As specified in [3], CoAP clients learn the address of the CoAP proxy via local configuration.

As was discussed above, all application-signaling between LNs and a PNs is CoAP. The PN takes care of registering the resources of LNs in the RELOAD overlay on behalf of the LNs. The way this is done depends on whether the LNs are aware of the RELOAD overlay.

When LNs are aware of the RELOAD overlay, they use CoAP to interact with the overlay. CoAP is used instead of the RELOAD client protocol since the latter is not suitable for the most constrained nodes and networks. When an LN is added into the 6LoWPAN, it needs to register its resources in the RELOAD overlay so that the resources become discoverable by other nodes. The registration procedure is illustrated in Figure 3. In step 1, the LN uses a CoAP PUT method addressed to the PN to initiate the registration procedure. In our implementation, the path component of the CoAP URI is set to a special value /reload-overlay to indicate to the PN that the LN wishes to "join" the RELOAD overlay. The purpose of the CoAP PUT is to store the following mapping into the overlay: \( \text{hash}(\text{CoAP-URI}), \text{destination list} \), where \( \text{hash}(X) \) is a SHA-1 hash over \( X \), and destination list contains the Node-IDs through which the messages destined to the LN need to be routed. The URI of the resource being registered is carried in the Proxy-Uri CoAP option. In step 2, the PN uses this URI to create a RELOAD resource-ID for the LN's record in the overlay by calculating a SHA-1 hash over the URI. The PN also either creates a RELOAD node-ID for the LN or, as in our implementation, uses the node-ID included by the client as a CoAP option to the CoAP PUT. We assume no special relationship between the LN's node-ID and the PN's node-ID, such as the PN being responsible for the LN's node-ID in the overlay. Instead, the CoAP usage for RELOAD stores a RELOAD destination list [6] including the PN's node-ID and the LN's node-ID in the overlay within the LN's resource record. This way other nodes in the overlay are able to reach the LN.

After the PN has created the node-ID, resource-ID, and destination list, it consults its local routing table and sends a RELOAD Store request to the appropriate peer in the routing table. The Store is routed across the overlay to the peer responsible for the resource-ID. In step 3, a RELOAD 200 OK reply to the Store is routed back to the PN. In step 4, the PN sends a CoAP ACK with an immediate response back to the LN. Note that all RELOAD hop-by-hop ACK messages have been omitted from the figure for brevity.

When LNs are unaware of the RELOAD overlay, they do not explicitly register their resources with the overlay. Instead, a PN discovers the LNs belonging to the same 6LoWPAN for instance by periodically broadcasting a CoAP GET on the well-known resource URI (i.e., /.well-known/core) defined in [19]. The responses the PN receives contain URIs for the resources hosted by each CoAP node. For each URI, the PN stores a CoAP resource record containing among other things a \( \langle \text{hash(CoAP-URI)}, \text{destination list} \rangle \) mapping into the RELOAD overlay. The resource-IDs and node-IDs are generated in the same way as above.

When a PN receives a CoAP DELETE method carrying the special path component /reload-overlay on the CoAP URI, it removes the specified resource from the RELOAD overlay. Naturally, only CoAP nodes that are aware of the RELOAD overlay send CoAP DELETEs.

On the cellular network side, the PN may be located behind a Network Address Translator (NAT). Therefore, all the connections to and from the PN, including those established for communication between the LN and the outside world, need to be negotiated using ICE. Figure 4 shows the messaging associated with establishing a CoAP observation relationship between two LNs in different 6LoWPANs, including the ICE negotiation phase. In the figure, LN-A in 6LoWPAN-A starts observing a resource hosted by LN-B in 6LoWPAN-B. In step 1, LN-A sends a CoAP GET to PN-A. The GET contains the request URI in a CoAP Proxy-Uri option. PN-A inspects the domain in the request URI. Since the domain is different from the domain of 6LoWPAN-A, PN-A performs a RELOAD lookup operation (Fetch request) in the RELOAD overlay to learn the destination list of LN-B. The Fetch request is routed across the overlay to the peer responsible for the resource-ID. The Node-ID is returned in step 3 inside a RELOAD Fetch 200 OK response. In step 4, PN-A sends a RELOAD Attach across the overlay towards the first node (the node-ID of PN-B) on the destination list. The Attach carries PN-A's ICE candidate addresses. The Attach is routed to PN-B. In steps 6-7, a RELOAD Attach 200 OK response with PN-B's ICE candidates is routed back to PN-A. In step 8, PN-A and PN-B perform ICE connectivity checks to establish a direct UDP connection for CoAP. Once the connectivity checks are over, PN-A sends the CoAP GET to PN-B in step 9 on the ICE-negotiated connection. PN-A is unaware that PN-B is a CoAP reverse proxy (a CoAP reverse proxy is a proxy that receives requests as if it was the origin server for the target resource). In step 10, PN-B, acting as a CoAP reverse proxy, sends the CoAP GET to LN-B. The GET establishes an observation relationship [4] between LN-A and the resource on LN-B. In steps 11-13, a CoAP ACK carrying an immediate response is sent from LN-B to LN-A. In the future, whenever the state of the resource on LN-B changes, LN-B will notify LN-A by sending a new CoAP response.

In some cases, such as when the frequency of CoAP
notifications associated with the observation relationships is very low, it may be too expensive to establish dedicated ICE-negotiated connections for CoAP. In such cases, CoAP messages can be sent tunneled (i.e., encapsulated in the payload of RELOAD messages) across the overlay. For this, we are using a new RELOAD request which we call the Tunnel request. The use of Tunnel requests is illustrated in Figure 5. In the figure, LN-A in 6LoWPAN-A wants to access a resource hosted by LN-B in 6LoWPAN-B without establishing an observation relationship. Steps 1-3 in the figure are identical to Figure 4. In step 4, having fetched the destination list of LN-B from the overlay, PN-A, instead of establishing an ICE-negotiated connection to PN-B, places the CoAP CON GET message in the payload of a RELOAD Tunnel request and routes the message towards PN-B, which is the first entry on the destination list. Having received the Tunnel request in step 5, PN-B forwards the GET message to LN-B in step 6 since LN-B happens to be awake. In step 7, PN-B receives a CoAP ACK carrying an immediate response from LN-B. PN-B places the CoAP ACK in the payload of the RELOAD Tunnel response and routes it back to PN-A across the overlay in step 8. PN-A forwards the CoAP ACK to LN-A in step 9.

If LN-B was sleeping when PN-B receives the encapsulated CoAP GET in step 5 of Figure 5, PN-B could either return cached information in the Tunnel response, or if no cached information was available, a CoAP ACK without an immediate response. When LN-B becomes available to produce a response, the response would be Tunneled across the overlay to PN-A inside a new RELOAD Tunnel request.

It is worth noting that although no observation relationship was established in Figure 5, the tunneling mechanism could be used even for observation relationships. In this case, the first tunneled CoAP method established the relationship. The actual notifications are sent in the payload of RELOAD Tunnel requests from PN-B to PN-A.

E. Gateway Nodes

A Gateway Node (GW) acts as a peer in the RELOAD overlay network. In addition, the GW acts as a HTTP/CoAP proxy [20]. The purpose of a HTTP/CoAP proxy is to provide interoperability between HTTP and CoAP. Thus, as a HTTP/CoAP proxy, the GW provides web applications access to the resources in the WSNs interconnected by the RELOAD overlay network. It also enables CoAP clients to access resources on web servers.

F. Monitoring and Control Nodes

Monitoring and Control Nodes (MCNs) are HTTP clients that use the CoAP REST interface to access resources in the WSNs federated by the RELOAD overlay network. A GW node acts, as a HTTP/CoAP proxy, as the first point of contact for MCNs in the CoAP+RELOAD network.

Figure 6 shows an example in which an MCN starts to observe a resource in a 6LoWPAN. In step 1 the MCN issues, using HTTP long polling [21], a hanging HTTP GET request to the GW node. Having received the HTTP GET, the GW, which is acting as both a HTTP/CoAP proxy and a peer in the RELOAD overlay, sends in step 2 a RELOAD Fetch (i.e., lookup) to the RELOAD overlay to learn the destination list of the LN hosting the resource specified in the HTTP GET. The Fetch is routed towards the node responsible for the resource-ID created by hashing the CoAP URI of the resource. In step 3, the destination list of the LN is returned in a RELOAD Fetch 200 OK response. In steps 4-5, a RELOAD Attach request is routed across the overlay to the PN behind which the LN is located (the PN is the first entry on the destination list). The purpose of the Attach request is to establish a direct UDP connection between the GW and PN across NATs using ICE. The PN returns its ICE candidates in a RELOAD Attach 200 OK response in steps 6-7. The ICE connectivity checks are performed in step 8. In step 9, ICE has established a direct UDP connection between the GW and the PN. The GW sends a CoAP GET to the PN. In step 10, the PN, acting as a CoAP reverse proxy, sends the CoAP GET to the LN in the 6LoWPAN. The LN happens to be awake and returns an immediate response in a CoAP ACK in steps 11-12.
Our architecture has several advantages. First, the architecture provides a federation of WSNs. CoAP is used to provide a common namespace for resources in all interconnected WSNs. A sensor in one WSN can access the resources of a sensor in another WSN since the WSNs are interconnected by a RELOAD overlay.

The architecture enables infrastructureless CoAP clouds. RELOAD is used to provide a lookup, storage, message routing, and NAT traversal service. RELOAD is used to map CoAP URIs to contact information of the sensors. In particular, since RELOAD is used, the system does not need to relay on infrastructure-based services like DNS-based service discovery (DNS-SD) [22] that is used by CoAP usages like the one specified in [23].

The architecture also integrates the WSNs to web. Due to the use of CoAP and GW nodes, web applications can access resources in the WSNs federated by the RELOAD overlay. CoAP clients can also access resources on web servers. In general, web integration of objects enables interesting new applications such as the Web of Things [24]. The architecture also makes resources in non-IP-based WSNs accessible through CoAP. This is enabled by PNs that terminate CoAP signaling and translate that to the protocols used in the non-IP-based WSNs.

Due to the use of P2P technologies (i.e., RELOAD), the architecture is scalable, robust, and cost-efficient. It is scalable since each additional PN added brings extra resources to the system, unlike in a client/server system where each new PN consumes additional resources on the central servers and eventually more capacity needs to be added. The system is also scalable when it comes to NAT traversal since in a P2P architecture, the peers can act as STUN and TURN servers to each other. However, in a C/S system, centralized TURN servers are necessary. In a real-world system, a subset of the nodes will typically be behind the most restrictive types of NATs. When two such nodes need to communicate with each other, all the traffic between them needs to be relayed by a TURN server. If the volume of data is high and the system large, the relay servers need to have high capacity. However, in a P2P system, the responsibility for relaying data can be distributed among publically reachable peers. The architecture is robust since it is not dependent on centralized elements (i.e., central points of failure) for rendezvous and relaying of data. The system is cost-efficient since it has both low capital expenditure and perhaps more importantly, low operational expenditure.

V. SIMULATIONS AND USE CASE

In this section, we will evaluate the proposed architecture through simulations. We focus on an example use case.

A. Simulator

The simulations were run using our P2P simulator, which is an event-driven, message-level simulator. It uses the same code base as our P2PSIP prototype that we have used to run experiments in PlanetLab in previous work [25], [26], [16]. We have previously used the simulator in [27]. We chose to use the simulator also in this paper since we wanted to experiment with large-scale overlays created by only nodes using cellular radio access (that is, the PNs). The simulator is implemented in the Java programming language. It uses a predecessor of the RELOAD protocol called the Peer-to-Peer Protocol (P2PP) [28] as the protocol between peers in the overlay. The RELOAD protocol [6], which is currently being standardized in the IETF, is based on P2PP (the P2PP proposal was merged with RELOAD during the IETF standardization process). P2PP/RELOAD connections are assumed to run over an unreliable transport. The Chord DHT [7] is used to organize the overlay. Chord was chosen since RELOAD specifies it as mandatory to implement [6].

The delay generator of the simulator uses a large set of delays we collected by measuring delays in a 3G High Speed Downlink Packet Access (HSDPA) network. We measured delays between mobile devices, and between mobile devices and a central server. We used a large number of packet sizes ranging from 10 bytes to 1400 bytes. The delay set was further extended by using results from our previous work of measuring ICE and RELOAD performance in mobile networks [16], [15].

For this paper, we extended the simulator by adding support for simulated LNs in WSNs that connect to the RELOAD overlay through a PN. We also implemented the CoAP protocol and integrated it to the simulator. All the simulated PNs use ICE to establish connections for RELOAD and CoAP.

B. Use Case

In the first set of simulations, we focus on a use case in which there are 2000 PNs. Each PN sits at the edge of a
WSN having 10 LNs, meaning that in total, there are 20000 LNs. There are 10 outgoing CoAP observation relationships from each WSN, which could mean for instance that each WSN observes all the LNs in a neighboring WSN or PNs of 10 other WSNs. Nodes notify their observers on the average every 10 minutes.

One specific use case having the above-mentioned characteristics could be a road traffic and road condition monitoring use case. As an example, the Finnish highway network has close to 10000 kilometers of road. If monitoring units consisting of equipment to monitor traffic and weather conditions are deployed on highways on the average every five kilometers, the resulting system has 2000 monitoring units acting as PNs. The PNs are equipped with 3G HSDPA modules. The monitoring units are coupled with variable speed limit displays and warning displays that are used to show information to users of the road. Every monitoring unit acts as a PN for 10 wireless sensors (LNs) in a local WSN. The sensors are used for monitoring various conditions such as road surface temperature (to detect ice formation), air temperature, wind speed, wind direction, humidity, light, the amount of chemicals applied for prevention of freezing of water, water layer thickness, black ice, traffic volume, speed of vehicles (e.g., using a radar), and accidents (e.g., using video cameras coupled with computer vision). The data from the sensors can be used to adjust the current speed limit and show warnings to the users of the road on the warning displays. We assume that a given monitoring unit uses not only local information but information from sensors in nearby WSNs as input for determining an appropriate speed limit to optimize the flow of traffic. For this, every monitoring unit is assumed to have on the average ten outgoing CoAP observation relationships.

In the simulations, we compare the performance of a RELOAD-based P2P architecture to a traditional Client/Server (C/S) architecture. In the C/S simulations, the overlay was replaced by a central server. We also study the cost of using direct connections for CoAP observation relationships between sensors and their observers (4) to tunneling of CoAP messages across the overlay (5) or through a central server. Thus, we study four different scenarios. In scenario 1, called RELOAD-dedicated, all the PNs are part of a RELOAD overlay and dedicated ICE-negotiated connections are used for CoAP observation relationships. In scenario 2, called RELOAD-tunnel, no dedicated connections are set up for CoAP observation relationships. Instead, all notifications from sensors to observers are tunneled across the RELOAD overlay in the payload of RELOAD messages. In scenario 3, called C/S-dedicated, there is no RELOAD overlay. Instead, a star topology in which all PNs communicate with a central server is used. However, there is still a P2P aspect present as dedicated CoAP observation relationships are established in a peer-to-peer manner directly between the sensors and their observers. In scenario 4, C/S-tunnel, no dedicated connections are used for CoAP. Instead, all traffic, including CoAP notifications are sent tunneled via the central server.

C. Traffic Model and Simulation Parameters

Based on the use case described above, the size of the overlay network is 2000 peers. The size of each WSN is 10 LNs, resulting in a total of 20000 LNs. Each WSN has ten outgoing CoAP observation relationships. Observed resources send CoAP notifications on the average every 10 minutes. The duration of the simulated period is one hour.

The Chord stabilization interval was set to 120s. With this stabilization rate, the overlay can handle a churn rate where up to 250 peers depart or 500 peers join the overlay during an one hour period, which is more than enough for our use case, in which the overlay is expected to be fairly static. The size of the Chord finger table was set to eight peers following the recommendation in [7] to use on the order of $O(\log N)$ fingers. The size of the successor and predecessor lists was set to three peers based on the minimum recommended in [6]. The ICE keepalive interval was set to 15s, which is the default value in ICE [8]. In the simulations, all PNs are located behind P2P-friendly NATs with endpoint independent mapping and filtering behavior, meaning that relay servers are never needed. We used the same ICE stopping criteria as in [16]. We assume that when a PN receives a CoAP request destined to an LN, the PN always returns an immediate CoAP response in a CoAP ACK. Further, the PN always serves the CoAP request from a local cache instead of forwarding the request to the LN. This choice was made since in the simulations, we are only interested in comparing the delays associated with using a RELOAD overlay to those associated with using a central server. We are not interested in the delays within the WSNs. The simulation parameters are summarized in Table I.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of PNs</td>
<td>2000</td>
</tr>
<tr>
<td>Number of LNs per WSN</td>
<td>10</td>
</tr>
<tr>
<td>Total number of LNs</td>
<td>20000</td>
</tr>
<tr>
<td>Outgoing CoAP observations per WSN</td>
<td>10</td>
</tr>
<tr>
<td>CoAP notification interval</td>
<td>10min</td>
</tr>
<tr>
<td>Duration</td>
<td>3600s</td>
</tr>
<tr>
<td>Chord finger pointers</td>
<td>8</td>
</tr>
<tr>
<td>Chord successors</td>
<td>3</td>
</tr>
<tr>
<td>Chord predecessors</td>
<td>3</td>
</tr>
<tr>
<td>Chord maintenance interval</td>
<td>120s</td>
</tr>
<tr>
<td>STUN keepalive interval</td>
<td>15s</td>
</tr>
</tbody>
</table>

VI. RESULTS

In this section, we will present the results of the simulations.

A. Delays

The CoAP delays of the four scenarios studied are shown in Figures 7 and 8. Figure 7 shows the average delay of establishing a CoAP observation relationship, whereas Figure 8 shows the average delay of CoAP transactions. The error bars in the figures represent 95% confidence intervals.
From Figure 7, we can observe that the delays of establishing a CoAP observation relationship using a dedicated ICE-negotiated connection are 28.1s and 6.3s in the RELOAD-dedicated and C/S-dedicated scenarios, respectively. When all CoAP messages are tunneled, the CoAP observation relationship establishment delays are 25.5s and 3.0s for the RELOAD-tunnel and C/S-tunnel scenarios, respectively. As expected, the RELOAD delays are multiple times higher than the C/S delays. This is because of the additional delay associated with sending RELOAD lookup, attach, and tunnel messages over multiple hops across the overlay. Each hop involves sending the message twice over the 3G wireless radio interface (i.e., in the sender’s and receiver’s wireless access networks).

From the figure, we can observe that the reason why dedicated connections are more expensive for both RELOAD and C/S than the use of RELOAD Tunnel requests to establish the observation relationships is due to the ICE negotiations, which take roughly 3.2s for both RELOAD and C/S (the difference in delays between RELOAD and C/S is not statistically significant).

Figure 8 shows the delays of subsequent CoAP transactions sent after the observation relationship has been established. As expected, the delays are equal, roughly 1.1s for both the RELOAD-dedicated and C/S-dedicated scenarios. The delays of tunneling CoAP messages across the overlay or via the central server are 11.7s and 1.8s for the RELOAD-tunnel and C/S-tunnel scenarios, respectively. Thus, the tunneled scenarios have clearly higher cost than the scenarios using dedicated connections. Especially the cost of CoAP tunneling across the RELOAD overlay is so high that it may not be feasible in practice if observers require real-time or near-real time information from the sensors.

Although the delay associated with establishing the CoAP observation relationship is 4.5 times higher for the RELOAD-dedicated than for the C/S-dedicated scenario, it is worth noting that this delays occurs only once when establishing the CoAP relationship. In the RELOAD-dedicated scenario, after this one-time cost, subsequent CoAP transactions experience no extra delay compared to the C/S-dedicated case. Thus, if this one-time additional cost can be tolerated, as one would expect in a typical use case, the use of a RELOAD overlay is no more expensive than the use of a central server when it comes to CoAP delays.

B. Traffic Load

The total amount of application protocol (CoAP, RELOAD, STUN) traffic exchanged in the overlay during the one-hour period is shown in Figure 9 for our four scenarios. In the RELOAD-dedicated scenario, the traffic consists of CoAP messages, STUN keepalives and connectivity check traffic for RELOAD and CoAP, and RELOAD overlay maintenance traffic. In the RELOAD-tunnel scenario, the traffic consists of RELOAD maintenance traffic, RELOAD application traffic (i.e., Tunnel requests), and STUN keepalives for RELOAD. In the C/S-dedicated scenario, the traffic consists only of STUN keepalives for CoAP and RELOAD, and CoAP messages. In the C/S-tunnel scenario, the traffic consists only of RELOAD application traffic and STUN keepalives for RELOAD.

From the figure, we can observe that in the RELOAD-dedicated scenario, the total traffic is 938MB, whereas in the C/S-dedicated scenario it is 365MB. The RELOAD-tunnel scenario generates 750MB of traffic, whereas C/S-tunnel generates only 124MB. The higher cost of the RELOAD scenarios is explained especially by the RELOAD overlay maintenance traffic and high amount of STUN traffic required to keep the connections between PNs alive. In all of the scenarios, a large part of the total traffic is STUN keepalives. The percentages are 71%, 43%, 96%, and 53% for the RELOAD-dedicated, RELOAD-tunnel, C/S-dedicated, and C/S-tunnel scenarios, respectively. For the RELOAD-dedicated and C/S-dedicated scenarios, the largest source of traffic are STUN keepalives for CoAP connections. The percentage of keepalives for CoAP out of total traffic are 40% and 89% for the RELOAD-dedicated and C/S-dedicated scenarios, respectively.

Thus, we can conclude that, as expected, with the traffic model described in Table I, the C/S scenarios generate considerably less total traffic than the RELOAD scenarios. However,
the main difference is of course that in the C/S scenario, the central server needs to handle either all (C/S-tunnel) or a part (C/S-observation) of the total traffic. Therefore, it is interesting to study also the incoming traffic of the central server. We will do this in the second set of simulations described below.

It is interesting to compare the traffic load of our four different scenarios in more challenging conditions. For this, we ran a second set of simulations. In these simulations, we increased the number of PNs to 1000 and varied the number of sensors per WSN between 1 and 100. Each WSN had one outgoing CoAP observation relationships per LN. CoAP notifications were sent on the average at 10 minute intervals. Further, we assumed that a subset of the PNs are behind P2P-unfriendly NATs with address and port dependent mapping and filtering behavior. When two such PNs need to communicate with each other, a TURN server needs to relay all the traffic between the PNs. The types of NATs in mobile operator networks is studied in [17]. Based on these results, we assumed that 11.1% of connections between peers require the use of a relay. As a comparison, for Google Talk and Skype, the corresponding figure has been found to be 8% [29] and 9.6% [30], respectively. The traffic model for the second set of simulations is summarized in Table II. The table shows only the parameters that were changed from those in Table I.

The total amount of traffic exchanged in each of the four scenarios is shown in Figure 10. The y-axis in the figure uses logarithmic scale. From the figure, we can see that the C/S-tunnel scenario still has the lowest amount of traffic. Unlike in Figure 9, the RELOAD-tunnel scenario has now the highest total amount of traffic (except for the case when there is only one LN per WSN). The amount of traffic in the C/S-dedicated scenario starts to approach and eventually becomes nearly equal to the traffic in the RELOAD-dedicated scenario as the number of LNs per WSN increases. This is because CoAP messages and STUN keepalives for CoAP start to dominate and thus the additional cost of using RELOAD over C/S becomes negligible when looking at the total traffic levels.

Figure 11 shows the number of incoming Mbit/s for the central server in the C/S scenarios and for an average peer in the RELOAD scenarios. The y-axis in the figure uses logarithmic scale. From the figure, we can see that in the RELOAD scenarios, the PNs are not especially loaded. However, in the C/S scenarios, the load of the central server starts to grow rapidly. For instance, when there are 10 LNs per WSN, the server needs to already be able to handle an incoming traffic load of 3.6 and 11.1 Mbit/s in the C/S-dedicated and C/S-tunnel scenarios, respectively. Thus, this figure clearly demonstrates the scalability of the RELOAD scenarios.

In our final set of simulations, we decreased the mean interval of CoAP notifications to 60s. The resulting total traffic...
traffic in our four scenarios is shown in Figure 12. From the figure, we can observe that when the frequency of CoAP notifications is high, both the RELOAD-tunnel and C/S-tunnel scenarios become clearly more expensive than the RELOAD-dedicated and C/S-dedicated scenarios in terms of total traffic. Further, when looking at the amount of incoming Mbit/s per average peer (RELOAD scenarios) or the central server (C/S scenarios) shown in Figure 13, we can see that the RELOAD-dedicated scenario outperforms the other scenarios. In the worst case (100 LNs per WSN), the average Mbit/s received by an average peer or the central server is 0.04 Mbit/s, 0.40 Mbit/s, 42 Mbit/s, and 1078 Mbit/s for the RELOAD-dedicated, RELOAD-tunnel, C/S-dedicated, and C/S-tunnel scenarios, respectively. Thus, we can again see that the RELOAD scenarios (especially RELOAD-dedicated) scales very well, whereas in the C/S scenarios, the traffic load can become very high for the central servers.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a new architecture for wide-area M2M networks. The architecture uses the CoAP and RELOAD protocols to provide a P2P federation of geographically distributed WSNs. It is completely decentralized, that is, not dependent on central application servers and infrastructure-based services like DNS-SD. Other advantages include scalability, self-organization, robustness, cost-efficiency (both low capital and operational expenditures), and web integration.

We implemented the architecture and evaluated it through simulations, comparing its performance to that of a traditional system using centralized application servers. TBD.

We are currently working towards testing the implementation on real hardware. TBD.

REFERENCES


