A Multi-protocol Software-Defined Networking Solution for the Internet of Things

Tryfon Theodorou, *Member, IEEE,* George Violettas, *Member, IEEE,* Polychronis Valsamas, *Member, IEEE,* Sophia Petridou, *Member, IEEE,* and Lefteris Mamatas, *Member, IEEE*

***Abstract*—Internet of Things (IoT) has evolved from an experimental to a backbone technology able to connect myriads of people, things, and services for a large range of businesses. At the same time, the emergence of Software-Defined Networking (SDN) can ideally handle IoT challenges for elasticity, heterogeneity, and mobility, offering an architecture that abstracts decision-making away from the data plane, and provides a programmable network facility. Along these lines, we propose MINOS, a multi-protocol SDN platform for IoT that implements service-awareness utilizing: (i) appropriate *SDN abstractions and interfaces* for logically-centralized network control of diverse and resource-constraint IoT environments; (ii) *two network protocols* that are deployable and configurable on-demand; and (iii) *a Graphical User Interface (GUI)* that provides a bespoke dashboard and a real-time visualization tool. Due to its components, MINOS enables: experimentation with novel network control features and protocols that realize optimized routing over heterogeneous IoT nodes; application of real-time strategies as a response to the dynamic network conditions; support of individual protocol configurations per node; and flexibility to accommodate new protocols and control algorithms. Our results demonstrate MINOS as an enabling platform for two protocols, the *CORAL-SDN* and the *Adaptable-RPL*, which, in comparison with the state-of-the-art IoT routing protocol RPL, improve the *packet delivery ratio* with relative small *control overhead*.**

[1]

I. Introduction

Nowadays, IoT technology holds a cardinal role as an enabler for a highly diverse set of services in respect to their *requirements*, including extremely high data-rates, ultra low latency, low power consumption, large number of connected devices, and high mobility. Paramedics, a typical e-health example, demand high data-rates in real-time to support live video streaming to hospitals. On the other hand, wide sensors' deployments gathering ground and atmospheric measurements in large areas, prioritize scalability issues over data-rates. Traffic prioritization is a crucial requirement in harsh working environments that use IoT devices' deployments for safety reasons (e.g. prevent or face accidents); in this case, connectivity is of paramount importance. Finally, applications with mobile IoT devices, such as drones or human wearables, strive for efficient solutions (e.g. neighbor discovery and routing), that handle mobility and take into account constraints, such as the remaining battery power.

The authors are with the Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece.

[1]Accepted for publication in IEEE Communication Magazine, November Issue, with ID COMMAG-19-00056.R2. Date of first submission Feb 02, 2019, Date of first review April 25 2019, Date of acceptance Aug 1, 2019. PREPRINT VERSION.

Relevant applications in the literature [1] are categorized in line with the type of communication as follows: *Data Collection* for many-to-one, *Alerts and Actions* for point-to-point and *Data Dissemination* for one-to-many communication [2]. Apparently, there is no single protocol or communication mechanism that can address multi-application requirements hosted by IoT networks. In response to the need for agile and configurable solutions, Software-Defined Networking (SDN) provides a new, elastic network paradigm that can transform the traditional network backbones into flexible service-delivery platforms. We define *three IoT research challenges* that SDN can ideally handle:

1) *elasticity*: to appropriately deploy and configure different network protocols toward satisfying applications' requirements; moreover, to adapt to the network context environment (i.e. responding to IoT network's feedback) by enforcing strategies for flexible and individual IoT devices' configuration, that improve performance and resource-allocation whilst reducing cost.
2) *heterogeneity*: to integrate hardware (e.g. communication interfaces) and software (e.g. messaging protocols like CoAP) particularities, as well as nodes' characteristics (e.g. battery-powered or not). Carefully designed abstractions are needed to hide heterogeneity and allow devices to export common features to the higher control and application planes.
3) *mobility*: to handle issues raised from IoT devices' mobility and consequent connectivity hand-overs (e.g. additional *control overhead* to maintain the topology), which become "costly" without suitable dynamic routing adjustments. Furthermore, mobility-aware mechanisms should not overload possible co-existing static nodes.

A. Contribution

Addressing the aforementioned challenges, we move forward in implementing service-awareness, a networking research key requirement. This article presents the MINOS, an SDN platform aiming at providing elasticity for heterogeneous and/or mobile IoT deployments, through the operation and dynamic configuration of different protocols. We hereby experimented with the CORAL-SDN, a pure SDN protocol, and the Adaptable RPL which is an SDN-like protocol. In detail, MINOS introduces the following unique features:

1) an *SDN-based architecture* decoupling the data from the control plane. This is an important design step toward elasticity for the following reasons: keeps network's

heterogeneity transparent to the control and application planes; employs programmable interfaces for getting cross-layer measurements and enforcing appropriate strategies for adaptable topology and flow-control; implements software controllers providing logically-centralized control though reducing management cost and complexity.

2) *two network protocols* that are benefited by the SDN-based architecture (demo videos for both protocols are available online [3]):
   a) the *CORAL-SDN* [4], a software-defined OpenFlow-like protocol introducing adaptive topology control and routing strategies for IoT. The CORAL-SDN dynamically enforces adaptive combinations of topology discovery and control algorithms, leveraging network's elasticity. The impact of on-the-fly decisions is reflected in Section IV results discussion, demonstrating achievements in terms of *packet delivery ratio (PDR)* and *control overhead*.
   b) the *Adaptable-RPL* [5], an evolutionary extension of the IPv6 Routing Protocol (RPL) for Low-Power and Lossy Networks (LLNs) [6] with improvements for mobile and heterogeneous IoT networks. The MINOS platform handles core RPL's parameters both dynamically (i.e. in real-time) and individually (i.e. mobile versus static nodes). The results in Section IV show that adaptations in the RPL configuration, which mitigate the mobility issues, can efficiently tune the protocol's performance trade-offs, such as *PDR* versus *control overhead*.
3) a *GUI* consisting of a bespoke dashboard and a real-time visualization tool. Such a facility maintains the global network view providing on-the-fly configuration options, visualizing dynamic topologies, and illustrating real-time measurements. In addition, it can be extended in a straightforward manner through the user interface to support new algorithms, network protocol parameters and measurements, allowing scalable evolution.

This work is partially inspired by related Software-Defined Wireless Sensor Networking (SDWSN) approaches, including: (i) the SDN-WISE [7], a stateful SDWSN solution; (ii) the Soft-WSN [8], a platform implementing basic SDN features, that is topology and device management over application, control, and infrastructure layers; (iii) the TinySDN [9] that implements a distributed control plane SDN architecture for WSN; and (iv) the Whisper [10] that introduces an SDN controller transmitting routing and scheduling messages compatible to RPL, that is to manipulate its operation. The challenges of high-complexity and high-overhead that SDN architecture brings to LLNs are of those works main drawbacks, although a recent proposal μSDN [2] moderates this effect by introducing a lightweight SDN framework.

Our *CORAL-SDN* protocol further improves the network overhead through advanced topology control and routing algorithms, utilizing a separate control channel. Trading the performance with cost, these solutions are suitable for network installations with similar requirements, such as the use-case discussed in Section II. In contrast to the fully centralized control approaches, the distributed protocols' solutions, like [11] and [12], suggest extensions and mechanisms improving the operation of RPL under mobility and high data traffic, respectively. Our *Adaptable-RPL* protocol, harmonized to the latter approach, improves RPL through centralized SDN-like adjustments in the protocol configuration. These solutions retain low complexity making them suitable for public heterogeneous networks, while preserving full compatibility with the RPL protocol; however, they may achieve lower improvements compared to a pure SDN solution, indicated by the results in Section IV.

Since the related works are mainly adjusted to specific network scenarios or contexts, we introduce a facility that: i) accommodates and adapts multiple IoT protocols, because there in no "single protocol fitting all services" solution; and ii) addresses different *application and network requirements* through dynamic protocol adaptations (e.g. expressing particular network conditions and device constraints). These two characteristics constitute the main novelties of the MINOS platform. To the best of our knowledge, MINOS is the first Software-Defined Multi-protocol platform for IoTs [3].

In the following, a motivating use-case scenario highlighting the advantages of the proposed platform and protocols is presented in Section II. We elaborate on the MINOS platform's architecture in Section III, while comparative results are presented in Section IV. Conclusion remarks and future work insights are discussed in Section V.

II. Motivating Use-case Scenario

To motivate our proposal, we discuss a smart city use-case, representing an ecosystem with large numbers of interconnected IoT devices ("motes"), characterized by a wide range of communication challenges, such as device types heterogeneity (e.g. co-existing mobile, wireless and sensor networks), mobility behaviour (e.g. humans, drones, or vehicles, with different mobility patterns), or application message-exchange patterns (e.g. many-to-one versus one-to-one communication).

In such a scenario, motes equipped with sensors (i.e. pollution and weather related, along with traffic conditions) are monitoring the smart city's facilities and infrastructures. Cases include monitoring an urban area for air pollution, or a (smart) traffic control system, where a number of wirelessly connected motes are collecting physical world data, delivering them to a central point for further processing. Each mote can either operate as an end-point or a forwarding device. A centralized SDN platform can take decisions according to the "global view", and enforce the necessary communication strategies (for example, to prioritize crucial information to reach their destination on-time). Such an SDN-inspired model can increase the response time and network operation reliability. Simultaneously, it contributes to the up-scaling of a smart city infrastructure where new innovative ideas can be enabled from challenging communication approaches, including: a) a drone flying around to collect data from fixed or mobile IoT sensors about the city's conditions or incidents (e.g. weather, noise, pollution, traffic, car accidents) can safely transmit crucial

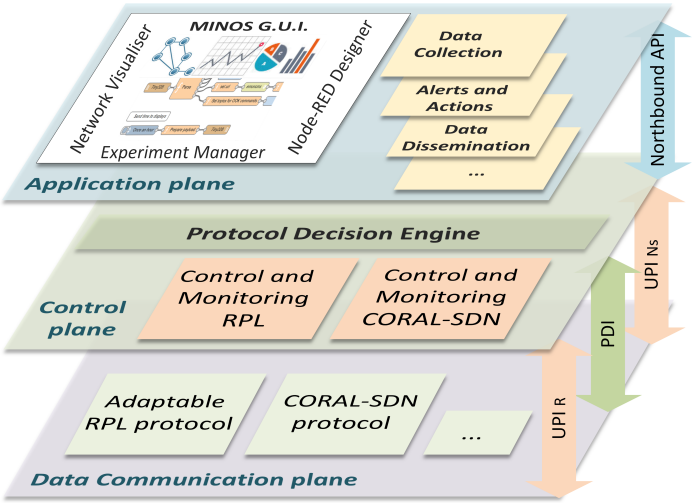


Fig. 1: The MINOS architecture

information; and b) human wearables or sensors embedded to smartphones or vehicles with diverse mobility patterns, that have to remain seamlessly connected to multiple networks along their way.

Arguably, an SDN-inspired platform, through the necessary abstractions, can support individual protocol deployments and configurations per groups of IoT nodes and realize adaptive communication strategies, mitigating the above challenging communication issues. The smart city use-case highlights that there is no single-protocol solution that can address the variety of requirements originated by dissimilar IoT applications. It also highlights the need of a service-aware platform accommodating multiple protocols, their on-demand deployment, along with network conditions' dynamic adaptations. The relevant MINOS solution is detailed below.

III. The MINOS Platform

The MINOS platform implements service-awareness by bringing together SDN with IoT technologies. It adapts IoT networks to the application requirements by on-demand deploying the appropriate network protocol, while considering the network environment constraints (e.g. limited node resource availability, mobility caused connectivity issues) by dynamically configuring the protocol in use. Fig. 1 presents a high-level view of the MINOS architecture. Aligned to the typical three-tier SDN paradigm, it consists of the following planes, described bottom-up as follows:

1) The *Data Communication plane* accommodates multiple dynamically-configurable protocols supporting diverse IoT devices operating either in real test-beds or in the Cooja emulator. It also provides radio and network protocol measurements to the upper layer, as well as on-demand protocol deployment.
2) The *Control plane* controls the network protocols at real-time based on information coming from both the *Application plane* (i.e. application requirements) and the *Data Communication plane* (i.e. network constraints). It consists of: (i) the *Protocol Decision Engine* that selects the protocol and its main configuration based on the application requirements; and (ii) protocol-specific control and monitoring components that are responsible for the run-time configuration adaptations and tracking the performance of the corresponding protocols, respectively.
3) The *Application plane* specifies the category of IoT application used (i.e. Data Collection, Alerts and Actions, or Data Dissemination) and the particular IoT node requirements, for example regarding their energy constraints and mobility support.

Right afterwards, we elaborate on the MINOS planes, interfaces, and functionalities.

A. The MINOS planes and interfaces

The bottom layer of the MINOS architecture (i.e. the *Data Communication plane*) supports multiple IoT protocols, real-time configuration and measuring of the protocols, as well as their deployment on-demand. MINOS currently supports two protocols:

1) the *CORAL-SDN* [4] is implemented in the context of the MINOS platform to provide adaptability to a range of IoT applications and network constraints (e.g. mobility and signal issues) through four alternative protocol mechanisms: (i) two network topology discovery and control algorithms, the first based on node advertisements, and the second on neighbor requests; and (ii) two types of flow establishment rules, configuring the full path or the next hop only, respectively. For example, in a fixed topology with one mobile node, it is resource-expensive to configure the whole path each time a single node changes its attachment point; instead, the next hop flow establishment option is more suitable. *CORAL-SDN* also supports other configuration options; for example, the link quality estimation method, the usage of acknowledgements and the control messages' interval time for the topology control.
2) the *Adaptable-RPL* [5] augments RPL, considered as the de facto routing standard for IoT, with dynamic reconfigurability to extend its applicability to alternative use-cases and dynamic network environments (e.g. it improves its responsiveness to sudden changes in the network conditions). At this point, MINOS adjusts a number of important RPL parameters (e.g. I_{min}, $I_{doubling}$) reflecting the responsiveness but also the communication overhead of the protocol, and the choice of the Objective Function to use for the distance-vector functionality of RPL. For example, in order to tackle the RPL's performance issues in mobile environments [5], MINOS adjusts the I_{min} parameter differently for nodes with particular characteristics, so communication overhead is offloaded from the mobile to the fixed nodes.

Furthermore, the *Data Communication plane* provides to the *Control plane* real-time measurements on the protocols' performance, or the configuration values of important parameters from both the radio (e.g. Received Signal Strength Indicator - RSSI or Link Quality Indicator - LQI), and network viewpoints (e.g. packet drops' number). The MINOS southbound

interfaces handle these interactions through utilizing novel APIs provided by the WiSHFUL project [13]. In particular, the *Universal Network and Radio Control Interfaces* (UPIs), that is one UPI_N per protocol, for network layer variables, and UPI_R for the radio channel. The WiSHFUL facilities provide hooks for dynamic adaptations in IoT communication protocols (e.g. RPL), and abstractions tackling the network or device heterogeneity. Following the WiSHFUL platform evolution, we further enriched its protocol adjustment capabilities, providing full support to our protocols via the MINOS platform.

The *Control plane* triggers the protocol deployment in an interchangeable manner through the *Protocol Deployment Interface (PDI)*. This plane currently supports two alternative ways for the on-demand protocol deployment: (i) *proactively*, for network operation scenarios with relaxed time constraints, where protocol changes are applied through updates in IoT devices firmware (i.e. a low memory-footprint approach with moderate deployment time); and (ii) *reactively*, for rapid protocol deployments, where a double-protocol stack above the link-layer dynamically selects one of the two alternative protocols (i.e. trading memory for quick protocol switching). Currently, we support proactive protocol deployment through device-specific Ansible scripts updating IoT devices' firmware. The reactive deployment is an on-going work, hence we released a first beta version of the double protocol stack [3]. Our plans also include experimentation with over-the-air protocol deployment approaches (e.g. utilizing elf contiki libraries).

The modular MINOS architecture allows easy existing protocols modifications (i.e. support of extra mechanisms or parameters) or further additions of new ones. As such, we are currently experimenting with an adaptable version of the Back-Pressure Routing (BPR) protocol (it is at an early stage). Our protocol implementations support diverse IoT hardware (e.g. RM090, and Zolertia Z1 devices).

The *Control plane* performs run-time network control and monitoring of the network environment through the *Protocol-specific Control and Monitoring* components, while it implements the service-awareness of MINOS, based on the application requirements originating from the *Application plane*, and being handled by the *Protocol Decision Engine (PDE)*.

The *Protocol-specific Control and Monitoring* components implement technology-specific local control loops for a subset or all nodes, monitoring the behavior of the network, while adjusting a rich set of network protocols' parameters to achieve the performance goals set by a particular application. At this point, MINOS supports two relevant components, reflecting the centralized network control features of the *CORAL-SDN* and *Adaptable-RPL* protocols:

- The *CORAL-SDN Control and Monitoring* component implements SDN controller functionalities that: (i) construct and maintain an abstract representation of the infrastructure network (i.e. a network connectivity structure with run-time node or link information), as for example the devices' battery level, or link quality measurements (e.g. RSSI or LQI); and (ii) perform centralized control of the data flows and define dynamic forwarding rules, responding to changes in the aforementioned network's abstract view, while matching the application requirements. For example, such control features perform dynamically topology local adjustments in the case of mobile nodes, to reduce the corresponding communication overhead.
- The *Adaptable-RPL Control and Monitoring* component collects measurements from the RPL protocol and triggers dynamic adaptations in the protocol parameters after changes in the network behavior, or to match application performance requirements. For example, an identification of mobile nodes coming from the *Application plane* results into different I_{min} parameter values for these particular nodes; that is to offload communication overhead to the fixed nodes with power source.
  Furthermore, if there is a need to prioritize a particular node-to-node communication (e.g. collecting data from all nodes to the sink may trigger an alert between two nodes), the MINOS platform may initiate minor topology changes through enforcing a new RPL Objective Function that prioritizes such communication (e.g. through link coloring), but with minor impact on the other co-existing nodes transmitting measurements to the sink node. This is another interesting extension of the current work.

The *Protocol Decision Engine (PDE)* selects the protocol to deploy, its enabled mechanisms (i.e. supported by the particular protocol) and initial configuration parameters, based on an *Application plane* request, that is to specify the communication type required by the application, the number and main node capabilities, as well as the global performance goals. Currently, *PDE* takes decisions based on hard-coded protocol strategies aligned to our experimentation analysis, but the results of an extensive experimentation analysis with more scenarios and a wide-range of network conditions will furthermore enrich its decision-making potential. Our short-term plans include a relevant machine-learning algorithm (i.e. neural network) inspired by [14].

Lastly, the *Application plane* specifies through the *Northbound API* the requirements of the application to be realized by MINOS. Such process is handled from the method: *configure_network(communication_type, nodes_configuration, prioritized_KPIs)*, where:

- The *communication_type* parameter actually defines the communication type, which can be: (i) many-to-one for the typical WSN *Data Collection* applications, where a set of nodes gather periodic measurements destined to a single sink node; (ii) one-to-many for *Data Dissemination* applications, where the sink node spreads data to all nodes in the network; and (iii) point-to-point for *Alerts and Actions* scenarios, where a node has to urgently interact with only one node. There is also the case of hybrid applications that support diverse communication methods for different parts of the network.
- The *nodes_configuration* parameter defines the number and characteristics of each IoT node, such as: (i) fixed or mobile; (ii) battery-powered or not; (iii) information on the particular resource constraints (e.g. maximum firmware size).

- The *prioritized_KPIs* parameter specifies the global performance goals for the application to be supported from IoT network. For example, an e-health application may request high bandwidth and low latency.

B. The MINOS GUI

The MINOS platform interacts with the user through a highly flexible GUI implemented in the Node-RED platform. The GUI allows the operator to override the *PDE* and select manually the protocol and its corresponding parameters; then the visualization outcome varies according to the protocol deployed. Representative screenshots are available online [3]. The dashboard performs the overall monitoring, while providing advanced functionality and configuration options through three sub-modules:

1) the *Experiment Manager*, providing configuration options related to available network protocols and experimentation set-ups;
2) the *Network Visualizer*, illustrating the network's topology, experiments' progress and results, and;
3) *the Node-RED Designer*, offering a library of the basic MINOS features implemented as Node-RED nodes and workflows. Hence, such features can be easily configured through the user interface, or parameterized short client-side Node.js scripts.

IV. Evaluation

We evaluated MINOS for handling mobility and heterogeneity in an IoT experimentation setup; for example, we assume a smart city scenario similar to the one described in Section II, where static nodes are located on buildings or stations, co-existing with mobile ones being placed in vehicles (all equipped with sensors offering city-monitoring facilities, e.g. temperature, humidity, noise, pollution). Through MINOS GUI we can proactively deploy our protocols (i.e. either the *CORAL-SDN*, or the *Adaptable-RPL*). Once one of them is deployed each time, we proceed with extracting results regarding two metrics: (i) the *PDR* (i.e. the proportion of packets delivered against total packets sent), and (ii) the *control overhead* (i.e. ratio of control packets to the total packets). In our comparative analysis, we use the *default RPL* as a baseline protocol, that is to contrast the MINOS platform against traditional IoT deployments.

A. Experimentation Setup

The protocols, accommodated in the *Data Communication* plane of the MINOS platform, are evaluated in a network with 21 IoT nodes (1 sink, 5 mobile, 15 static nodes), exploiting real-traces extracted from Stockholm city buses' routes, available via the MONROE project [15]. All our experiments are deterministic, hence we do not need to evaluate the results' statistical accuracy with multiple experiment runs. The combination of Cooja emulator scalability issues and the available processing power (we used an Intel(R) Core(TM) i5-2410M, 2.30 GHz processor), along with real-traces order of magnitude limitations, allows experimentation of this scale. Table I summarizes all experiments' settings.

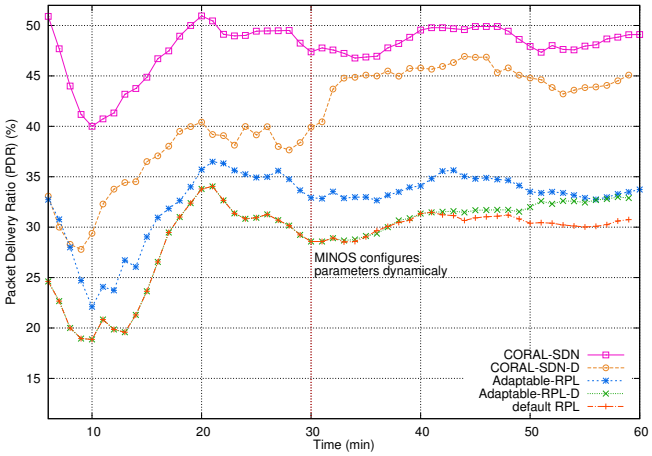
TABLE I: Experimentation Setup

Layer Setting	**Description**	**Notes**
Transport	UDP	Packet size 60 B
Network	IPv6/Rime	RPL/CORAL-SDN
Adaptation	6LoWPAN	
MAC	CSMA	
Physical	IEEE 802.15.4	Channel 26
	Radio Duty Cycle	128 Hz
IoT Motes	Zolertia Z1	Transceiver 2.4 GHz
OS	Contiki-OS	ver 3.0
Simulation	Cooja	
TX/RX	100%	Reliable Radio
Traffic load	Mobile: 120 data pckt/h	
	Fixed: 6 data pckt/h	
Mobility Model	real traces [15]	Canvas 750×750 m
Duration	1 h	

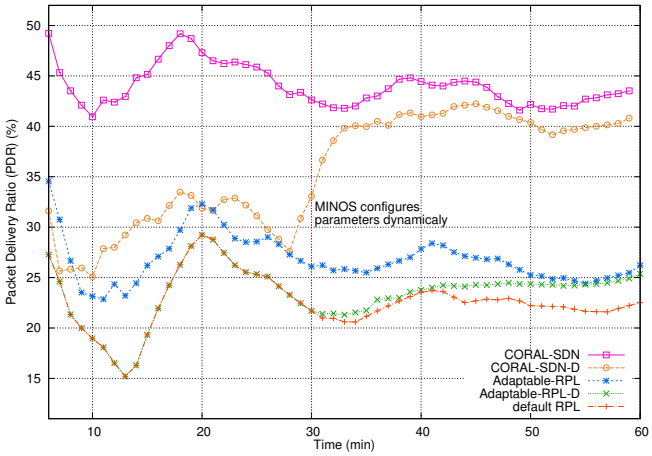
Methodologically, we conducted the same scenario five times as follows: (i) the first run employs the *default RPL* to provide a "ground truth" curve (red line in Figs. 2(a)-2(c) graphs); (ii) the next two runs consider the *Adaptable-RPL*, where the MINOS either Dynamically configures its parameters on-the-fly (via the UPIs) at the 30 *min* (results are indicated by the *Adaptable-RPL-D* green curve), or it adapts its configuration parameters from the beginning of the experiment (blue curve), and; (iii) the last two runs provide results for the *CORAL-SDN*, where the MINOS again either Dynamically adapts its parameters at the 30 *min* (*CORAL-SDN-D* purple curve), or it configures the protocol when the experiment starts (orange curve). The MINOS interventions' impacts are clearly shown in Figs. 2(a)-2(c), where after 30 *min* both the dynamically adapted scenarios (i.e. the *Adaptable-RPL-D* and *CORAL-SDN-D*), are progressively converging to the *Adaptable-RPL* and *CORAL-SDN* respectively, which in turn, are constantly superior to the *default RPL* which keeps the default parameters (i.e. I_{min} = 12 and $I_{doubling}$ = 8). For the *Adaptable-RPL-D* scenario, after the 30 *min*, the *Control and Monitoring* RPL component of MINOS triggers only the sink and fixed nodes to look in a more frequent basis for disconnected mobile nodes, that is tuning I_{min} = 8 and $I_{doubling}$ = 0. Similarly, for the *CORAL-SDN-D* scenario, also after 30 *min*, the corresponding component employs the *neighbor request* topology control algorithm that considers the neighbor requests [4], and configures topology maintenance parameter's rate at 4 *sec* for the mobile nodes and 10 *min* for the static ones. These last values could be adjusted by an intelligent algorithm or configured by the administrator to match the typical mobility patterns of buses. The results demonstrate the positive impact of dynamically tuning the aforementioned parameters through the MINOS platform on the *PDR*. Even in low density networks, there is a trade-off between *PDR* and *control overhead*, reflected differently in the two protocols.

B. Experimental Results

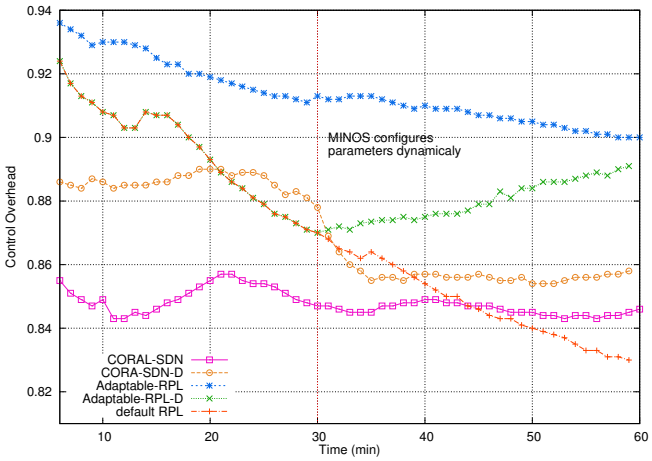
Figures 2(a)-2(c) show the performance of *default RPL*, *Adaptable-RPL*, and *CORAL-SDN* protocols, in terms of *PDR* and *control overhead*. We use these graphs with the time parameter on x-axis to demonstrate the ability of the MINOS to dynamically configure its protocols. An important



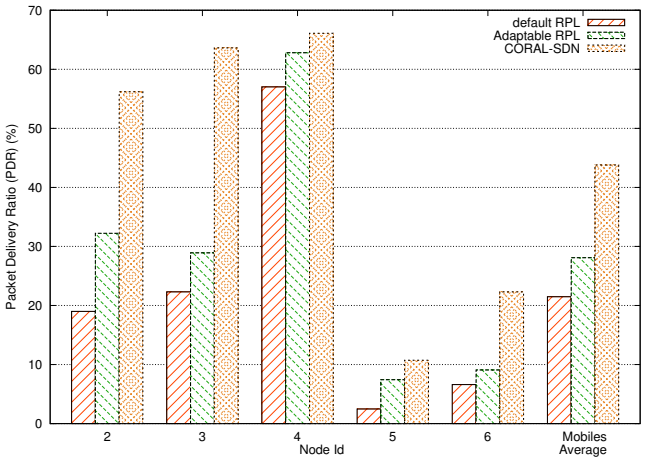
(a) *PDR* as a function of time for all nodes



(b) *PDR* as a function of time for the mobile nodes



(c) *Control overhead* as a function of time



(d) *PDR* for mobile nodes and mobiles' average

Fig. 2: MINOS experimental results for *PDR* and *control overhead*

observation derived by all sub-figures is that our platform and protocols outperform the default routing protocol regarding the *PDR*; especially in the case of the mobile nodes, achieves an improvement up to 7.74% for the *Adaptable-RPL* and 19.4% for the *CORAL-SDN* in the whole network (Fig. 2(a)), which rises up to 8.15% for the *Adaptable-RPL* and 21.5% for the *CORAL-SDN* for the mobile nodes (Fig. 2(b)). This outcome also highlights the benefits of offloading the *control overhead* to the static nodes. Since the mobility pattern for the nodes $2-6$ is a moving buses' emulation, there are long time periods of no connectivity for the mobile nodes because of radio limitations. This explains the fact that *PDR* does not exceed 50% in Fig. 2(b). Another interesting outcome is that our platform can achieve better *PDR* with relatively small *control overhead* (Fig. 2(c)), in case of employing the *CORAL-SDN* protocol. Fig. 2(d) presents the average PDR (over a period of 60 *min*) for each mobile node when the *Adaptable-RPL* and *CORAL-SDN* are used to tune their mobility-aware parameters from the beginning of the experiments. The bars clearly show that the mobile nodes can be potentially double advanced by the MINOS treatment compared to the standard RPL handling (e.g. nodes $2, 3$).

In general, the MINOS platform by selecting the *CORAL-SDN* protocol [4] achieves a high *PDR* with a marginally worse *control overhead*. In practice, the advanced topology construction algorithms of the protocol reduce the discovery time, while avoiding to flood the network with control messages. We indicatively found improvements in the *PDR* of the mobile nodes $2, 3$ and 6, up-to 37.1%, 41.3%, and 15.7% respectively. This happens because the *CORAL-SDN* succeeds to route messages through their neighboring mobile nodes. However, as previously indicated, this performance is brought with the additional cost of equipping the devices with a separate control channel. This investment is sensible in use-case scenarios requiring reliability in routing crucial messages (e.g. a smoke detection alarm, or critical infrastructure sectors).

Then again, in public heterogeneous networks such as the smart city environment where the extra hardware cost may not be reasonable, low complexity solutions like the *Adaptable-RPL* could be more suitable than the *CORAL-SDN* by steadily offering a higher *PDR* compared to the default RPL protocol (Figs. 2(a), 2(b)), since the mobile nodes remain connected to

the topology for longer periods. The improved *PDR* is traded for the increased *control overhead* (Fig. 2(c)), occasionally tolerated if, for example, the mobile nodes are powered by the hosting vehicle.

V. Conclusions and Future Work Insights

This paper presents the MINOS platform, a multi-protocol SDN facility, implementing service-awareness as a feature which amplifies the cardinal role of IoT technology. It stands between revolutionary approaches fully exploiting the SDN paradigm to provide centralized control, and evolutionary ones, which enhance IoT-oriented mechanisms with SDN-inspired functionalities to keep their pros, while moderating their inabilities in terms of elasticity, heterogeneity, and mobility. We adopt the SDN approach to build an architecture which can host multiple IoT protocols, while having the functional components to deploy them on-demand according to the application requirements, and configuring them in real-time responding to dynamic network conditions. MINOS follows a modular architecture and its planes serve as further experimentation place-holders.

In the evolution of this work we consider: (i) using the IoT-LAB of the FIT experimental platform, and the CityLab Fed4FIRE+ testbed, both allowing multi-hop deployments; (ii) evolving sophisticated decision mechanisms for (reactive) protocol deployment and network control; (iii) accommodating more protocols and optimal switching among them; and (iv) investigating performance trade-offs in respect to the network scale.

Acknowledgment

This work is partially supported by the H2020 NECOS project (grant agreement number 777067) and the open call schemes of the WiSHFUL (grant agreement number 645274) and MONROE (grant agreement number 644399) projects.

References

[1] I. Yaqoob *et al.*, "Internet of Things Architecture: Recent Advances, Taxonomy, Requirements, and Open Challenges," *IEEE Wireless Commun.*, vol. 24, no. 3, pp. 10–16, Jun. 2017.

[2] M. Baddeley *et al.*, "Evolving SDN for low-power IoT networks," in *4th IEEE Conf. on Network Softwarization and Workshops*, Jun. 2018, pp. 71–79.

[3] "MINOS open-source software and demo videos." [Online]. Available: https://github.com/SWNRG/minos [Accessed: Jun. 18, 2019].

[4] T. Theodorou and L. Mamatas, "Software defined topology control strategies for the Internet of Things," in *IEEE Conf. on Network Function Virtualization and Software Defined Networks*, Nov. 2017, pp. 236 – 241.

[5] G. Violettas, S. Petridou, and L. Mamatas, "Routing under heterogeneity and mobility for the Internet of Things: a centralized control approach," in *IEEE Global Commun. Conf.* IEEE, 2018, pp. 1–7.

[6] T. Winter, "RPL: IPv6 routing protocol for low-power and lossy networks," RFC 6550, Mar. 2012.

[7] L. Galluccio *et al.*, "SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for WIreless SEnsor networks," in *IEEE Conf. on Comput. Commun. (INFOCOM)*, Apr. 2015, pp. 513–521.

[8] S. Bera *et al.*, "Soft-WSN: Software-defined WSN manage. syst. for IoT appl." *IEEE Syst. J.*, 2016.

[9] B. T. de Oliveira and C. B. Margi, "TinySDN: Enabling tinyOS to software-defined wireless sensor networks," *XXXIV Simpósio Brasileiro de Redes de Computadores*, pp. 1229–1237, 2016.

[10] E. Municio *et al.*, "Whisper: Programmable and Flexible Control on Ind. IoT Networks," *Sensors*, vol. 18, no. 11, p. 4048, 2018.

[11] D. Carels *et al.*, "RPL mobility support for point-to-point traffic flows towards mobile nodes," *Int. J. of Distributed Sensor Networks*, vol. 11, no. 6, p. 470349, Jun. 2015.

[12] Y. Tahir, S. Yang, and J. McCann, "BRPL: Backpressure RPL for high-throughput and mobile IoTs," *IEEE Trans. Mobile Comput.*, vol. 17, no. 1, pp. 29–43, Jan. 2018.

[13] P. Ruckebusch *et al.*, "WiSHFUL: Enabling Coordination Solutions for Managing Heterogeneous Wireless Networks," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 118–125, 2017.

[14] S. A. Seidel *et al.*, "Analysis of large-scale experimental data from wireless networks," in *IEEE Conf. on Comput. Commun. Workshops*, Apr. 2018, pp. 535–540.

[15] O. Alay *et al.*, "MONROE: Measuring mobile broadband networks in Europe," in *IRTF & ISOC Workshop on Res. and Appl. of Internet Measurements*, 2015.

Tryfon Theodorou is a Ph.D. candidate at the Dep. of Applied Informatics, University of Macedonia, Greece. His area of expertise includes wireless sensor networks, software-defined networks, information security and IoT.

George Violettas pursues his Ph.D. in the area of Software-Defined Mobile Networks at the University of Macedonia, Greece. He holds a MSc Degree in Applied Informatics from the same University.

Polychronis Valsamas pursues his Ph.D. in the area of Cloud Computing and the Network Edge at the University of Macedonia, Greece. He holds a M.Sc. Degree in System Engineering and Management, Democritus University of Thrace.

Sophia Petridou is an Assistant Professor at the Dep. of Applied Informatics, University of Macedonia, Greece. Her main research interests are in the areas of optical and wireless networks.

Lefteris Mamatas is an Assistant Professor at the Dep. of Applied Informatics, University of Macedonia, Greece. His research interests lie in the areas of Software-Defined Networks, Network Function Virtualization, Mobile Edge and Fog Computing.