End-to-End Delay Enhancement in 6LoWPAN Testbed Using Programmable Network Concepts

Bilal R. Al-Kaseem, *Member, IEEE*, Yousif Al-Dunainawi and Hamed S. Al-Raweshidy, *Senior Member, IEEE*

***Abstract*—This paper introduces a proof-of-concept 6LoWPAN testbed to study the integration of programmable network technologies in relaxed throughput and low-power IoT devices. Open source software and hardware platforms are used in the implemented testbed to increase the possibility of future network extension. The proposed architecture offers end-to-end connectivity via the 6LoWPAN gateway to integrate IPv6 hosts and the low data rate devices directly. Nowadays, Software-Defined Networking (SDN) and Network Function Virtualization (NFV) are the most promising technologies for dealing with the massive increase in M2M devices and achieving agile traffic. The developed approach in this paper is entitled tailored Software Defined-Network Function Virtualization (SD-NFV), which is aimed at reducing the end-to-end delay and improving the energy depletion in sensor nodes. Experimental scenarios of the implemented testbed are conducted using a simple sensing application and the obtained results indicate that the introduced approach is appropriate for constrained IoT devices. By utilizing SD-NFV scheme in 6LoWPAN network, the data delivery ratio increased by 5-14%, the node operational time prolonged by 70%, the end-to-end latency for gathering sensor data minimized by ≈160%, and the latency for transmitting control messages to a specified node diminished by ≈63% when compared to a traditional (non SDN-enabled) 6LoWPAN network.**

***Index Terms*—M2M; 6LoWPAN Testbed; IoT; Programmable Network; NFV; Tailored SDN Controller; Cloud Computing.**

I. Introduction

The number of connected devices to the Internet is foreseeable to surpass the world's projected population by 2021, with there being 1.5 devices per capita [1]. The main purpose of these devices is to provide useful information about the surrounding environment and make it smarter. Machine-to-Machine (M2M) communication enables smart end-user devices to communicate with each other without human intervention [2]. Such communication will bring together people, objects, and processes to make the connected network more valuable and relevant through the Internet-of-Things (IoT) [3]. In the recent future, the IoT will bridge different technologies and produce new trends of applications via the connected users, computing systems and everyday objects.The connotations relating to the IoT concepts can be achieved through two seamless perspectives: communication standards and network types. The communication standards include but not limited to Bluetooth Low Energy (BLE), Radio Frequency Identification (RFID), ZigBee and IPv6 over Low power Wireless Personal Area Network (6LoWPAN) and Wireless-Fidelity (Wi-Fi) including the 5.850-5.925 GHz band that is used by the Wireless Access in Vehicular Environment (WAVE) communication system. While the network types are Wireless Sensor Network (WSN), Low Power Wide Area Network (LPWAN) and cellular network [4] [5]. The M2M network is completely different from the conventional one, in that it is characterized by a massive number of wireless devices that have the ability to communicate autonomously using low-power and low-data rate communications. Most of the M2M nodes are battery-powered and generate tremendous amounts of data, which have to be processed, stored, and presented in an interpretable form [6]. These sensor nodes are deployed in many applications, such as healthcare monitoring, natural disaster relief, surveillance, smart cities and transportation. However, their full effect will not been realized unless their data are being shared over the Internet [7].

Two models have been proposed for providing interaction between M2M sensor networks and the Internet. The first model is a proxy based system in which the Internet user's queries are forwarded to the sensor nodes via the system base station (gateway) [8]. This model suffers from high energy consumption and high end-to-end delay between the base station and the sensor nodes. Hence, network scalability and application development are restrained. Whilst the second model is an Internet protocol (IP)-based system, which considered as being an extension to the Internet [9]. In this model, the gateway is responsible for bridging the traditional communication network with a sensor network. However, IP-based M2M networks face considerable challenges, such as global addressing scheme, large IP header overhead, and the development of appropriate sensing applications.

In order to tackle the aforementioned challenges, the Internet Engineering Task Force (IETF) working group introduced the 6LoWPAN, which is based on the IEEE 802.15.4 standard. 6LoWPAN outlined the specifications required to carry out Internet protocols over lossy and relaxed throughput wireless devices. It also ensures the interoperability of the sensor networks and the Internet by taking into account the constrained characteristics of the M2M sensor nodes regarding the limited processor, small memory footprint, and scarce energy source. In addition, 6LoWPAN has an adaptation layer in its protocol stack, which achieves the suitability and sustainability of IPv6 packet when it is transmitted or received over the MAC and PHY layers [10].

The programmable networks have received considerable research interest, with many projects, such as Active Networking [11], Open Signaling [12], Ethane [13], and ForCES [14] having been implemented, but the outcomes have failed to be widely adopted by the research community. Two key reasons for this, are that they only focused on data plane

programmability, while neglecting the network control plane, and that they were vendor-dependent. Hence, the findings from these projects cannot be integrated directly into the IoT infrastructure owing to its heterogeneity.

Nowadays, Software-Defined Networking (SDN) [15] and Network Function Virtualization (NFV) [16] are the most promising advances for providing agile and seamless integration of different technologies in the IoT environment. On the one hand, SDN disengages the data forwarding functionalities (data plane) and the network control and management plane (control plane) to provide centralized and programmable network control. On the other hand, NFV implements network hardware functions in software packages and hence, delivers more elastic and efficient resource management.

The cloud-computing platform is another component set to help the IoT succeed. It enables end-to-end service delivery for remote end-users to access the M2M sensor network application from anywhere. The IoT generates tremendous amounts of data, and cloud computing provides a pathway for these to be stored and processed. That is, cloud computing architectures leverage several networks' infrastructure, functions, and services of the IoT [17] [18].

The remainder of this paper is organized as follows: Section II highlights the strategies for constructing 6LoWPAN networks that are related to the proposed approach. Section III details the general architecture and topologies of 6LoWPAN that have been introduced by the IETF working group. Section IV presents the programmable network concepts, whilst in Section V, the proposed approach is described and discussed. In Section VI, the performance of the developed approach is analyzed. Finally, the paper is concluded in Section VII with the current research challenges and future research directions.

II. Related Works

Before introducing our proposed approach, first, we explore the related research and give a brief review about the interoperability of programmable network concepts (i.e. SDN and NFV) in cloud-based LoWPANs (i.e. ZigBee and 6LoWPAN). In this paper the network programmability approaches are reviewed to assess their impacts on network performance.

The approach introduced by Mahmud *et al.* [19] can be recognized as one of the first efforts that leveraged SDN and network programmability for WSNs. Their proposed method, called the Flow-Sensor, was aimed at tackling the inherent problems of WSN and led to considerable enhancement in cloud–based network performance through network virtualization. The reachability points of the Flow-Sensor were more than that of a typical sensor. Finally, the authors concluded that improved outcomes might be attained if Flow-Sensor method deployed in large scale WSN.

Luo *et al.* [20] took a step forward by introducing a more coherent framework through identifying two common WSN problems: network management and difficulty of policy changes. In traditional WSN, network management was hard to perform and hence, a new paradigm was needed, where node re-tasking would be easily managed through software. Manual policy reconfiguration was required, if they needed to be deployed in alternative ways. Their developed architecture called, Software Defined-WSN (SD-WSN), was very similar to traditional SDN architecture. A modified version of OpenFlow protocol was deployed, called Sensor OpenFlow (SOF). However, their proposal was limited as no experimental validation was conducted.

Costanzo *et al.* [21] introduced another SDN-based architecture, called the Software-Defined Wireless Network (SDWN). The authors analyzed the impact of SDN in Wireless Personal Area Networks (WPANs) and discussed the SDWN requirements for achieving network flexibility through flow table rules and enhancing node duty cycles in wireless networks that are based on the IEEE 802.15.4 standard.

Gante *et al.* [22] developed a smart management scheme for WSN based on SDN. In their approach, the SDN controller was executed inside the network base station. In addition, a new solution was offered by their architecture for dealing with some of the inherited problems, such as network management and energy efficiency. The forwarding rules were simply modified by the network application at the SDN controller, which then disseminated the changes to sensor nodes.

Kruger *et al.* [23] introduced an IoT gateway using a Raspberry PI B board, based on a 6LoWPAN protocol stack. The sensor nodes were connected to the IPv6 network using a Constrained Application Protocol (CoAP) and tested using a smart water meter application. Krylovskiy [24] utilized several synthetic and application benchmarks to determine the virtualized layer overhead for efficiently designing IoT gateways. His proposed approach was implemented using a Raspberry Pi 2 board, but it lacked an energy efficiency evaluation. Morabito *et al.* [25] proposed a multi-functional IoT gateway using Raspberry Pi 3 and Odroid C2 boards. They investigated the interaction and orchestration in distributed data processing applications using different hardware platforms. Petrolo *et al.* [26] utilized virtualized software to support lightweight and dense deployment of services at the gateway level using a Raspberry Pi 2 board. Their proposed approach involved analyzing the possibility of sensor node and gateway interaction in an IoT environment through dynamic allocation of virtualized services. Another sensor virtualization scheme for WSN was presented in [27], which was first introduced by Khan *et al.* [28]. Their proposed approach was based on multi-layer architecture aimed at enhancing the performance of the already deployed WSN hardware by enabling the sensor nodes to run concurrent tasks that belong to different applications.

A large number of published studies sought to investigate the effectiveness of cloud computing and programmable network concepts in WSN. To the best of our knowledge, none of this research involved investigating the incorporation of SDN, NFV, and cloud computing in IoT, except for that of Bizanis *et al.* [15] and Al-Kaseem *et al.* [29]. The works presented by Yang *et al.* [30] and Li and Chen [31] are relevant to both SDN and virtualization in wireless and mobile networks. Such surveys, however, have failed to target IoT applications. Sood *et al.* [32] reviewed the emergence of SDN and highlighted its recent significant in wireless and optical domains with the aim of consolidating SDN and IoT, but the authors did not give any attention to network virtualization.

Surveys, such as that conducted by Khan *et al.* [27],

were focused on virtualization in WSN without considering SDN integration in IoT architectures. On the other hand, a survey presented by Al-Fuqaha *et al.* [33] offered probably the most comprehensive information regarding IoT enabling technologies, protocols, and applications. However, it had certain limitations regarding SDN and/or NFV as future IoT enabling technologies.

This paper is motivated from the observation that an enormous number of M2M sensor nodes perform control decisions independently, which makes network management and control more difficult. The ever-growing M2M devices will generate massive traffic, which result in an energy scarcity in M2M sensor nodes and may increase the latency. One of the paramount challenges to be resolved in developing IoT-based applications, is the integration of different wireless communication standards and programmable network enabling technologies, as well as different types of sensor hardware. Recent developments in M2M sensor networks have strengthened the need for node re-tasking even after deployment. SDN and NFV are being used to add programmability and flexibility to M2M sensor networks and to enable a node's reconfiguration during its lifetime. Finally, 6LoWPAN is enabling IPv6 for constrained IoT devices, because it can deal with the limited processing capability, small memory size, and finite energy source of the M2M sensor nodes.

From the published studies in the literature explored previously, it would appear that there has been limited research that integrates SDN and NFV with available 6LoWPAN hardware. In contrast to these proposed solutions, our published work in [29] comes up with a new architecture to enhance the performance of 6LoWPAN network and analyze the network from energy efficiency perspectives. The proposed approach comprises of a tailored SDN controller and NFV technology, where these technologies being deployed in 6LoWPAN network with cloud connectivity. When designing 6LoWPAN testbed with node re-tasking, there are some systematic and situational challenges need to be considered, these include: a) The cost and type of network infrastructure that support different new technologies; b) Available software tools and hardware platforms to build and manage the network; c) The risk of shifting to a new technology to enhance the performance of the existing network; d) The ability of deploying new protocols, procedures and applications that support other technologies. It is therefore, this paper distinguishes from our previous work [29] by the following main contributions:

1) This paper proposes SDN controller based functionalities that are capable of accessing and updating the running processes on programmable sensor nodes by modifying sensor node properties on the fly and performing autonomous decisions;
2) Developing new packet forwarding procedure for the introduced packet format to minimize the latency in flow-table construction and reduce the delay during network discovery and data communication phases;
3) Minimizing path construction delay and performing address-independent source routing in 6LoWPAN networks by running concurrent virtualized procedures of 6LoWPAN protocol stack on the controller.

In this work, the entire 6LoWPAN platform relies on low cost and open source components (hardware and software). It is validated through a proof-of-concept testbed experiments consisted of 12 simple nodes, 12 advanced nodes and one gateway. The obtained experimental results show a remarkable enhancement compared to a traditional (non SDN-enabled) 6LoWPAN network in terms of end-to-end delay.

III. IPv6 over IEEE 802.15.4 Standard (6LoWPAN)

In 2003, the first version of the IEEE 802.15.4 standard [34] was released and this was revised in 2006 [35]. This standard defines the radio communication at 868 MHz, 915 MHz and 2.4 GHz for low-power, short range and low-data rate wireless embedded devices.

Depending on the operational frequency of the IEEE 802.15.4 standard, different data rates are provided ranging from 20-250 kbps, while the channel access mechanism operates using Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). In CSMA/CA, the node with a packet to transmit tries to listen to the communication channel to check whether there is another node transmitting a packet within its communication range. If this is the case, the node waits until the current transmission is complete. If the medium becomes free and no traffic is generated, the node will start transmitting a packet, whilst otherwise, it has to wait until the medium becomes clear, as it is being used by another node. In addition, CSMA/CA with acknowledgment, is provided to increase the reliability at the MAC layer and the 128-bit Advanced Encryption Standard (AES) is provided to enhance security at the link layer. Unicast and broadcast addressing capabilities are provided with two types: short (16-bit) and long (64-bit). Also, the MAC layer runs in one of two modes: beacon-enabled and beaconless modes. The beacon-enabled mode uses Time Division Multiple Access (TDMA) for allocating time slots to each node, while the beaconless mode uses the pure CSMA mechanism. Finally, the PHY layer payload is up to 127 bytes, and it uses different types of modulation depending on the operational radio frequency.

The Internet can be divided into three categories [36]: the core, the edge and the wireless embedded Internet. The core internet pertains to the build-up of servers, routers and the connected network devices, while the edge Internet relates to the user's units, such as PCs, cell phones, etc. The last category is not very large-scaled and is the wireless embedded Internet. It refers to the smart objects and small embedded devices with limited computation capability and that are power constrained. The wireless embedded Internet is built-up by many small stub networks. 6LoWPAN is one of these, where the edge router or gateway shares the same IPv6 address prefix with all the connected nodes. The edge router can be connected to the Internet and be responsible for routing the traffic from and to the 6LoWPAN nodes. 6LoWPAN networks can also exist without a edge router connected to the Internet, as so-called Ad-hoc LoWPAN, which is outside of the scope of this paper.

As stated earlier, 6LoWPAN was developed by the IETF working group in 2007 and IPv6 is the newest version of the Internet protocol, with 6LoWPAN providing direct integration

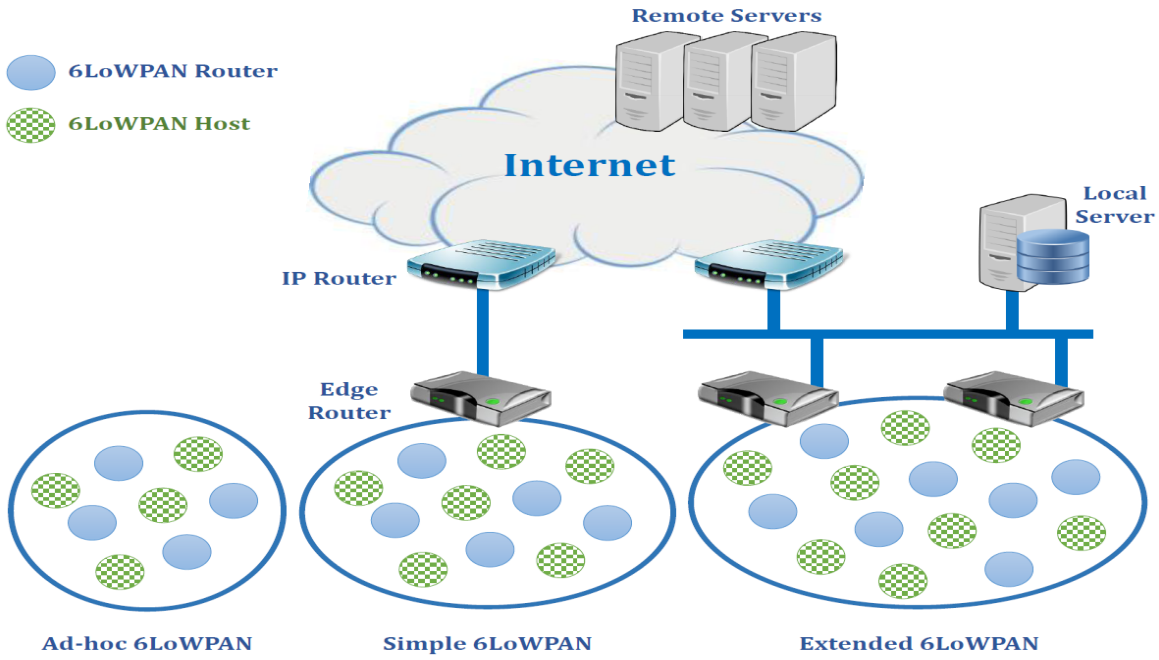

Fig. 1. The 6LoWPAN architecture

of IPv6 over the IEEE 802.15.4 standard. Accordingly, each node in the 6LoWPAN network becomes accessible through the Internet. 6LoWPAN has been designed with IEEE 802.15.4 in mind and consequently, some of its specifications are closely tied to features of the IEEE 802.15.4 link layer, such as using the Personal Area Network Identification (PAN ID) for address management. 6LoWPAN networks run in the beaconless mode of the CSMA/CA via IEEE 802.15.4 channel, which is called unslotted by the IEEE 802.15.4 standard. According to the 6LoWPAN specifications provided in [10], the acknowledgment frames are recommended in order to enhance network reliability and to recover the lost frames during transmissions at the data link layer.

Due to M2M node constraints, supporting IPv6 to these heavily constrained devices poses several challenges: IPv6 datagrams do not directly fit with LoWPAN, there is limited buffer size in M2M devices, and there are energy efficiency requirements. The minimum Maximum Transmission Unit (MTU) required for IPv6 is 1,280 bytes, whereas the IEEE 802.15.4 link layer frame is only 127 bytes long, which is thus just one-tenth of the IPv6 frame. Accordingly, data fragmentation and compression are needed. 6LoWPAN defines an intermediary adaptation layer between the IEEE 802.15.4 MAC layer and the IPv6 layer for compressing the IPv6 header, performing fragmentation and for the assembly of an IPv6 packet when it is transmitted or received over IEEE 802.15.4 as well as providing seamless integration with the existing Internet network [37].

A 6LoWPAN network consists of many embedded wireless devices that are recognized by power constrained, low-data rate, and limited memory. The 6LoWPAN architecture is depicted in Fig. 1 in which the end-to-end communication for interconnecting LoWPANs to the Internet is illustrated. Each connected LoWPAN is an IPv6 stub network on the Internet, because the IP packets can be received from or sent to it, but there cannot be packet transit to other Internet networks.

As shown in Fig. 1, the architecture defines three types of LoWPANs, where each consists of multiple 6LoWPAN nodes with role and function: Ad-Hoc LoWPAN, simple LoWPAN and extended LoWPAN. Ad-Hoc LoWPAN is recognized as being infrastructureless and functions autonomously without being connected to the Internet. Within the 6LoWPAN network, there are two types of connected devices: host and router. The host does not route any packets and it is also known as end-device. The router can route data destined to another node (hosts and routers) inside the 6LoWPAN network. The simple and the extended LoWPANs are also infrastructureless. However, the former is connected to another IP network through one 6LoWPAN edge router, whilst the latter has multiple edge routers for connecting LoWPAN to external IP networks.The multiple edge routers of extended LoWPAN are connected together through a backbone link. The edge router is the coordinator of the 6LoWPAN network. It handles the translations between the 6LoWPAN and IPv6 networks. Typical 6LoWPAN consists of many nodes with one or more edge router, with the nodes using the same active link to communicate with each other and to send data to the IP network via the edge router. In the scenario of one or multiple edge routers, LoWPAN may be connected to the external IP networks through one or more dedicated links, such as Ethernet, Wi-Fi or General Packet Radio Service (GPRS) communications [36] [38].

IV. Programmable Network Concepts

The data communication in traditional sensor networks occurs between one end of the system and the other. The network scalability and availability are limited to the routing

algorithm, congestion and moderate the Quality-of-Service (QoS). In recent years, increasing interest has been raised in deploying more functions inside current network elements in order to achieve better performance in terms of network services and cost. As a result, programmable networks have been developed to cope the limitations present in traditional networks, which are able to build adaptive networks that have the ability to be reprogrammed even after deployment. The major difference between the traditional and programmable networks is that for the latter network elements are directly programmed using a minimal set of Application Programming Interfaces (APIs) by the user. The programmable networks target reconfiguration, simplifying and accelerating network programmability in a secure and centralized manner. The most recent approaches to programmable networks are SDN and NFV. The SDN architecture provides centralized control and management by increasing the network programmability through a centralized software-based controller (the control plane), while the network devices become simple packet forwarding devices (the data plane). On the other hand, the NFV architecture provides programmability features by running different network applications simultaneously on a single physical network node. SDN and NFV with the cloud-based gateway enable the remote users to deploy an executable code dynamically to perform new operations at runtime [39] [40].

A. *Software-Defined Networking*

Traditional (non SDN-enabled) network devices, such as switches and routers, have control and data planes. Network intelligence is centered on the control plane, where the decision what to do with incoming traffic is made. On the other hand, the data plane performs action on the arrived packets based on the decision taken by the control plane. The standardized protocols provided by the IETF and IEEE are usually implemented by different vendors. As a result, the same protocol may be executed in several ways by various vendors. In addition, these vendors may add proprietary features to the standardized protocols, which results in complex configuration and their being prone to error due to vendors' diversity in manufacturing network devices. These actions mean that the control plane of legacy networks is distributed. That is, every node in the network is independent and does not have the complete information about the network connectivity [41].

The Open Networking Foundation (ONF) [42] is a non-profit user-driven organization, concentrated on developing a new open standard for SDN. The main goal of SDN is to make the network manageable and agile by deploying programmable components in the network architecture. Two main features distinguish SDN from traditional network architecture: separating of the network intelligence, i.e. the control plane, from the packet-forwarding engine, i.e. the data plane, as well as enabling a centralized programmable component in network intelligence at the controller. SDN enables instantaneous monitoring of network resources and applying user defined policies, which brings new innovations for optimizing the network configurations through the centralized SDN controller [43].

The connotation of programmable networks has been introduced as an unprecedented method to expedite network control and management. SDN paradigm improves network configuration through programmable data-paths by disassociating the data plane from the control plane. Since network control is no longer included in each network element, SDN introduces a new component: the centralized SDN controller. The forwarding components disassociation makes the network more flexible to adopt the deployment of newfangled services straightforwardly via the centralized network controller by the network administrator [44] [45]. Fig. 2 shows the SDN architecture, where the network management is logically centralized in software-based controllers (the control plane), and network devices become a hardware forwarding device (the data plane). The application plane and data plane can be reconfigured via open programmable interfaces namely northbound and southbound interfaces respectively [46].

OpenFlow (OF) can be view as the first multivendor standard introduced by the ONF in order to implement SDN in network devices (switches and routers). It provides the interface between the SDN controller and network devices, both physical and virtual (hypervisor-based). The OF protocol enables the SDN controller to command the network devices on how to handle ingress data packets. In addition, the OF protocol enables independent innovation to be added to the network without any hardware replacement and start testing new application under different configurations [47] [48].

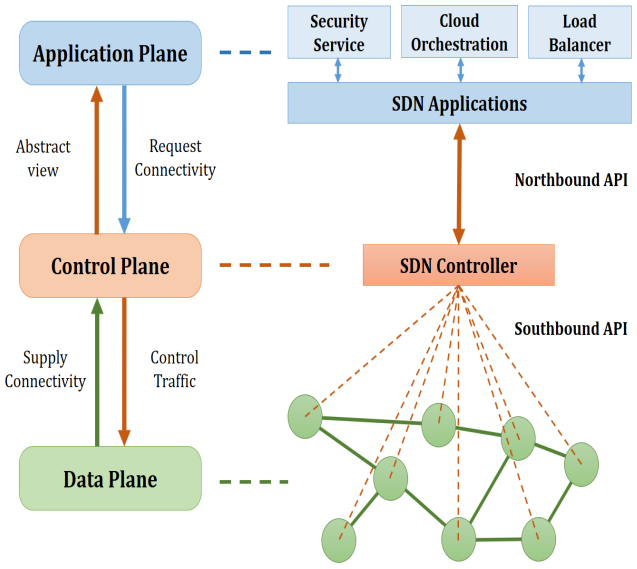

Fig. 2. High-level SDN architecture

When the control plane is separated from the data plane of networking devices, the whole network architecture is changed and hence, the SDN architecture encompasses three fundamental parts [43]:

1) The data plane pertains to the physical network forwarding devices (network elements), such as switches and routers, which provide network connectivity. Network elements become responsible for routing the data depending on the flow table entries that are provided by the SDN controller. The southbound Application Pro-

gramming Interfaces (APIs) are used by the controller to optimize and configure the network.

2) The control plane or the centralized controller is the mastermind and the organizer of SDN-based infrastructure. It is in charged of administrating the flow control (forwarding rules) in SDN hardware components, based on the desired performance requested from the SDN applications. The SDN controller configures the network elements via the general southbound APIs, and provides an abstract view about the network to the SDN applications via the northbound one. The SDN controller brings intelligence to the network architecture by enabling network programmability and management in centralized manner.
3) The application plane comprises programs that reciprocate network information with the centralized control plane via northbound APIs. These programs are able to construct a conceptualized network infrastructure depending on the gathered information by the centralized controller. The advantages over the current state networks, is that this enables new protocols to be deployed and tested, with the network being customized by only replacing the applications. In sum, network behavior can be altered via software with SDN, because it is fast, inexpensive, and easy to replace or update.

B. Network Function Virtualization

In the near future, the vast number of embedded devices connected to the Internet will need a new architecture in order to handle the massive quantities of generated traffic. NFV has the aim of virtualizing network services or applications with a view to running them on a distinct programmable unit. The NFV research has received considerable critical attention from both academia and industry as an essential trend towards the virtualization of network applications. It diminishes the operational and capital expenditures, whilst also empowering the integration of various services using software packages instead of physical network devices. These software-based services accelerate the deployment of user-defined applications on the same network physical platform and modify network behavior during its operation. Upgrading and replacing of virtualized software-based services is much easier than doing so for any physical device [41] [49].

The basic NFV architecture is illustrated in Fig. 3, where the Network Function (NF) has been abstracted from the physical network devices and implemented separately. In the NFV technique, the Virtual Network Function (VNF) is analogous to Physical Network Functions (PNF) in conventional networks. The relevance between VNF and PNF can be: one to one relationship in which single PNF is allocated to a single VNF, or one to many relationship in which diversified PNFs may be allocated to a unique VNF, or many to one mapping in which a single PNF is dispensed to multiple VNFs. By optimizing these mapping relationships, the network management and control is enhanced. NFV is one of the promising technologies appropriate for future IoT infrastructure, because of the following features [50] [51]:

- Network Performance: the same network performance obtained from traditional network functions running on dedicated hardware should be achieved, if the NFV technique is adopted. This task can be accomplished by minimizing network deadlocks;
- Heterogeneity Support: one of the trickier challenges needing to be addressed by NFV is the considerable variety of participating devices. Network heterogeneity can be generated from proprietary hardware based service perspectives and consequently, NFV is able to surmount the existing challenges of working across different communication standards;
- Dynamic Resource Allocation: NFV enables different network functions to be performed by the same physical hardware devices at different times. This can be achieved by reallocating the physical network resources among the available software packages.

NFV and its relationship with the complementary fields of SDN and cloud computing is discussed in [51] and [31], with comprehension of the state-of-the-art being provided in both works. A case study on NFV based gateways for Virtualized Wireless Sensor Networks (VWSN) was proposed in [52]. Bizanis et al. [15] reviewed some general SDN/NFV-enabled IoT architectures and identified promising research directions for future research.

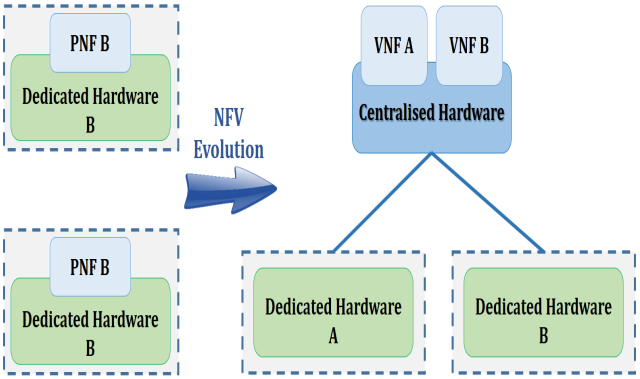

Fig. 3. Elementary NFV architecture

V. The Proposed Programmable Approach

The topics M2M communication, SDN, NFV and cloud computing are the in demand research domains of recent years. Much investigation has been focused on how to integrate them together to obtain consistent connectivity and to harmony the traffic load across the network. Extended SDN and NFV to M2M sensor networks were considered to be unpractical due to the resource constrained features of the embedded IoT devices in terms of small memory size, limited computation and low energy consumption. Most M2M devices in the IoT environment format the transmitted packets close to 127 bytes, while the current OpenFlow packets are close to 1,500 bytes. In addition, the addressing scheme of an M2M sensor network is unlike that used by IP-based ones. Consequently, the current OpenFlow protocol is not suitable for M2M sensor networks, because of the addressing scheme and the limited resources. In order to make the SDN concept valuable and applicable to

M2M sensor nodes in the IoT environment, there is a need for an unprecedented tailored SDN controller with an OpenFlow-like approach to overcome the limitations of these nodes, as outlined earlier.

In this paper, we adopted the programmable approach published in our previous work [29], which is called Software Defined-Network Functioning Virtualization (SD-NFV). On the other hand, this paper introduces a modified version of the tailored SDN controller where a new flow-table construction algorithm is presented. The main goal of this algorithm is to shorten path setup latency and accelerate link failure recovery. It is being deployed in a cloud-based 6LoWPAN testbed with **12 simple nodes**, **12 advanced node** and **1 border gateway**. The introduced approach in the paper at hand reduces end-to-end delay and prolongs the network operational time through software injection techniques that modified node's properties during its operation.

A. Testbed Hardware Components

The main components of an M2M sensor node are a microcontroller for performing some processing, a transceiver to communicate with other connected nodes in the network, an external memory to store sensory data, a power source to turn on the node and one or more sensors to gather sensory information from the surrounding environment. These wireless M2M devices have traditionally been connected using the IEEE 802.15.4 standard as the base with standards like ZigBee [53], 6LoWPAN [10] or WirelessHART [54] forming the upper layers. 6LoWPAN is the forerunner protocol endeavored at enabling constrained IoT devices that have limited computational capability and scarce energy to have IPv6 connectivity. In recent years, there has been a surge of interest in developing various open source and commercial 6LoWPAN testbeds. Most of these have been implemented to target operating system based nodes, where the 6LoWPAN protocol stack is run along with the node's operating system. However, the 6LoWPAN sensor nodes are typified as being severely resource-constrained devices and thus, it is unpractical to embed an operating system with dedicated software applications into those constrained nodes jointly.

With a view to promoting resource-constrained sensor nodes having IP connectivity, 6LoWPAN open source software and hardware resources need to be utilized efficiently for faster integration with existing IP networks. Accordingly, in this paper, an open source hardware based proposal for both sensor and sink nodes is introduced.

Recent research suggests that choosing an M2M node processing platform is among the most important factors that affects a node's performance. In order to meet the IoT network requirements, the 6LoWPAN nodes need to be energy-efficient and cost-effective. In this paper, the proposed approach is based on Arduino boards, which are open source microcontroller boards, based on the ATmega328 chip, as a computational platform. The Arduino based approach has a number of attractive features: small size, large memory, cost-effectiveness, low energy consumption, and it supports high level programming language. Based on the aforementioned features, Arduino boards pertain to a new perspective for enhancing network programmability and control via the integration of open source hardware platforms in IoT infrastructure. An XBee module is integrated with the Arduino board to transmit and receive IPv6 packets over the IEEE 802.15.4 standard. Finally, a temperature and humidity sensor is attached to the node's board as a sensing unit.

In order to implement the IoT paradigm, the things (connected objects) must be addressable and reachable via the Internet to enhance their controllability. For this research, a simple sensing application is used to investigate the effectiveness of the modified version of the SD-NFV approach. The 6LoWPAN nodes disseminate the sensor data to the 6LoWPAN edge router (gateway), and these sensor nodes are prototyped into three categories, as follows:

1) A simple node is capable of sensing the surrounding environment and communicating with other joined nodes in the network. However, this type of 6LoWPAN node has not been able to serve as cluster head in hierarchical topology. The simple 6LoWPAN node is based on the built-in microcontroller of the XBee module for data processing in which the temperature sensor (TMP36) and the data indicator (LED) are connected to it directly. A 5 V / 1200 mAh power bank is used to power the simple node and this is shown in Fig. 4.

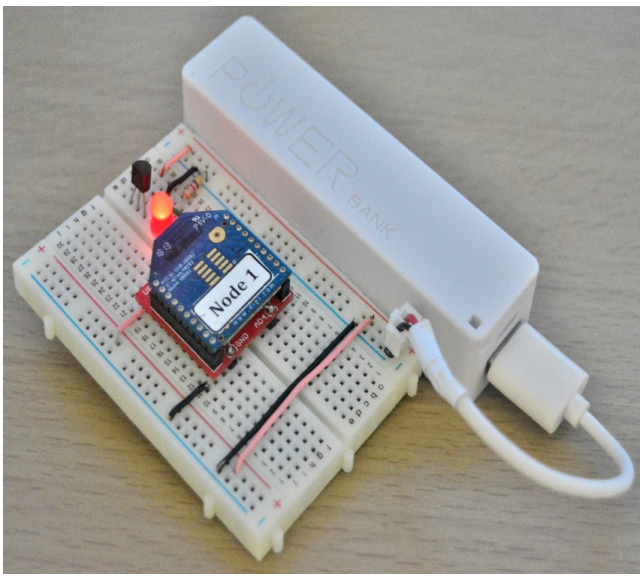

Fig. 4. 6LoWPAN-based simple node prototype

2) An advanced node has higher initial energy than a simple node. It performs the same functions of the simple node. In addition, It acts as a cluster head in hierarchical topology because it has extra processing power compare to simple nodes. The advanced 6LoWPAN node functions through an Arduino Uno board, which is connected to DHT11 sensor, XBee module, and an LED. A power bank with 5 V / 4000 mAh is used to turn on the advanced node components, as is shown in Fig. 5.

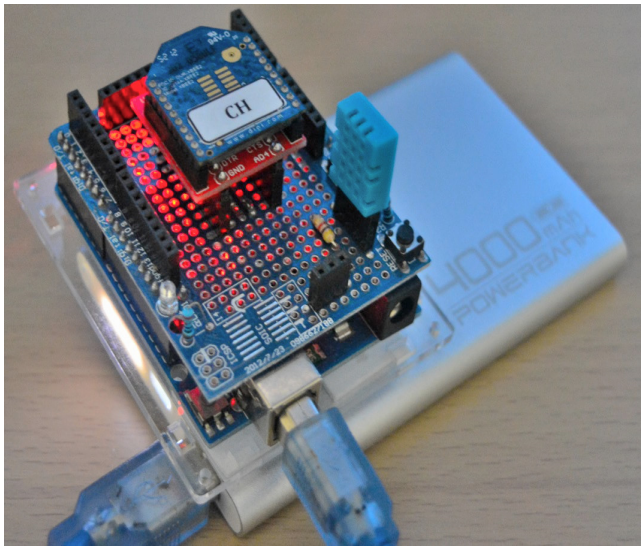

Fig. 5. 6LoWPAN-based advanced node prototype

3) A border gateway or sink node is the eventual destination of the entire 6LoWPAN nodes data, which lies between the sensory field and the cloud-computing platform. The 6LoWPAN gateway is based on a Arduino Mega board that is connected to two communication modules (XBee and ESP8266) and a Secure Digital Card (SD Card). The XBee module is used for low data rate 6LoWPAN communication, while the ESP8266 is deployed to connect the 6LoWPAN network to existing IP networks in conjunction with the cloud platform via the Internet. The SD card is used to provide large memory space to store the SDN flow tables. The sink node is permanently powered and is shown in Fig. 6.

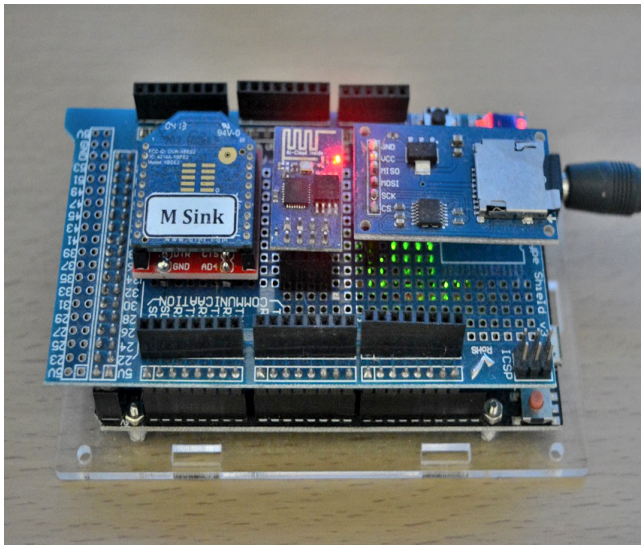

Fig. 6. 6LoWPAN-based sink node prototype

B. Tailored and Centralized Control Plane

The SDN and NFV are important trends that often exist together in IoT infrastructure. They aimed at evolving open software solutions for vendor-dependent networking devices. The NFV technology is directed towards bringing desired programmable components in order to utilize network resources efficiently by pinpointing their location in the network infrastructure. The SDN technology enables reconfiguration of network computing and programmability in network management by decoupling the control plane from the data plane. On the other hand, in spite of these explicit advantages about implementing SDN and NFV separately, bringing them together in one network infrastructure will attain greater achievements.

Unfortunately, the previously mentioned methods for connecting M2M sensor networks with the existing IP ones do not always guarantee a reliable connectivity and there are certain drawbacks associated with the use of them. Accordingly, another possible alternative approach is necessary in order to address these shortcomings. The proposed approach is to integrate SDN and NFV in 6LoWPAN networks, which enables the correlation of various routing algorithms via the deployed VNF in the 6LoWPAN gateway with integrated cloud connectivity. The typical architecture of a 6LoWPAN based sensor node is shown in Fig. 7 alongside 6LoWPAN and TCP/IP stacks. The SDN controller is a firmware network element that acts as strategic control point for deploying intelligent networks by enabling programmable components through various APIs. The SDN approach to M2M sensor networks is envisaged as resolving most of the inherent energy, routing, and latency challenges. In this paper, a tailored SDN controller is developed to bridge the research gap found in current literature.

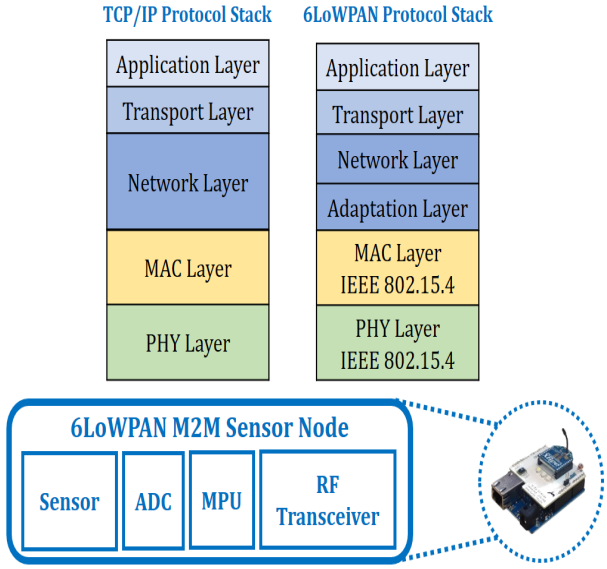

Fig. 7. Ideal architecture of M2M sensor node

The tailored controller is the brain of the SDN architecture that handles simultaneous flow control to provide intelligent networking. The tailored controller is in charge of the following: (i) construction of the global network topology; (ii) management of different network services; (iii) virtualization of various network functions; and (iv) load balancing and data routing. Furthermore, the SD-NFV approach enables the insertion of a new flow table entry to overcome the intensive memory profiteering of the programmable interface.

The architecture of the tailored SDN controller of the SD-NFV scheme is depicted in Fig. 8. The 6LoWPAN gateway comprises the network coordinator and the SDN controller. The 6LoWPAN coordinator is responsible for network initialization using a unique PAN ID and the topology discovery manager is in charge of constructing the global network topology using a discovery function. This function periodically checks the availability of the alive or newly joined 6LoWPAN nodes to update the alive node table entries in case of any network topology changes. The service manager has the vital function of dynamically assigning every node with a distinct function based on the node's preference in the tailored flow table entries. In addition, the service manager equips the 6LoWPAN network with cloud service connectivity, whilst the virtualization manager authorizes various sensor nodes to share the similar network function in the 6LoWPAN boarder router. Also, it enables each 6LoWPAN node to have virtual individual connectivity to the cloud computing platform. The last component of the SDN control plane is the load balancing and routing manager that engages in performing various data routing algorithms and running different load balancing techniques to optimize sensor network resources and to attain improved throughput with minimum end-to-end latency.

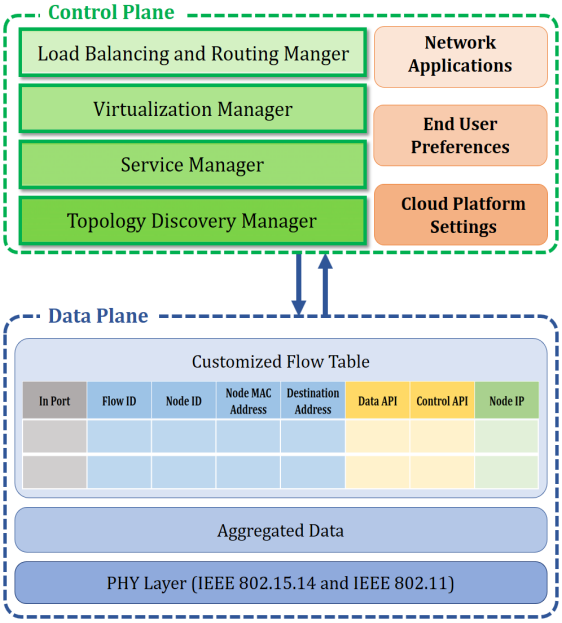

Fig. 8. The architecture of the tailored SDN controller

The OpenFlow protocol [46] has entirely been developed for wired networks and cannot be applied directly to wireless M2M sensor networks due to its complexity. The flow table entries are shown in Fig. 9, which are customized from the conventional SDN concepts and the OpenFlow standard. Its entries take into account the constrained nature of the M2M sensor nodes and each table entry is divided in three entities. The matching fields contain the conditions a packet needs to comply in order to be processed, the action field specifies the executed action, and the statistics field is used for processed packet counting. The most common actions are to discard the packet, to forward it or to modify the flow table entries.

Fig. 9 epitomizes the proposed approach of this paper which illustrates the association between 6LoWPAN based sensor networks, SDN, NFV and a cloud computing platform. It is clearly evident from Fig. 9 that each technology dissociates a specific function from several network resources and the significant advantages of using them are analogous in terms of saving node energy, agility of network traffic, cost-effectiveness, and scalability of the network.

The pico Internet Protocol version 6 (pIPv6) [55] is an open source Arduino-based software library used to implement the 6LoWPAN protocol stack in the current testbed. An C++ program that represents the tailored controller was executed within the 6LoWPAN gateway. The NFV is applied to remove the most energy harvesting layers from the physical 6LoWPAN node protocol stack to a logically centralized SDN controller at the gateway. The introduced virtualized mechanism is called Sensor Function Virtualization (SFV), which alters several node functions into separated applications executed within the border router. As a result, the border router utilizes the tailored SDN controller, the PAN coordinator, and the couple of virtualized layers (network and adaptation layers) from the sensor node protocol stack.

The first IPv6 specification of node discovery was released by Requests for Comments (RFC) 4861, which was then extended in 2012, by RFC 6775, in order to support the special requirements of 6LoWPAN node discovery. The border router, where the 6LoWPAN network coordinator is executed, is in charge of interconnecting the PAN network to the external IP networks and disseminating the IPv6 prefixes in association with the whole sensor nodes. In a traditional 6LoWPAN network, every node performs massive control message exchange with the border router to maintain its reachability. These control messages are Node Discovery (ND), Router Advertisement (RA), Neighbor Advertisement (NA), Neighbor Unreachability Detection (NUD), Duplicate Address Request (DAR), and Duplicate Address Confirmation (DAC). The 6LoWPAN node continuously transmits NUD messages until it receives an affirmation from the border router, even if it does not have data to send. The main issue in traditional 6LoWPAN topology discovery is that a considerable amount of energy is dissipated to keep reliable network connectivity through massive packet transmissions over the IEEE 802.15.4 medium. The challenge now is to develop a node discovery mechanism for a 6LoWPAN network that utilizes less packet exchange to maintain network connectivity, consumes less energy, thus prolonging network lifetime, and has minimum latency when discovering the global network topology.

The topology discovery manager, which is executed inside the centralized SDN controller, is in charge of keeping the 6LoWPAN network topology updated for dynamic data routing. The developed topology discovery mechanism benefits from the virtualized layers, as IP connectivity is not necessary at the 6LoWPAN node stage. Accordingly, the proposed scheme will minimize the number of transmitted packets to discover the node and increase the nodes' energy saving by eliminating the frequent NUD packets transmission and its correlated packet reciprocation. The developed discovery mechanism depends on the SDN flow table records. Upon

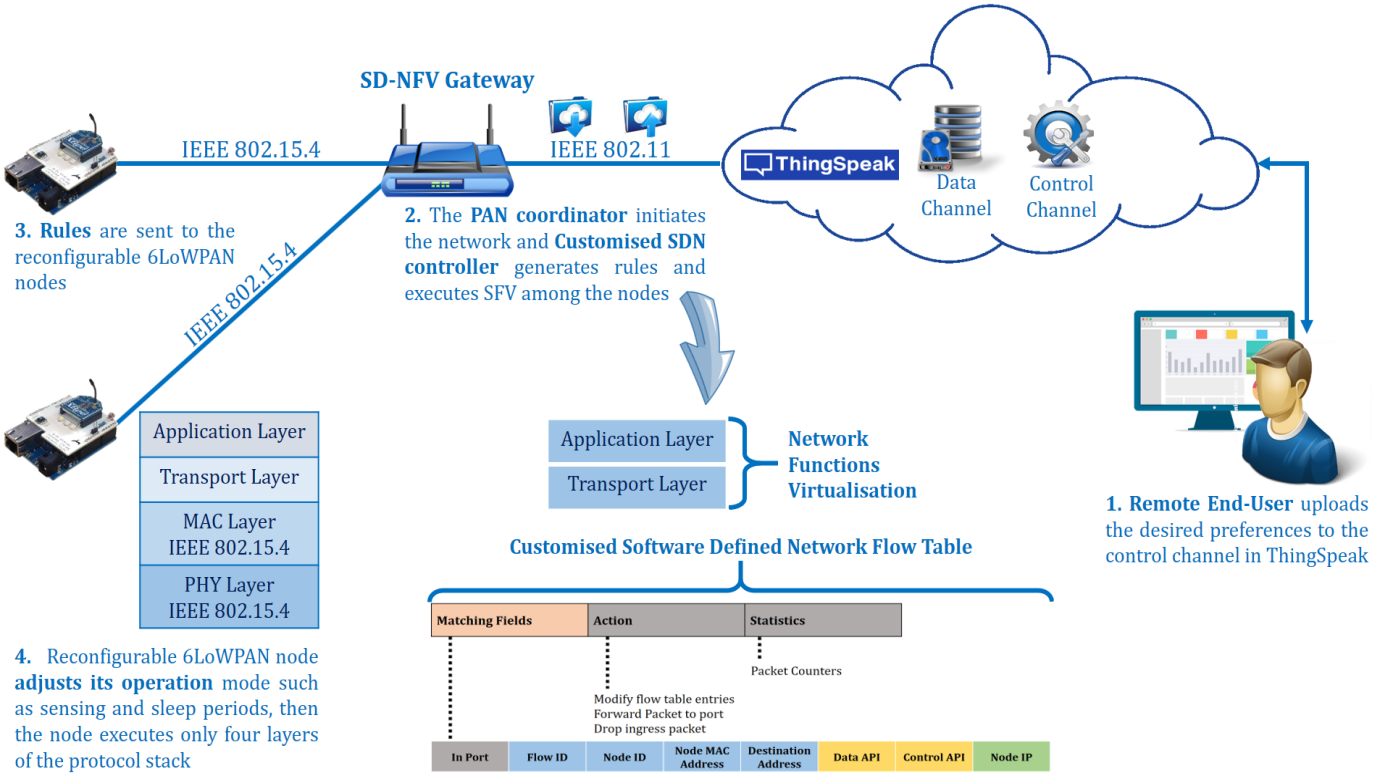

Fig. 9. The implemented programmable scheme of the 6LoWPAN sensor network with cloud connectivity

the completion of the network initialization phase, the addresses of the active nodes are reported to the SDN controller by the 6LoWPAN coordinator. Thereafter, the tailored SDN controller assigns an IP address to each node and stores this entry in the alive node table. The SDN controller is able to setup the flow table to every 6LoWPAN node with the corresponding IP assignment based on the global knowledge of the network topology, as depicted in Fig. 9. The topology discovery manager of the proposed approach executes the nodes discovery function periodical, however, the alive node table will not be modified unless a new node associated with or dissociated from 6LoWPAN network. In summary, the introduced SD-NFV scheme decreases the node discovery time beside increases the energy conservation in 6LoWPAN nodes throughout network initiation and nodes discovery phases.

Fig. 10 shows the detailed implementation of the flow table at the 6LoWPAN gateway and it is continued in Fig. 11. The flow table contains the list of rules to perform certain actions depending on the ingress and egress flow entities. In the proposed SD-NFV testbed, the control traffic from the controller to the data plane (i.e., downstream and upstream traffics) contains three actions: forward, modify-state (configuration), and drop.

In SDN devices, the flow tables are the essential data structures that authorize these devices to appraise ingress packet and perform the relevant action based on packet's information. At least, one flow-entry per flow is needed to be installed in each 6LoWPAN advanced node by the tailored controller. Each flow in the flow tables has a simple action associated with it. After the network discovery phase, the tailored SDN controller retrieves information from the PAN coordinator to make intelligent control plane decisions. Upon the arrival of the first packet at the controller, the controller determines a path for the flow then setup a new flow entry in the flow tables. Then the tailored SDN controller disseminate the flow tables to the 6LoWPAN advanced nodes which function as SDN switches. The switches monitor the incoming packets and check the matching field to perform the associated action. If no match is existed, the switch sends the packet to the controller in order to determine how to handle the packet. The controller will insert new flow entries and update the switch flow table. Accordingly, the controller's burden will be minimized and the network became programmable by the applications that are running on top of the tailored SDN controller. The proposed approach enhances the network control and visibility because immediate issues are reported directly to the controller.

We believe that this paper provides an exciting opportunity to advance our knowledge in relation to leveraging of SDN, NFV, and cloud computing technologies in the 6LoWPAN sensor node. The modified version of the SD-NFV scheme prolongs the network lifetime by normalizing the energy consumption with the quality of information and reduces end-to-end delay by shorten flow-table construction time. Multi-vendor compatibility for WSN can be achieved by adopting the tailored SDN controller with NFV technology and the integrated cloud platform. These technologies enable smooth protocol evaluation and implementation. The identified research gap will be bridged by the SD-NFV scheme, which provides hardware-independent approach with on-demand programmable component execution.

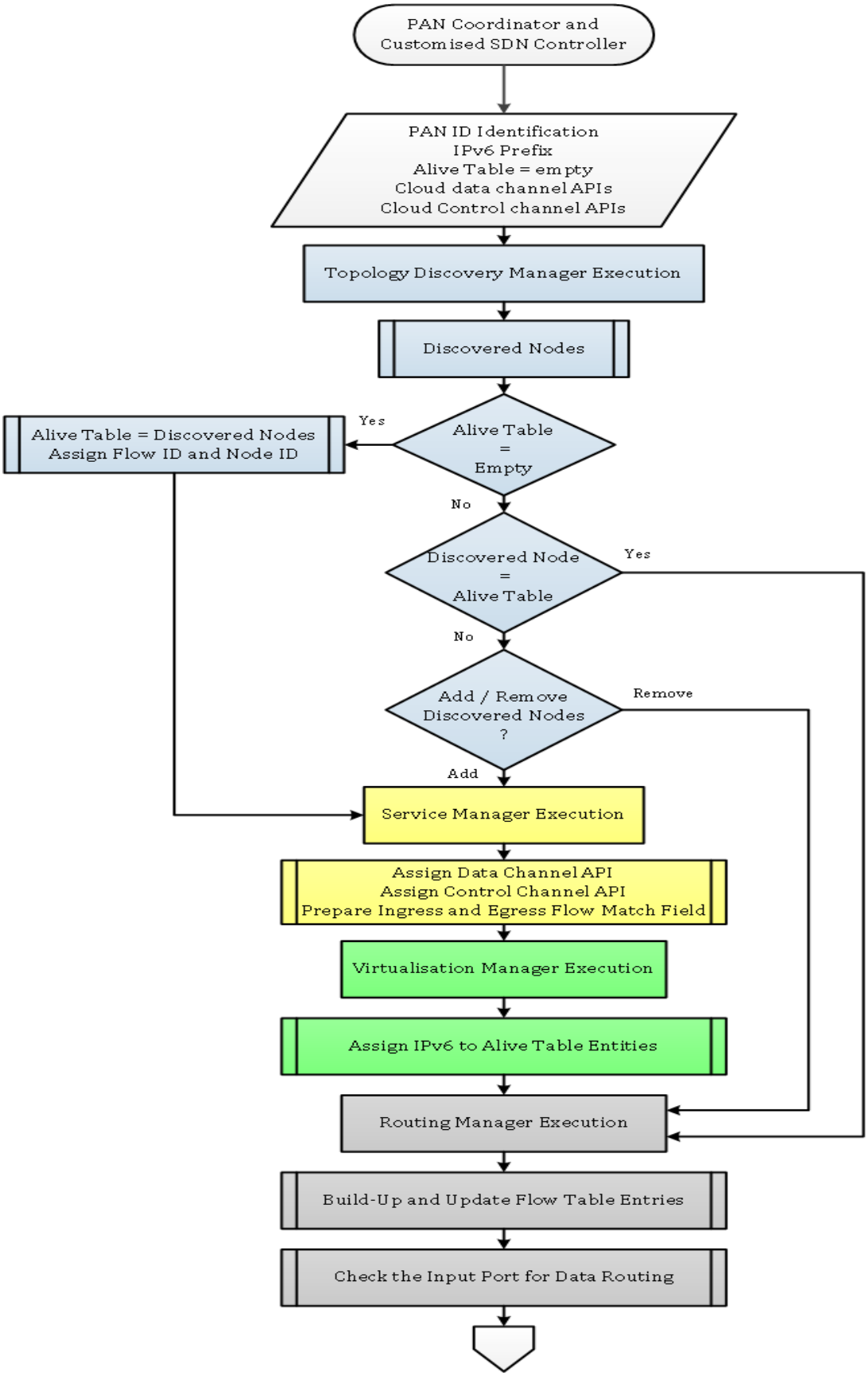

Fig. 10. Tailored SDN controller flow chart

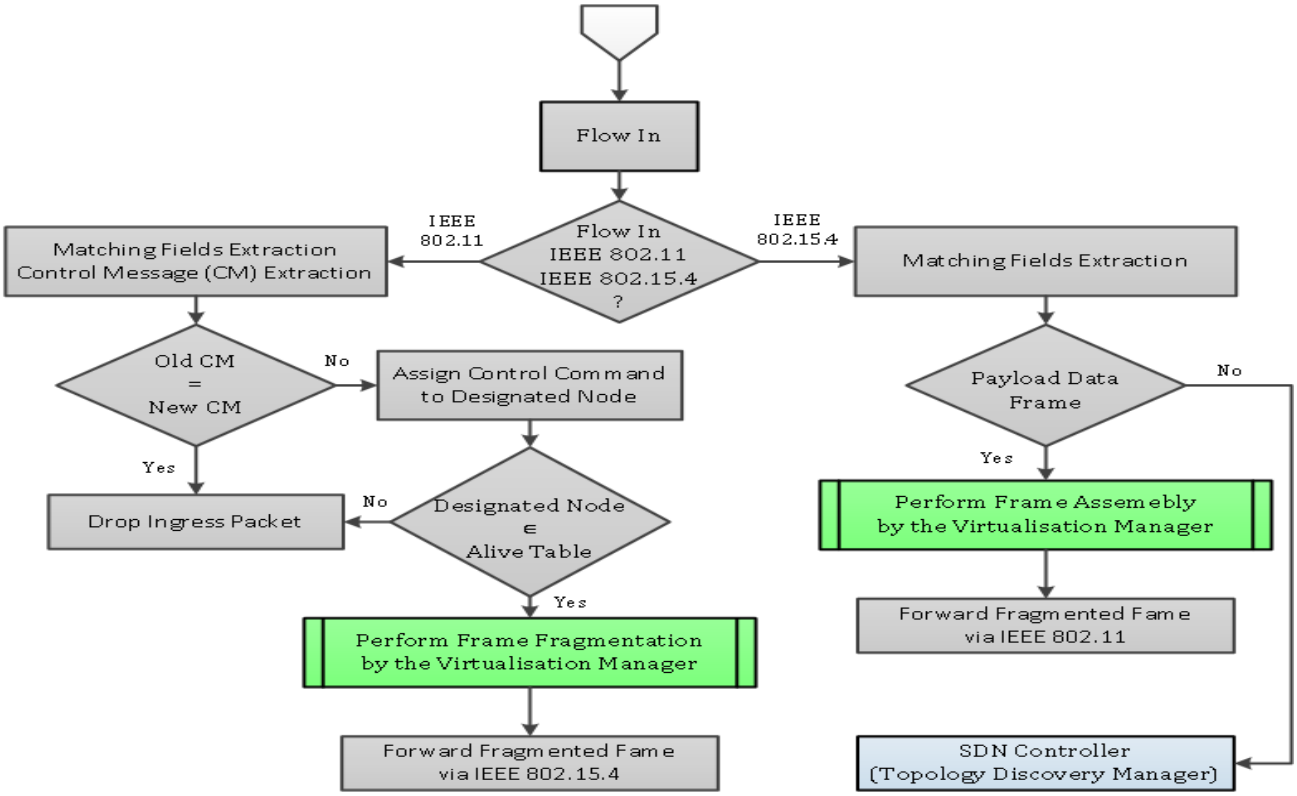

Fig. 11. Tailored SDN controller flow chart (continued)

C. Enabling of Cloud Connectivity to 6LoWPAN Gateway

ThingSpeak is an open source cloud platform for IoT applications that uses APIs to store and retrieve things data using the Hypertext Transfer Protocol (HTTP) over the Internet. ThingSpeak was chosen because it enables the users to create sensor-logging applications with status updates and to analyze and visualize the uploaded data using MATLAB. The proposed modified SD-NFV approach has been integrated with ThingSpeak through the 6LoWPAN gateway.

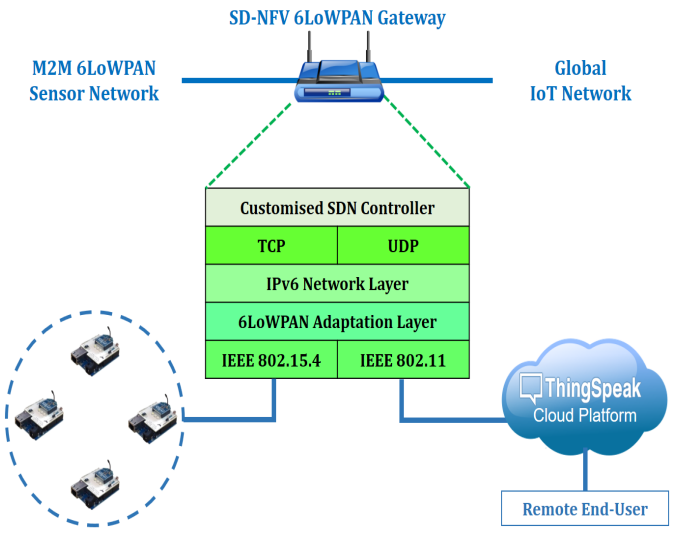

Fig. 12. Cloud-based 6LoWPAN border router

Fig. 12 shows the interoperation of services and network connectivity for the introduced approach. Two types of ThingSpeak channels are introduced: the data and control channels. The 6LoWPAN sensor nodes data are stored in their corresponding data channels that are assigned by the tailored SDN controller, while the control messages are sent to a particular 6LoWPAN node via the control channels over the IP network. The main purpose of these channels is to ensure that the 6LoWPAN gateway supports dual communication path for the sensor network. The cloud architecture has two ends: the front- and back-end. The front-end is available for the user to execute various algorithms and to modify network settings and options. Whilst the back-end is located near the 6LoWPAN nodes to provide ubiquitous connectivity through the border gateway via the Internet.

D. End-User Monitoring Application

In order to simulate external IP accessibility to 6LoWPAN sensor network, the MATLAB software has been used to execute an elementary end-user monitoring application. The application primarily is used to retrieve the sensed data which are already stored in ThingSpeak and to analyze these data on a remote PC. Furthermore, it is able to send control messages to a specified 6LoWPAN node (altering the state of the connected LED on/off) in order to demonstrate the IP accessibility in a heterogeneous network. Fig. 13 shows the Graphical User Interface (GUI) of the remote monitoring application. The remote application works as follows: it retrieves the sensed data that stored in the ThingSpeak platform through the data channels and visualizes it on a remote PC, then based on

these readings, the end-user forwards the modified settings of the sensor network to the border router through the control channels of the ThingSpeak platform via the Internet.

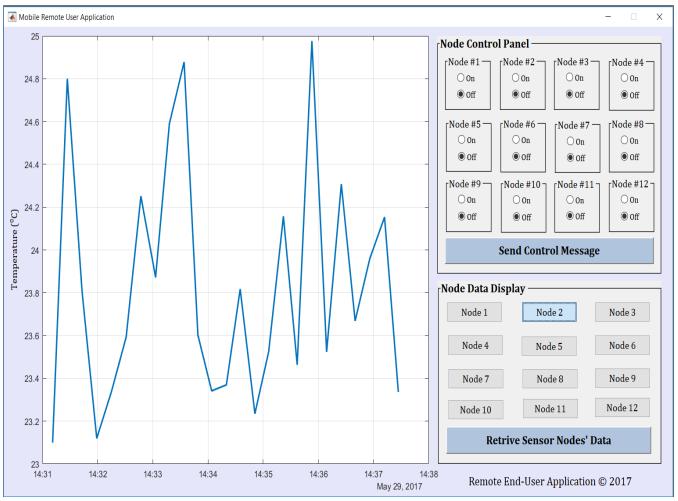

Fig. 13. Remote monitoring application

VI. Performance Evaluation Results

This section discusses the obtained results from the proof-of-concept M2M testbed that integrates SDN and NFV in 6LoWPAN testbed with cloud connectivity. The experimental scenarios were conducted in an indoor LAB environment at Brunel University London (UK) and Al-Iraqia University (Iraq). The 6LoWPAN PAN coordinator initiates two sequential processes that run concurrently with the SDN controller. First, the topology discovery manager is executed prior to any sensing task in order to construct the global topology of the 6LoWPAN network at the tailored SDN controller. Thereafter, the second process is commenced by running simple sensing function in every 6LoWPAN node and disseminating the measured phenomena to the 6LoWPAN edge router before being stored on the cloud platform (ThingSpeak), which enables the SD-NFV approach to have an ubiquitous connectivity among the remote end-users.

It is resource intensive and strenuous to detect all the alive nodes manually in a 6LoWPAN network. Hence, a cognitive mechanism for global 6LoWPAN topology discovery is developed to scan the state of every alive node in the sensor network, which is accredited to the PAN coordinator. It and the tailored SDN controller are merged together, which enables them to work with consistency and thus, reduces the communication overhead between the nodes. By the same token, high bandwidth utilization with respect to a traditional (non SDN-enabled) 6LoWPAN network can be achieved by the proposed node discovery function in which the energy consumed by communication is more than that of data processing and environmental sensing. The information in the flow-tables can be used to predict the behavior of a 6LoWPAN network during its operational lifetime.

The global topology discovery mechanism aimed at attaining minimum packet exchanges between the sensor nodes to maintain their connectivity with the 6LoWPAN coordinator. Both the tailored controller and the PAN coordinator are delegated by the discovery manager to be in charge of looking after nodes status updates across the entire network. The nodes status updates are propagated to the tailored SDN controller to build-up the flow-tables for the entire network. The tailored SDN controller is responsible of creating two individual tables that contain all the up-to-date information about the network infrastructure. The first table comprises the alive 6LoWPAN nodes that are associated with the PAN coordinator, and the second table includes the network flow-table that includes the node to IP and ThingSpeak APIs assignments.

We go a step further in evaluating the performance of the tailored controller that is responsible for building the control plane rules as well as tracing the packets at the data plane. Accordingly, the flow-table installation latency and data delivery metrics have been considered in SDN performance evaluation. Fig. 14 pinpoint the most important metric that influence the speed of data delivery process. The flow-table latency increases as the number of 6LoWPAN nodes joined the controller. Therefore, the rules update time will increase accordingly. The 10 rules indicate that the network consists of 4 simple nodes and 1 advance node while 100 rules stands for the network of 12 simple nodes and 12 advanced nodes. The proposed tailored SDN controller ensures a lower and constant rate to modify the existing rules or insert new rules in the current flow-table. This firmness is a result of the developed systematic way that takes into account the total flow-table size, and the cost of modifying the table with new rules.

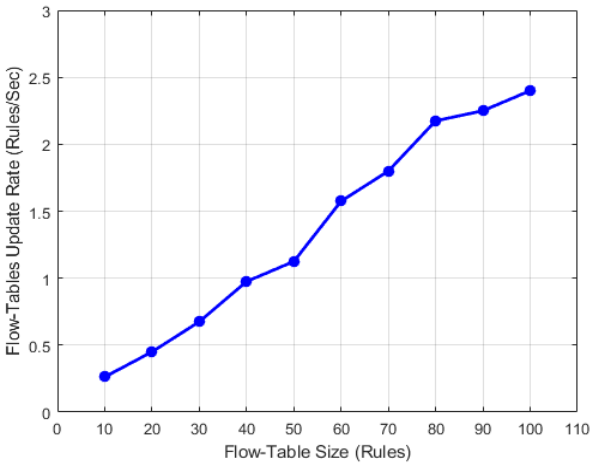

Fig. 14. The flow-tables update latency

The term end-to-end delay in this paper is defined as the delay from the moment a source node has a data packet ready to send until the moment it reaches the application layer of the destination node. Due to the nature of a 6LoWPAN link, giving delay guarantees is not always straightforward. The end-to-end delay is based on multiple tentative variables. That is, it contains all potential delays during packet generation, propagation, route discovery latency, transfer times and delays of retransmission at the MAC layer until it is successfully received by the application layer of the destination node. Fig. 15 shows the end-to-end delay between the 6LoWPAN sensor nodes and the 6LoWPAN gateway for both the proposed

and the traditional approaches. It is clear that SD-NFV has lower delay in collecting the sensors' data as compared to a traditional 6LoWPAN (non SDN-enabled) network. In the developed SD-NFV approach, the end-to-end delay increases as the number of sensor nodes increases, because the packet waiting time in the SDN controller queue is raised. On the other hand, a traditional 6LoWPAN network has higher delay than the SD-NFV approach for varying numbers of connected nodes. In a non SDN-enabled 6LoWPAN network, the sending rates of the sensor nodes are increased periodically to enable the sensor nodes transmitting the fragmented IPv6 packets over the MAC and PHY layers of the IEEE 802.15.4 standard. As a result, the fragmented packets wait a longer time in the nodes' buffers and hence, the end-to-end delay increases. In summary, the proposed SD-NFV approach enhances end-to-end delay for reading the sensors' data by ≈160% in comparison to the traditional (non SDN-enabled) 6LoWPAN network.

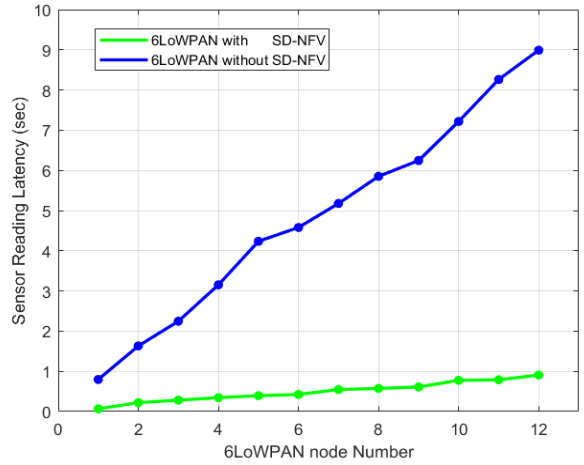

Fig. 15. Sensor reading latency

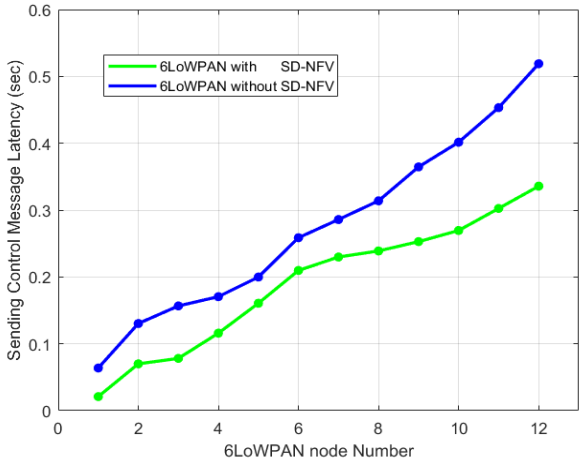

Fig. 16. Sending control command latency

Fig. 16 shows the other path of communication that exists in the 6LoWPAN based networks, which pertains to the data communication between the remote user and the sensor node via the connected border gateway. End-to-end delay for sending control messages from the 6LoWPAN gateway to the sensor nodes slightly increases when more packets come back and forth between the 6LoWPAN nodes, but this remains very low and does not vary much as the number of connected nodes is increased. In the traditional 6LoWPAN network, end-to-end delay is high when compared to the SD-NFV approach, which occurs because a larger number of packets are being transmitted to each individual node. In summary, the proposed SD-NFV approach improves end-to-end delay for sending control messages to sensor nodes by ≈63% in comparison to the traditional (non SDN-enabled) 6LoWPAN network.

Fig. 17 shows the end-to-end latency for both the proposed SD-NFV and traditional 6LoWPAN approaches in terms of gateway and remote end-user connectivity with the cloud platform (ThingSpeak). There is a slight variance between the two approaches regarding end-user connectivity with the cloud platform, because it uses HTTP for data sending and retrieval. This disparity may result from the Internet connection speed owing to the varying end-to-end latency values during the testbed runtime. On the other hand, there is a considerable difference in the end-to-end latencies between the proposed SD-NFV and traditional 6LoWPAN approaches. This disproportion results from the way that the 6LoWPAN gateway transports and stores the sensed data on ThingSpeak.

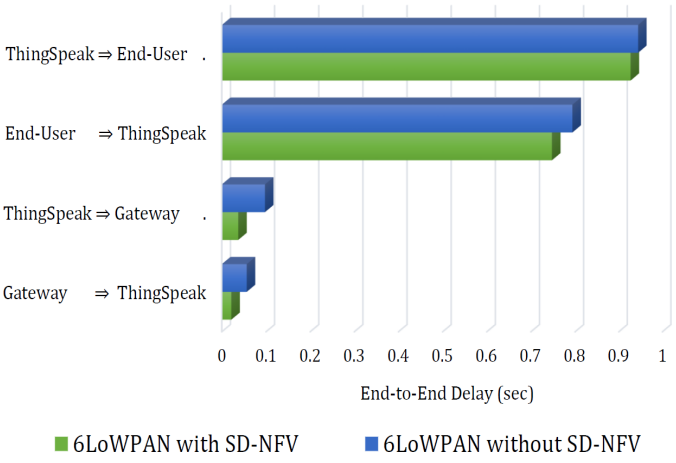

Fig. 17. End-to-End connectivity delay

Upon the execution of the simple sensing function in every 6LoWPAN nodes, the boarder router will be in charge of ingress and egress traffic refinement for the sensor network. The 6LoWPAN gateway will perform assembly or fragmentation tasks for the received packets based on the collected transceiver. The network lifetime substantially relies on the individual nodes lifetime that establish the sensor network. The lifetime is one of the important metrics that are used to evaluate the performance of a specific application or algorithm because changing or recharging the nodes' battery is not feasible in many scenarios. Generally, it is defined as maximum duration of time that a 6LoWPAN node would be considered fully functional. Two factors affects the sensor node operational lifespan: how much residual energy is available for node's usage and how much energy it exhausts over time. Current analysis of node lifetime focuses only on the advanced

node energy consumption as it can be run as a cluster in cluster-based topologies. Compared to traditional (non SDN-enabled) approach, the introduced approach in this paper extends the advanced 6LoWPAN node lifetime by ≈70%, as presented in Fig. 18 for theoretical and testbed results.

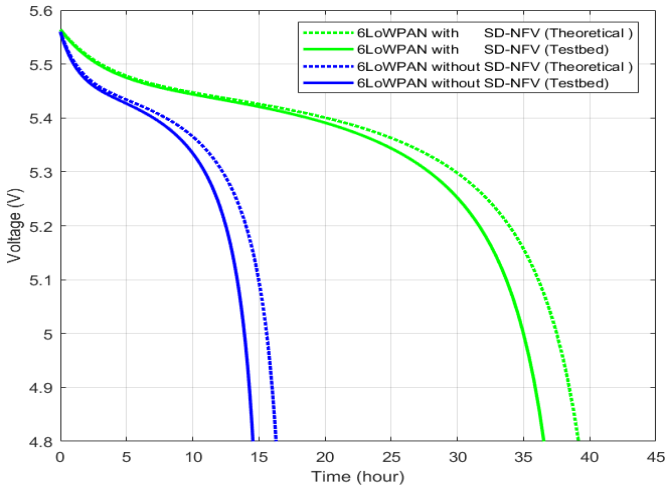

Fig. 18. The operational lifespan of the advanced 6LoWPAN node

The theoretical battery lifetime is modeled and evaluated using MATLAB/Simulink 2018a, as shown in Fig. 19. The model is based on the same parameters of the power banks that are used to turn on the advanced nodes, whilst the 6LoWPAN node is modeled as variable resistor load with time to emulate the sensor node activity. In summary, the sensor node joins the 6LoWPAN gateway that runs the SD-NFV approach can exhaust its energy wisely, as superfluous IPv6 packet transmissions are excluded (i.e. IPv6 headers). The virtualized layers of the 6LoWPAN protocol stack facilitate the utilization of a low-energy sleep mode in the M2M sensor nodes to sustain the available energy for a longer period.

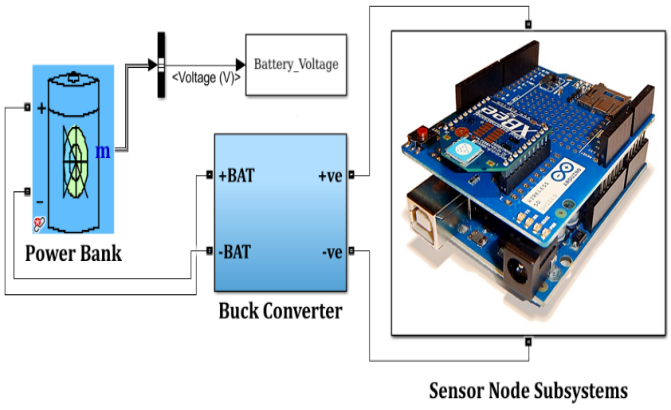

Fig. 19. 6LoWPAN sensor node model

It is possible to calculate the approximate autonomous time (AT) as the ratio between the battery capacity (BC) and the device power consumption (DPC) [56]. In general, the theoretical battery discharge curve that is shown in Fig. 18 was calculated using Eq. 1 within the MATLAB/Simulink environment. A lithium-ion polymer battery with 4000 mAh capacity was used to power the advanced node and the same specifications was adopted in the proposed battery model to reflect the same testbed conditions in order to obtain reliable and trusted results.

$$AT = \frac{BC[mAh]}{DPC[mA]} \tag{1}$$

Fig. 20 and Fig. 21 show the data delivery rate for both SDN and non SDN-enabled 6LoWPAN sensor networks respectively. For small network size, both 6LoWPAN networks have approximately similar data delivery rate, while as the number of connected nodes increases more variations were appeared. These variations in network performance in terms of data delivery rate can be interpreted as the sensor nodes far away from the PAN coordinator would not join the network. Therefore, the farthest nodes will not be able to deliver data packets and decrease the overall delivery rate. However, the proposed flow-table management policy aimed at improving the scalability by modifying the data flow path in the flow-tables entries. In addition, the long lifetime of the sensor nodes that adopts SD-NFV approach will increase the data delivery rate by approximately 5-14% depending on the network size.

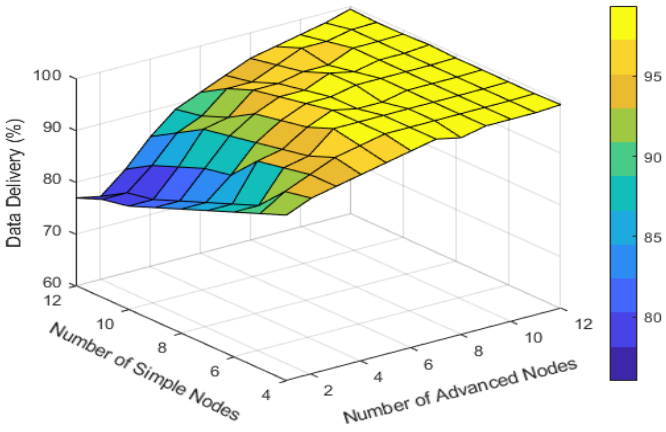

Fig. 20. Data delivery results for SDN enabled 6LoWPAN sensor network

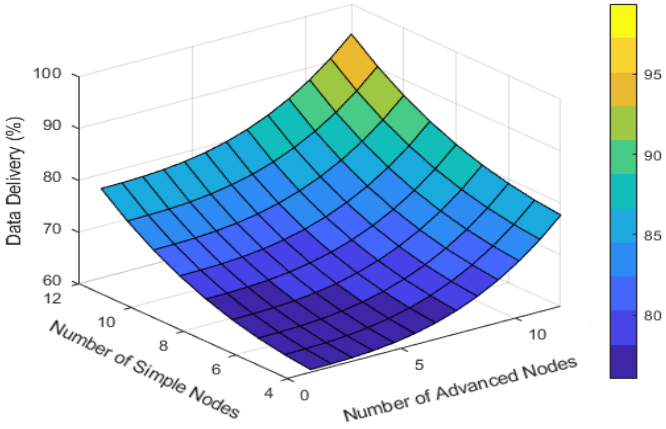

Fig. 21. Data delivery results for non SDN-enabled 6LoWPAN sensor network

We realize that an Arduino board draws relatively higher current than other existing microcontroller boards, thus only working for shorter periods. However, they were selected so as to accelerate the observation process of the implemented approach, thereby obtaining the testbed results within a brief time period.

VII. Conclusion

A proof-of-concept real-time testbed has been implemented to study the impact of programmable network techniques (i.e. SDN and NFV) on an IEEE 802.15.4-based sensor network (i.e. 6LoWPAN) with integrated cloud service. In addition, the implemented testbed is aimed at tackling the extant challenges of engaging SDN and NFV in IPv6 M2M sensor nodes based on the IEEE 802.15.4 standard. Simple sensing application is executed in the current testbed with a particular programmable network approach, being called Software Defined-Network Function Virtualization (SD-NFV).

The main objectives of the proposed SD-NFV architecture are to enhance end-to-end delay and to improve node energy consumption during their lifetime with acceptable flow-table update latency. The SD-NFV approach outperforms the traditional (non SDN-enabled) 6LoWPAN network in terms of network discovery and end-to-end delay. In the implemented 6LoWPAN testbed, the network data delivery ratio is improved by 5-14%, and the 6LoWPAN node operational time prolonged by 70%, when compared to the traditional (non SDN-enabled) approach. These achievements of the introduced approach are obtained, because the proposed SD-NFV architecture abstracts the most profligate energy layers from the sensor node protocol stack and virtualized them in the 6LoWPAN gateway. The virtualized scheme makes the functions of both layers visible in association with the remainder 6LoWPAN nodes via the tailored SDN controller.

The end-to-end delay is also enhanced by the SD-NFV approach. Specifically, the end-to-end latency for reading sensor data by the 6LoWPAN gateway is reduced by $\approx$160% and the latency for sending a control message to a specified node is minimized by $\approx$63% when compared to a traditional 6LoWPAN network. These reductions in end-to-end delay of a 6LoWPAN network were obtained, because the SDN controller eliminates the need for multiple control packet exchange to maintain node connectivity and the virtualized layers eliminate the transmission of a large number of fragmented packets over the IEEE 802.15.4 standard.

References

[1] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016-2021," 2017, White Paper, [Accessed on September 2017].

[2] N. C. Luong, D. T. Hoang, P. Wang, D. Niyato, D. I. Kim, and Z. Han, "Data Collection and Wireless Communication in Internet of Things (IoT) Using Economic Analysis and Pricing Models: A Survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 4, pp. 2546–2590, Fourthquarter 2016.

[3] V. Gazis, "A Survey of Standards for Machine-to-Machine and the Internet of Things," *IEEE Communications Surveys Tutorials*, vol. 19, no. 1, pp. 482–511, Firstquarter 2017.

[4] A. H. Ngu, M. Gutierrez, V. Metsis, S. Nepal, and Q. Z. Sheng, "IoT Middleware: A Survey on Issues and Enabling Technologies," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 1–20, Feb 2017.

[5] G. A. Akpakwu, B. J. Silva, G. P. Hancke, and A. M. Abu-Mahfouz, "A Survey on 5G Networks for the Internet of Things: Communication Technologies and Challenges," *IEEE Access*, vol. 6, pp. 3619–3647, 2018.

[6] A. Aijaz and A. Aghvami, "Cognitive Machine-to-Machine Communications for Internet-of-Things: A Protocol Stack Perspective," *Internet of Things Journal, IEEE*, vol. 2, no. 2, pp. 103–112, April 2015.

[7] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *Communications Surveys Tutorials, IEEE*, vol. 17, no. 4, pp. 2347–2376, Fourthquarter 2015.

[8] N. Correia, D. Sacramento, and G. Schütz, "Dynamic Aggregation and Scheduling in CoAP/Observe-Based Wireless Sensor Networks," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 923–936, Dec 2016.

[9] K. Kwon, M. Ha, T. Kim, S. H. Kim, and D. Kim, "The Stateless Point to Point Routing Protocol Based on Shortcut Tree Routing Algorithm for IP-WSN," in *2012 3rd IEEE International Conference on the Internet of Things*, Oct 2012, pp. 167–174.

[10] IETF, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks, RFC 4944, Network Working Group," https://tools.ietf.org/html/rfc4944, 2007, [Accessed on September 2017].

[11] D. L. Tennenhouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall, and G. J. Minden, "A Survey of Active Network Research," *IEEE Communications Magazine*, vol. 35, no. 1, pp. 80–86, Jan 1997.

[12] A. T. Campbell, I. Katzela, K. Miki, and J. Vicente, "Open Signaling for ATM, Internet and Mobile Networks (OPENSIG'98)," *SIGOPS Oper. Syst. Rev.*, vol. 33, no. 2, pp. 15–28, Apr. 1999.

[13] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "Ethane: Taking Control of the Enterprise," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 1–12, Aug. 2007.

[14] IETF, "Forwarding and Control Element Separation (ForCES) Protocol Specification," https://tools.ietf.org/html/rfc5810, 2010, [Accessed on September 2017].

[15] N. Bizanis and F. A. Kuipers, "SDN and Virtualization Solutions for the Internet of Things: A Survey," *IEEE Access*, vol. 4, pp. 5591–5606, 2016.

[16] J. G. Herrera and J. F. Botero, "Resource Allocation in NFV: A Comprehensive Survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, Sept 2016.

[17] C. Wang, Z. Bi, and L. D. Xu, "IoT and Cloud Computing in Automation of Assembly Modeling Systems," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1426–1434, May 2014.

[18] S. Mubeen, P. Nikolaidis, A. Didic, H. Pei-Breivold, K. Sandström, and M. Behnam, "Delay Mitigation in Offloaded Cloud Controllers in Industrial IoT," *IEEE Access*, vol. 5, pp. 4418–4430, 2017.

[19] A. Mahmud, R. Rahmani, and T. Kanter, "Deployment of Flow-Sensors in Internet of Things' Virtualization via OpenFlow," in *Mobile, Ubiquitous, and Intelligent Computing (MUSIC), 2012 Third FTRA International Conference on*, June 2012, pp. 195–200.

[20] T. Luo, H. P. Tan, and T. Q. S. Quek, "Sensor OpenFlow: Enabling Software-Defined Wireless Sensor Networks," *IEEE Communications Letters*, vol. 16, no. 11, pp. 1896–1899, November 2012.

[21] S. Costanzo, L. Galluccio, G. Morabito, and S. Palazzo, "Software Defined Wireless Networks: Unbridling SDNs," in *2012 European Workshop on Software Defined Networking*, Oct 2012, pp. 1–6.

[22] A. D. Gante, M. Aslan, and A. Matrawy, "Smart Wireless Sensor Network Management Based on Software-Defined Networking," in *2014 27th Biennial Symposium on Communications (QBSC)*, June 2014, pp. 71–75.

[23] C. P. Kruger, A. M. Abu-Mahfouz, and G. P. Hancke, "Rapid Prototyping of a Wireless Sensor Network Gateway for the Internet of Things Using Off-The-Shelf Components," in *2015 IEEE International Conference on Industrial Technology (ICIT)*, March 2015, pp. 1926–1931.

[24] A. Krylovskiy, "Internet of Things Gateways Meet Linux Containers: Performance Evaluation and Discussion," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, Dec 2015, pp. 222–227.

[25] R. Morabito and N. Beijar, "Enabling Data Processing at the Network Edge through Lightweight Virtualization Technologies," in *2016 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops)*, June 2016, pp. 1–6.

[26] R. Petrolo, R. Morabito, V. Loscrì, and N. Mitton, "The Design of the Gateway for the Cloud of Things," *Annals of Telecommunications*, vol. 72, no. 1, pp. 31–40, 2017. [Online]. Available: http://dx.doi.org/10.1007/s12243-016-0521-z

[27] I. Khan, F. Belqasmi, R. Glitho, N. Crespi, M. Morrow, and P. Polakos, "Wireless Sensor Network Virtualization: A Survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 553–576, Firstquarter 2016.

[28] I. Khan, F. Belqasmi, R. Glitho, and N. Crespi, "A Multi-Layer Architecture for Wireless Sensor Network Virtualization," in *6th Joint IFIP Wireless and Mobile Networking Conference (WMNC)*, April 2013, pp. 1–4.

[29] B. R. Al-Kaseem and H. S. Al-Raweshidy, "SD-NFV as an Energy Efficient Approach for M2M Networks Using Cloud-Based 6LoWPAN Testbed," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2017.

[30] M. Yang, Y. Li, D. Jin, L. Zeng, X. Wu, and A. V. Vasilakos, "Software-Defined and Virtualized Future Mobile and Wireless Networks: A Survey," *Mobile Networks and Applications*, vol. 20, no. 1, pp. 4–18, 2015. [Online]. Available: http://dx.doi.org/10.1007/s11036-014-0533-8

[31] Y. Li and M. Chen, "Software-Defined Network Function Virtualization: A Survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015.

[32] K. Sood, S. Yu, and Y. Xiang, "Software-Defined Wireless Networking Opportunities and Challenges for Internet-of-Things: A Review," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 453–463, Aug 2016.

[33] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *Communications Surveys Tutorials, IEEE*, vol. 17, no. 4, pp. 2347–2376, Fourthquarter 2015.

[34] IEEE 802.15.4-2003, "IEEE Standard for Telecommunications and Information Exchange Between Systems - LAN/MAN Specific Requirements - Part 15: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPAN)," https://standards.ieee.org, 2003, [Accessed on September 2017].

[35] IEEE 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003), "IEEE Standard for Information Technology - Local and Metropolitan Area Networks - Specific Requirements - Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs)," https://standards.ieee.org, 2006, [Accessed on September 2017].

[36] Z. Shelby and C. Bormann, *6LoWPAN: The Wireless Embedded Internet*. Wiley Publishing, 2010.

[37] Y. Qiu and M. Ma, "A Mutual Authentication and Key Establishment Scheme for M2M Communication in 6LoWPAN Networks," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 6, pp. 2074–2085, Dec 2016.

[38] S. H. Yang, *Wireless Sensor Networks: Principles, Design and Applications*. Springer Publishing Company, Incorporated, 2013.

[39] A. Lara, A. Kolasani, and B. Ramamurthy, "Network Innovation Using OpenFlow: A Survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 493–512, First 2014.

[40] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 3, pp. 1617–1634, Third 2014.

[41] W. Stallings, *Foundations of Modern Networking: SDN, NFV, QoE, IoT, and Cloud*. Pearson Education, 2016.

[42] Open Networking Foundation, https://www.opennetworking.org/, [Accessed on September 2017].

[43] J. Doherty, *SDN and NFV Simplified: A Visual Guide to Understanding Software Defined Networks and Network Function Virtualization*, 1st ed. Addison-Wesley Professional, 2016.

[44] B. R. Al-Kaseem and H. S. Al-Raweshidy, "Enabling Wireless Software Defined Networking in Cloud based Machine-to-Machine Gateway," in *2016 8th Computer Science and Electronic Engineering (CEEC)*, Sept 2016, pp. 24–29.

[45] T. D. Nadeau and K. Gray, *SDN: Software Defined Networks*. "O'Reilly Media, Inc.", 2013.

[46] F. Hu, *Network Innovation Through OpenFlow and SDN: Principles and Design*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 2014.

[47] S. Al-Rubaye, E. Kadhum, Q. Ni, and A. Anpalagan, "Industrial Internet of Things Driven by SDN Platform for Smart Grid Resiliency," *IEEE Internet of Things Journal*, vol. Early Access Article, pp. 1–1, 2018.

[48] F. Bannour, S. Souihi, and A. Mellouk, "Distributed SDN Control: Survey, Taxonomy, and Challenges," *IEEE Communications Surveys Tutorials*, vol. 20, no. 1, pp. 333–354, Firstquarter 2018.

[49] R. Chayapathi, S. F. Hassan, and P. Shah, *Network Functions Virtualization (NFV) with a Touch of SDN*, 1st ed. Indianapolis, Indiana, USA: Pearson Education, Inc., 2016.

[50] Q. Duan, N. Ansari, and M. Toy, "Software-Defined Network Virtualization: an Architectural Framework for Integrating SDN and NFV for Service Provisioning in Future Networks," *IEEE Network*, vol. 30, no. 5, pp. 10–16, September 2016.

[51] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network Function Virtualization: State-of-the-Art and Research Challenges," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 236–262, Firstquarter 2016.

[52] C. Mouradian, T. Saha, J. Sahoo, R. Glitho, M. Morrow, and P. Polakos, "NFV Based Gateways for Virtualized Wireless Sensor Networks: A Case Study," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, June 2015, pp. 1883–1888.

[53] ZigBee Alliance, "Zigbee Overview," http://www.zigbee.org, 2003, [Accessed September 2017].

[54] I. Muller, J. C. Netto, and C. E. Pereira, "WirelessHART Field Devices," *IEEE Instrumentation Measurement Magazine*, vol. 14, no. 6, pp. 20–25, December 2011.

[55] Arduino pico IPv6 stack (pIPv6), https://github.com/telecombretagne/Arduino-pIPv6Stack, [Accessed on September 2017].

[56] M. Pulpito, P. Fornarelli, C. Pomo, P. Boccadoro, and L. A. Grieco, "On Fast Prototyping LoRaWAN: a Cheap and Open Platform for Daily Experiments," *IET Wireless Sensor Systems*, vol. 8, no. 5, pp. 237–245, 2018.