SD–NFV as an Energy Efficient Approach for M2M Networks Using Cloud–Based 6LoWPAN Testbed

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

***Abstract*—Machine–to–Machine (M2M) communication is the leading technology for realising the Internet–of–Things (IoT). The M2M sensor nodes are characterised by low–power and low–data rates devices which have increased exponentially over the years. IPv6 over Low power Wireless Personal Area Network (6LoWPAN) is the first protocol that provides IPv6 connectivity to the wireless M2M sensor nodes. Having a tremendous number of M2M sensor nodes execute independent control decision leads to difficulty in network control and management. In addition, these ever–growing devices generate massive traffic and cause energy scarcity which affects the M2M sensor node lifetime. Recently, Software–Defined Networking (SDN) and Network Functioning Virtualisation (NFV) are being used in M2M sensor networks to add programmability and flexibility features in order to adopt the exponential increment in wireless M2M traffic and enable network configuration even after deployment. This paper presents a proof–of–concept implementation which aims to analyse how SDN, NFV, and cloud computing can interact together in the 6LoWPAN gateway to provide simplicity and flexibility in network management. The proposed approach is called customised Software Defined–Network Functioning Virtualisation (SD–NFV), and has been tested and verified by implementing a real–time 6LoWPAN testbed. The experimental results indicated that the SD–NFV approach reduced the network discovery time by 60% and extended the node's lifetime by 65% in comparison to the traditional 6LoWPAN network. The implemented testbed has one sink which is the M2M 6LoWPAN gateway where the network coordinator and the SDN controller are executed. There are many possible ways to implement 6LoWPAN testbed but limited are based on open standards development boards (e.g., Arduino, Raspberry Pi, and Beagle Bones). In the current testbed, the Arduino board is chosen and the SDN controller is customised and written using C++ language to fit the 6LoWPAN network requirements. Finally, SDN and NFV have been envisioned as the most promising techniques to improve network programmability, simplicity, and management in cloud–based 6LoWPAN gateway.**

***Index Terms*—Energy Efficiency; Customised SD–NFV; M2M; IoT; SDN; NFV; Cloud Computing; 6LoWPAN Testbed.**

I. INTRODUCTION

ONLY a few years from now, by 2020, the number of devices connected to the Internet will increase exponentially [1]. The connected devices will be quite diverse in functionality and processing capability, having the ability to sense, actuate, process, and store data. These devices can communicate with each other and exchange information in a Machine–to–Machine (M2M) paradigm [2]. M2M communication refers to the communication between two connected devices in homogeneous or heterogeneous networks without or with limited human intervention. M2M communication constitutes the principle communication paradigm in realising the IoT revolution [3]. The IoT enables physical objects to have virtual identity and will be integrated into a wide range of applications to enhance daily life activities, such as home and industrial automation, healthcare monitoring, energy management, etc. Moreover, cloud computing architectures are the most promising technology in leveraging some of the applications, services, and networks of the IoT [4].

M2M sensor networks are composed mainly of a large number of small devices that run on batteries. The limited battery power of the M2M node is consumed during the node's lifetime performing the sensing, collecting, and transmitting of data. According to the limited energy source, there is a need to balance the energy consumption and the quality of information, because the lifetime of any M2M node depends on the availability of the residual energy. Energy efficiency is an important characteristic for battery powered wireless networks. Therefore, the energy efficiency of the M2M sensor network has emerged as a major research issue and drawn considerable interest from both industry and academia [5].

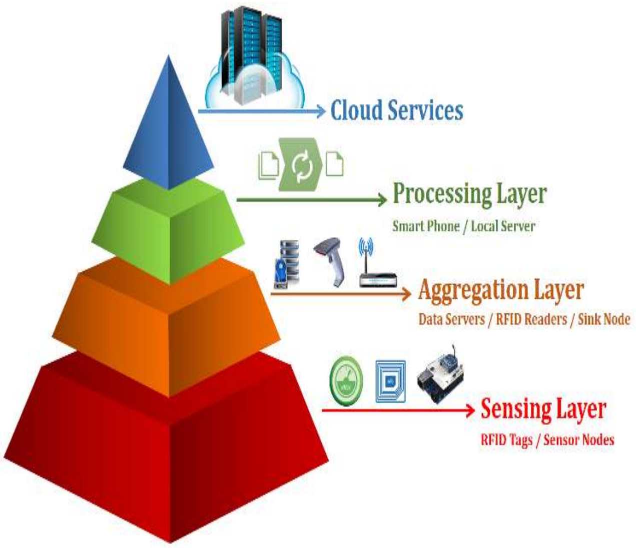


Fig. 1. The four layers of the IoT architecture

The IoT components are characterised by four layers, as depicted in Fig. 1. The first layer includes all sensor nodes, Radio Frequency Identification (RFID) tags, and is called the

Bilal R. Al-Kaseem and Hamed S. Al-Raweshidy are with the Department of Electrical and Electronic Engineering, College of Engineering, Design and Physical Sciences, Brunel University London, United Kingdom. (e-mail: Bilal.Al-Kaseem@brunel.ac.uk ; Hamed.Al-Raweshidy@brunel.ac.uk)

sensing layer. The data being generated by the sensing layer is collected by the available data aggregators in the second layer. Accordingly, the second layer is called the *aggregator layer*, and the data aggregators could be sink nodes for sensor networks or RFID readers or intermediate local storage. The third layer is the *processing layer*, and it is to here that the aggregators forward their data for further processing. After the data processing is completed, the data can be uploaded to the cloud in *cloud layer* (fourth layer) to provide ubiquitous connectivity for data exchange with anything, anywhere, at any time.

IoT applications are built–up of a large number of M2M nodes. The main objective of the IoT is to have energy efficient and scalable routing protocol to prolong the lifetime of the connected hardware with the physical environment by converting the consumed energy into useful information. The scalability can be achieved by using hierarchical or clustering approach in M2M nodes. In addition, cluster–based network enhances the network lifetime by performing data aggregation in selective nodes in the M2M sensor networks [6].

M2M sensor networks may adopt unexpected topology changes and node mobility, where M2M nodes are being deployed randomly in the target area. In order to improve the availability of shared resources, M2M sensor nodes should have the ability of node reconfiguration even after deployment. Software–Defined Networking (SDN) has been proposed to separate the control plane from the data plane. SDN can enable sensor node re–tasking in M2M networks and it also provides seamless resource management for implementing different algorithms through the centralised controller. [7].

The purpose of Network Function Virtualisation (NFV) is to reduce the operational and capital expenses of the network. NFV can be applied to any packet processing plane (data plane) and route decision plane (control plane) of the M2M network infrastructure. SDN and NFV represent the most promising advances in terms of a programmable network and dynamic resource allocation for M2M networks in IoT architecture [8].

IPv6 over Low power Wireless Personal Area Network (6LoWPAN) has been introduced by the Internet Engineering Task Force (IETF) working group. 6LoWPAN defines the implementation of Internet protocols over low–power, low–data rate devices using the IEEE 802.15.4 as Medium Access Control (MAC) and Physical (PHY) layers standard. The 6LoWPAN protocol stack has an extra layer between the network layer and MAC layer which is called the adaptation layer. This adaptation layer is responsible for header compression, fragmentation, and reassembly of IPv6 packet when it is sent or received over the IEEE 802.15.4 standard [9].

The rest of the paper is organised as follows. Section II briefly reviews related studies. The overview of SDN, NFV, and cloud computing are introduced in Sections III, IV, and V, respectively. The methodology of the proposed SD–NFV approach to obtain the experimental results are detailed in Section VI. In Section VII, the obtained experimental results are discussed and validated. Finally, Section VIII concludes the paper outcomes.

II. RELATED WORK

This section introduces the concerning studies that are closely related to the integration and implementation of SDN and NFV in low–power and low–data rate wireless networks (i.e. ZigBee and 6LoWPAN) with cloud computing, as well as the current possibilities to enable these technologies to work together in the same network infrastructure.

The authors in [10] presented the deployment of OpenFlow technology in Wireless Sensor Network (WSN), the proposed approach being called the Flow–Sensor, which led to considerable achievements in IoT and cloud computing through network virtualisation. In an ideal scenario, the Flow–Sensor had reachability points more than the typical sensor. The authors concluded that better results might be achieved in large scale network.

The authors in [11] identified two problems in WSN, including the difficulty of policy changes and network management. They developed a new architecture called Software Defined–WSN (SD–WSN) with Sensor OpenFlow (SOF) to address the key technical challenges in WSN.

The authors in [12] introduced Software–Defined Wireless Network (SDWN) that benefited from the wireless infrastructureless networking environments with special emphasis on Wireless Personal Area Networks (WPAN). They analysed SDN in IEEE 802.15.4–based WPAN and discussed the SDWN requirements to adopt flexibility in flow table rules and node's duty cycle.

The authors in [13] suggested a cost–effective implementation of SDN testbed using Raspberry Pi and Open vSwitch (OVS). The testbed was validated using OpenFlow specification 1.0 and proven to maximise the network throughput compared to NetFPGA–1G.

The authors in [14] introduced Software Defined Networking –WIreless SEnsor networks (SDN–WISE) to reduce packets exchange between the nodes and SDN controller, as well as to make the nodes programmable for running different applications. The Application Programming Interface (API) of the SDN–WISE allowed the developers to build SDN controllers using the preferred programming language. The SDN–WISE prototype was implemented using real SDN controller and OMNet++ simulator.

An OpenFlow testbed was implemented in [15] using a low–priced computer board and an open source base virtual switch, the testbed being called the Pi Stack Switch. The programmable network implemented using ONOS SDN controller and OpenVirteX as network hypervisor, the implemented network infrastructure consisted of the SDN control layer and the virtualisation layer.

A structured and hierarchical management mechanism was proposed in [16] based on SDN for WSN. The proposed approach was called Software–Defined Clustered Sensor Networks (SDCSN), which the authors argued that the SDCSN approach solved the inherent problems in WSN, and they highlighted some suggestions for future research in ad–hoc networks.

A real–time Software–Defined Wireless Networks (SDWN) testbed was implemented in [17] using Raspberry Pi as

OpenFlow Switches. An OpenDayLight controller was used to analyse the network events. In addition, traffic–aware routing algorithm was implemented to manage and monitor the network flow and Quality–of–Service (QoS) requirements.

The hybrid approach proposed in [18] enabled the traditional IP network to work together with SDN–based network within the same service provider. This approach was called OSHI. OSHI system was implemented using pseudo wire and virtual switches with Mininet emulator.

A detailed overview of SDN/NFV service on top of cloud computing platform of ADRENALINE testbed was described in [19]. The authors proposed a generic architecture for SDN/NFV over multi–domain transport network. The virtual Path Computation Element (vPCE) and the deployment of virtual SDN controller (vSDN) were used as two cases on top virtualised transport networks.

To the best of our knowledge, there is limited research in the literature which implements SDN and NFV using the existing 6LoWPAN hardware. To this end, this paper focuses on implementing a customised SDN controller for cloud–based 6LoWPAN network with integrated NFV technology. This paper also attempts to validate the implementation proof–of–concept, which aims to analyse how SDN, NFV, and cloud computing can interact together in the 6LoWPAN gateway. The main contributions are:

1) Designing a cost–effective M2M sensor node based on the 6LoWPAN protocol stack using open source hardware and software platforms;
2) Building up a customised SDN controller to meet the 6LoWPAN network requirements in terms of packet size and node discovery function. The customised SDN controller integrated with the PAN coordinator;
3) Virtualising two layers of the 6LoWPAN protocol stack by migrating network and adaptation layers form node protocol stack to the centralised SDN controller, which is worked as an edge router (gateway) for the cloud–based 6LoWPAN network;
4) The M2M gateway based on the 6LoWPAN is connected to the cloud computing platform for data storage and provide global connectivity to the M2M sensor network.

The proposed approach is called customised Software Defined–Network Functioning Virtualisation (SD–NFV). The SD–NFV infrastructure provides a dynamic and scalable deployment for both M2M sensor nodes and applications in heterogeneous M2M network. The aim of the SD–NFV approach is to simplify network management through a programmability feature and to provide global connectivity for IoT architecture. The implemented testbed shows a remarkable enhancement in reducing the energy consumption of M2M sensor nodes compared to traditional network.

III. SOFTWARE–DEFINED NETWORKING

Open Networking Foundation (ONF) [20] is a user–driven non–profit organisation. which focuses on open standards development of Software–Defined Networking (SDN). SDN is an umbrella term covering different types of network architectures, where the aim has been to make the network as agile and manageable through programming. SDN can be characterised by two main features, mainly the decoupling of the data and control planes as well as adding centralised programmability in the network control plane. Fig. 2 describes the SDN versus traditional networking architecture im which the control planes are separated from the network forwarding devices and located in a centralised SDN controller [21].

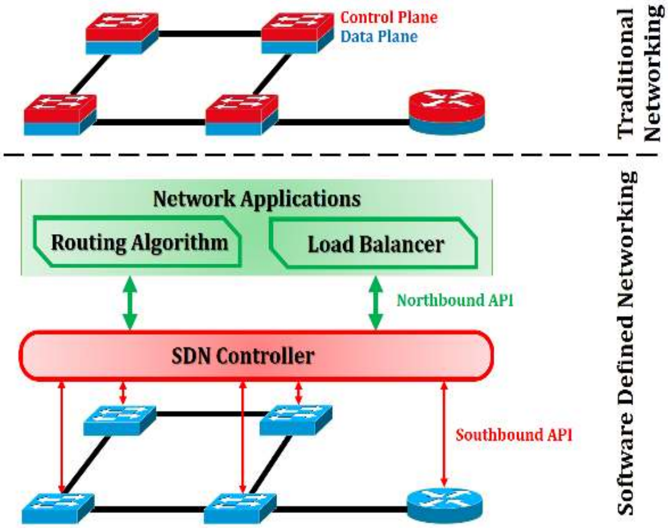


Fig. 2. SDN versus traditional networking architectures

The decoupling feature of the SDN provides greater control and management of network resources by programming the control plane. The centralised programmability brings a new innovation to optimise network configurations and improves its performance through instantaneous monitoring and applying user–defined policies [22]. A comparison between SDN and traditional networking architecture is summarised in Table I.

The decoupling of control and data planes makes the SDN architecture consists of three main components:

1) *SDN Applications* can be viewed as programmes that exchange information with the SDN controller via Application Programming Interface (API). The applications construct an abstracted network infrastructure based on the information collected by the SDN controller. The APIs can be classified into:
    - Northbound API defines the communication way between SDN controller and SDN applications;
    - Southbound API defines the way that SDN controller communicates with the physically SDN networking devices and they can be open standard or user proprietary API.
2) *SDN Controller* is the brain of the SDN networking infrastructure. It is responsible for managing the flow control in SDN networking devices via southbound API and the SDN applications via northbound API to bring intelligent to the network architecture.
3) *SDN Networking Devices* are the physical network forwarding devices that is used to route the data in the network based on the flow tables supported by the SDN controller through the southbound APIs.

TABLE I
COMPARISON BETWEEN SOFTWARE–DEFINED NETWORKING AND TRADITIONAL NETWORKING

Features	**Software–Defined Networking**	**Traditional Networking**
Methodology	Centralised protocol by separation of data and control planes	Dedicated protocol for each problem
Configuration	Automated and centralised configuration	Manual configuration
Control	Cross layers and dynamic global control	Single layer and static control
Implementation	Software–based environment and new ideas implemented in software	Hardware–based environment and limited implementation of new ideas due to hardware difficulty

The OpenFlow defines the southbound communication protocol that enables the SDN controller to communicate directly with the SDN networking devices [23]. In an OpenFlow environment, the routers and switches should support OpenFlow protocol to exchange information with the SDN controller, and it is considered as one of the first SDN communication protocol standards. The proprietary southbound APIs can be also defined by the users to customise the SDN controllers for a particular application [24] [25].

IV. NETWORK FUNCTIONING VIRTUALISATION

The Network Functioning Virtualisation (NFV) technology endeavours to virtualise the network applications or services in order to be executed on a single programmable component. NFV has drawn considerable interest from both industry and academia as important technology toward virtualisation of network applications. It reduces operating and capital expenses, whilst also enabling the deployment of different services across the network by decoupling the network functions from the physical network devices on which the functions run and new services can be deployed faster over the same physical platform [26].

The NFV increases the network infrastructure flexibility and reduces the hardware cost because it sets out to accomplish network functions in software installed on the shared server instead of running on dedicated hardware devices. Accordingly, NFV will simplify, organise and expand the network services more quickly with less installation cost.

Fig. 3 shows the basic concept of NFV where Network Function (NF) has been implemented apart from the network devices hardware. The Virtual Network Function (VNF) in NFV technique is similar to Physical Network Functions (PNF) in traditional networking. Multiple PNFs can be assembled into a single VNF or a single PNF can be divided into multiple VNF. The relationship between VNF and PNF could be one to one mapping, or one to many. These mapping relationships can be optimised to enhance network resource management [27] and consequently, NFV may be an adequate technology for future network infrastructure in terms of the following [28]:

- Network performance: NFV architecture might be able to obtain the same network performance compared to that achieved from NFs running on dedicated hardware by evaluating network deadlocks and mitigating them;
- Heterogeneity support: the big challenge to the NFV is to support network heterogeneity from proprietary hardware–based service perspectives and fragmentise the barriers to synchronise different standards;
- Dynamic resource allocation: NFV should perform different network functions at various times on the same physical hardware by reallocating the shared infrastructure resources among the hardware and software components of the network.

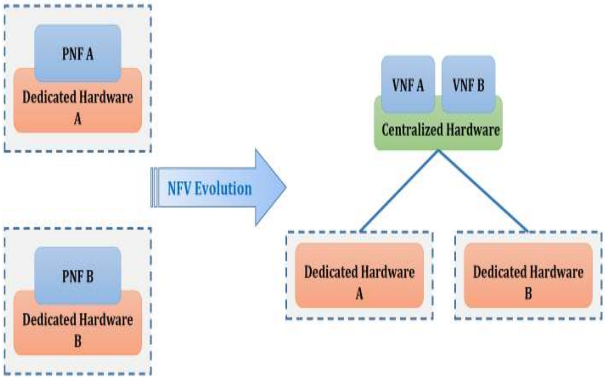
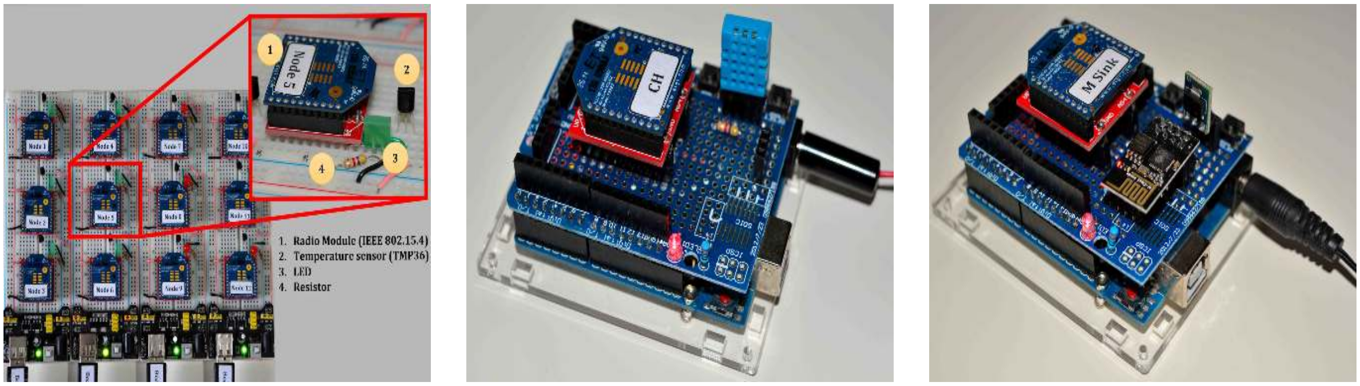


Fig. 3. Basic NFV architecture

V. CLOUD COMPUTING PLATFORM

In recent years, cloud computing and IoT are emerging technologies that led the evolution of programmable networks. Cloud computing means accessing stored data and programmes over the Internet, while the IoT simplifies the way in which large amounts of data are being collected over the interconnected M2M nodes. The cloud computing has no borders and has global communication paths. It is characterised by having on–demand service, global network and shared pool of resources. On–demand service refers to the user requesting to manage his own computing resources. A global network provides ubiquitous connectivity over the Internet to deliver different services. The shared pool of resources allows the user to fetch data from the shared resources located in the remote data centres [29].

The main advantages of cloud computing are represented in providing *scalability* to the network shared resources in terms of processing and storage, delivering *reliability* by allowing access to the cloud resources via the Internet, and it is



(a) Simple node (b) Advanvced node (c) Sink node

Fig. 4. M2M sensor node prototypes used in the cloud–based 6LoWPAN testbed

considered to be *efficient*, because it enables the deployment of new algorithms and applications for delivering new services for remote M2M sensor networks.

There are three service models for cloud computing, commonly known as: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS), which differ in terms of the control privilege that the users have over the stored information [30]:

1) Software as a Service (SaaS) is a software distribution model where the software is owned by the cloud provider and enables the remote users to access and use applications that are hosted in the cloud;
2) Platform as a Service (PaaS) is shared hardware and software platforms which enables the users to deploy their own applications with certain constrains;
3) Infrastructure as a Service (IaaS) is virtualised computing resources over the Internet where the users can manage and control the cloud applications, storage, and network connectivity without the ability to control the cloud infrastructure.

VI. PROPOSED SD–NFV APPROACH USING 6LOWPAN

This section details the implementation steps of the proposed Software Defined–Network Functioning Visualisation (SD–NFV) architecture in the cloud computing platform based on 6LoWPAN testbed. The SD-NFV approach has been proposed as an energy–efficient way to prolong the 6LoWPAN network lifetime through network programmability feature. The description of the testbed will be discussed in detail, including hardware and software components. The developed 6LoWPAN testbed has been integrated with a cloud computing platform to provide global access to the M2M sensor network.

A. 6LoWPAN Hardware Platform

The M2M sensor node consists of various sub–systems, such as sensing, computation, communication and power sub–systems. 6LoWPAN is a pioneer protocol aimed at providing small devices that have constrained processing and limited energy with the ability to have IPv6 global connectivity. Over the last few years, there have been several free and commercial solutions developed for 6LoWPAN. Most of the developed approaches were implemented based on an operating system, where the 6LoWPAN protocol stack was used along with the node's operating system. However, as the M2M sensor nodes are characterised by a small memory and moderate processing unit with limited energy source, so it is not practical to include an operating system with dedicated software applications onto those devices at the same time.

In order to develop a 6LoWPAN protocol stack with a low memory M2M sensor node and make it workable with existing IP networks, it is necessary to use the available open source resources to the maximum possible extent. Consequently, the open source hardware platform has been chosen and integrated to fit into the designed M2M sensor node scheme as well as the 6LoWPAN gateway.

One of the most important features of the M2M sensor node is the selection of the processing platform. The M2M nodes need to be cost–effective and energy–efficient to meet the IoT network promises. There are several types of nodes available in both commercial and open–source domains. The proposed approach is based on an open source hardware platform represented by the Arduino [31], which is a microcontroller board based on the ATmega328 chip, as a processing platform. The XBee module is deployed as a radio communication for MAC and PHY layers of the IEEE 802.15.4 standard, while a temperature and humidity sensor is used as a sensing unit for the M2M node. The Arduino board has been chosen due to its low energy consumption, small size, cost–effectiveness, and programmability feature. Choosing the Arduino board will open new horizons to increase network programmability and management through open source hardware platforms.

To realise the concept of the IoT paradigm, it is essential to make the things (connected objects) addressable, controllable, and accessible via the Internet. The proposed approach has been tested using a simple temperature and humidity sensing application in which the sensor nodes transmit the sensed data to the M2M gateway. Fig. 4 shows 6LoWPAN–based M2M sensor node prototypes, which are classified into:

- *Simple node*: the simple node performs sensing and communication only, without any processing capabilities and it cannot be selected as a cluster head. The simple node is composed of a temperature sensor (TMP36) attached directly to the XBee module, and an LED used as an indicator for receiving the control signal from the cluster head in hierarchical topology or from the sink node in star topology. The simple node is battery powered using 3.7 V/1000 mAh battery and is shown in Fig. 4(a).

- *Advanced node*: the advanced node performs sensing, communication, and processing of the sensed data. It can be selected as cluster head among cluster members. It comprises a temperature and humidity sensor (DHT11), XBee module, and an LED. All these components are attached to the Arduino Uno board. The advanced node is battery powered using 9 V/1600 mAh battery and is shown in Fig. 4(b).
- *Sink node*: the sink node is the final destination for all the data being sensed by the M2M sensor nodes. It could be static or mobile depending on the application. It is built–up of Arduino Uno board equipped with two communication modules (XBee and ESP8266). The XBee module is used for the M2M sensor network communication, while the ESP8266 is used for Internet communication. The ESP8266 connects the 6LoWPAN network with the IP networks as well as the cloud platform where the data is being stored. The sink node is permanently powered with extra storage capability and is shown in Fig. 4(c).

B. Customised SDN Controller Design Considerations

SDN and NFV are complementary technologies which have a lot in common, because they are both aimed at developing open software for standardised network hardware. The NFV technology is geared towards creating on–demand programmable network functions and locate them on most suitable location in the network infrastructure using adequate network resources [32]. The SDN technology can decouple the control plane and data plane in order to increase the network programmability and reconfiguration. Whilst SDN and NFV have value when implemented separately, combining them in one network will achieve greater value.

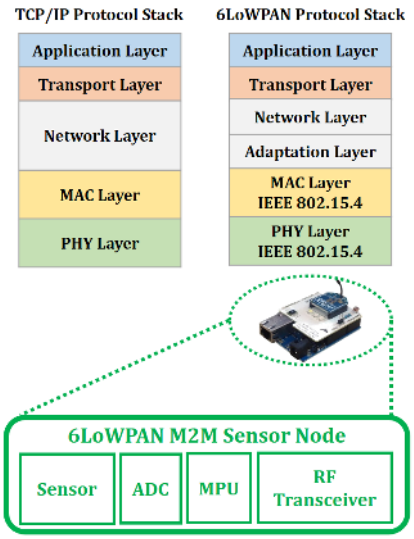


Fig. 5. M2M sensor node architecture

Current approaches to integrating M2M sensor nodes in the Internet have several drawbacks, and hence, alternative architectures need to be proposed and evaluated. The proposed integration of SDN and NFV is aimed at deploying different routing algorithms for the 6LoWPAN network by deploying the VNF on top of the integrated cloud–based 6LoWPAN gateway for homogeneous and heterogeneous M2M sensor networks. Fig. 5 shows the typical architecture of the M2M sensor node with a 6LoWPAN protocol stack alongside the TCP/IP stack. The SDN controller is a software–based network entity that is used to manage and control the network devices using programmable elements via different APIs. In order to adopt the SDN concept in the M2M sensor network, a novel customised SDN controller is proposed to fill the research gap outlined earlier. The customised SDN controller should take into account the limited memory and processing unit of the M2M sensor nodes to achieve a low software footprint.

The SDN controller is a software artifact being customised to fit a 6LoWPAN protocol stack, which enables end–to–end services on resource constrained devices. The controller is responsible for the following: (i) discovery of network topology; (ii) service management; (iii) virtualisation service; and (iv) data routing and load balancing. Additionally, the SD–NFV introduces a new flow table entry to cope with the high memory usage of the programmable interface.

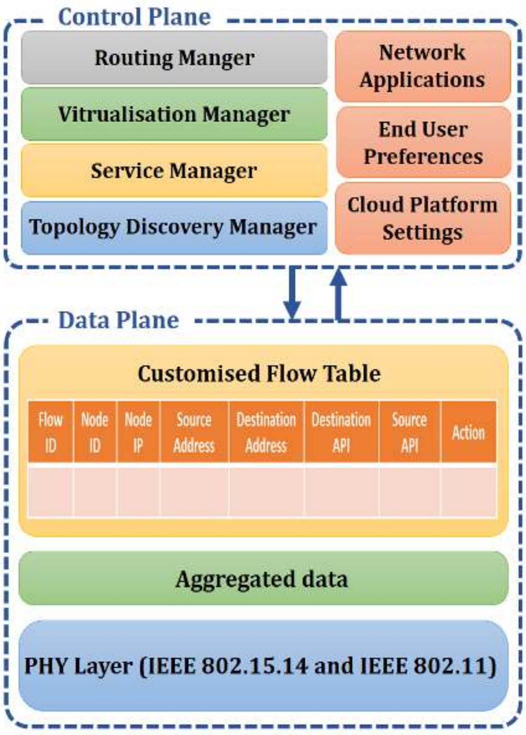


Fig. 6. Customised SDN controller architecture

Fig. 6 illustrates the architecture of the customised SDN controller used in the proposed SD–NFV approach. The controller is part of the 6LoWPAN coordinator, which is the 6LoWPAN gateway that starts up and initiates the network using unique PAN ID. The network discovery manager uses a discovery function to check the available alive nodes or

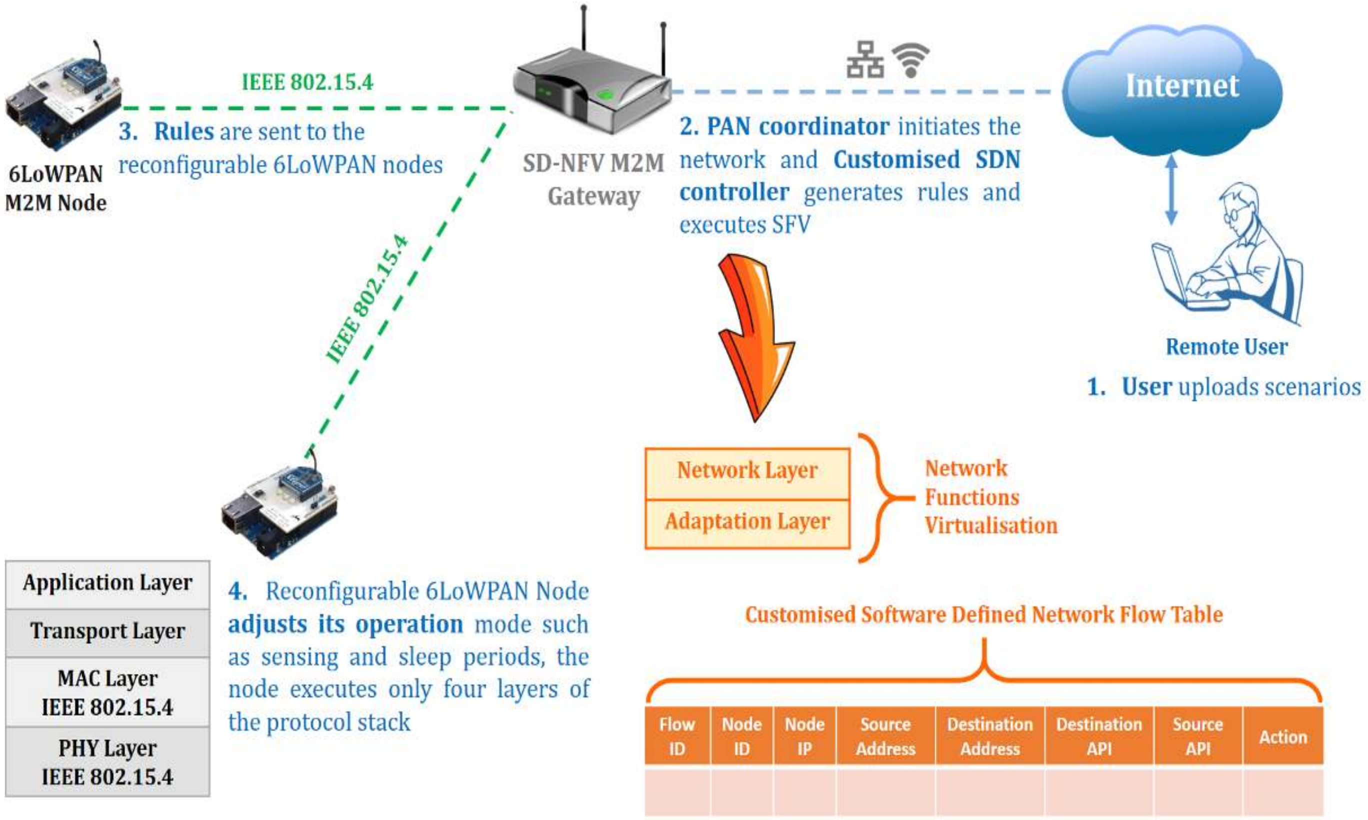


Fig. 7. The proposed architecture of the SD–NFV gateway using cloud–based 6LoWPAN testbed

newly joined nodes; this function is performed periodically to keep the global topology of the network up–to–date and to modify the alive node table entries. The service manager is important for allocating each node with a different level of services depending on the node's priority in the customised flow table entities. Also, the service manager is responsible for providing cloud service connectivity to the 6LoWPAN network. While the virtualisation manager allows different 6LoWPAN nodes to share the same network functions in the gateway as well as provide virtual individual connectivity between the 6LoWPAN nodes and cloud computing platform. Finally, the routing and load balancing manager is capable of executing different routing algorithm and performs load balancing optimisation techniques to achieve high throughput and reduce the end–to–end delay in M2M sensor network based on 6LoWPAN protocol stack.

To summarise the proposed approach, Fig. 7 illustrates the relationship between SDN, NFV, and cloud computing with the 6LoWPAN M2M sensor network. It is clear that each technology abstracts certain function from different network resources; the benefits obtained from each of them are similar in terms of traffic agility, cost–effectiveness, reduction in nodes' energy consumption, and dynamic network scalability. In addition, Fig. 7 depicts the layer's abstraction and the customised SDN flow table entries.

The 6LoWPAN protocol stack has been implemented using the Arduino pico Internet Protocol version 6 (pIPv6) stack, the library being available at [33]. The customised SDN controller was built using C++ language and deployed in the 6LoWPAN gateway, while the NFV used to migrate the network layer and adaptation layer from node's protocol stack to the gateway protocol stack and merge them with the SDN controller. This virtualisation function is called Sensor Function Virtualisation (SFV), which transforms multiple node tasks into software packages inside the 6LoWPAN gateway. Accordingly, the 6LoWPAN gateway now handles the 6LoWPAN coordinator for network initialisation, the customised SDN controller, and the two layers (network and adaptation layers) from 6LoWPAN protocol stack.

The Requests for Comments (RFC) 4861 [34] was the first IPv6 node discovery specification, which was revised in 2012 by RFC 6775 [35] so as to be able to adopt 6LoWPAN node discovery requirements. The 6LoWPAN edge router or PAN coordinator is responsible for connecting the 6LoWPAN network to the external IP networks and propagate the IPv6 prefixes among the 6LoWPAN nodes. In the traditional 6LoWPAN network, each node keeps checking its reachability to the edge router by performing a heavy control message exchange, such as Node Discovery (ND), Router Advertisement (RA), Neighbour Advertisement (NA), Neighbour Unreachability Detection (NUD), Duplicate Address Request (DAR), and Duplicate Address Confirmation (DAC). The 6LoWPAN node periodically sends NUDs until it receives a confirmation, even if it does not have data to send. The major issues in traditional node discovery are heavy packet transmission over the IEEE 802.15.4 medium, significant energy consumption to maintain the network connectivity, and reduction in link reliability. The main challenge in 6LoWPAN node discovery is to develop a mechanism that provides less packet exchange for network connectivity with minimum discovery latency and power consumption to form the global network topology.

On the other hand, the customised SDN controller has topology discovery manager that is responsible for maintaining the network connectivity for efficient data routing. The proposed topology discovery mechanism takes advantage of the virtualised layers and the 6LoWPAN network does not need any IP connectivity at the node level. Hence, this will reduce the generated packets for node discovery and minimise the node's energy consumption by preventing the periodic NUDs and relevant message exchange. The proposed approach is based on the SDN flow table entries, whereby after the network initiation phase is completed, the 6LoWPAN coordinator reports to the SDN controller the address of the alive nodes. Subsequently, the customised SDN controller assigned each node an IP and saves this entry in the alive node table. As the SDN controller knows the global topology of the network, it can build–up the flow table to each node including its IP assignment, as shown in Fig. 7. The topology discovery manager performs network discovery on a regular basis, but the table will only be updated when a node joins or leaves the 6LoWPAN network. The proposed SD–NFV approach reduces the network discovery latency as well as reducing energy consumption in 6LoWPAN nodes during the network topology discovery phase.

To our best knowledge, the leveraging of cloud, SDN, and NFV technologies in the 6LoWPAN node discovery and data routing have not been considered in previous literature. The proposed SD–NFV approach compromises the energy consumption with the end–to–end delay to prolong the network lifetime. Each technology abstracts certain functions to provide a wireless programmable network that supports heterogeneous M2M networks. The customised SDN controller with the NFV and cloud computing is aimed at providing multi–vendors compatibility for smooth protocol evaluation and implementation. In sum, the proposed approach will bridge the research gap with the SD–NFV approach that offers hardware–independent and on–demand function installation.

C. Integration of Cloud Computing Services

ThingSpeak [36] is the cloud computing platform used in the implementation of SD–NFV approach based on 6LoWPAN testbed, which provides a free storage with a data visualisation feature. The cloud platform connects the 6LoWPAN network with the global Internet through the SD–NFV gateway. There are two types of channels, namely: data channels and control channels. The data channels are used for storing the sensed data, while the control channels are used for sending control commands to a specific node over the IP network. These ThingSpeak channels are chosen to demonstrate that the SD–NFV gateway provides bidirectional communication for the 6LoWPAN nodes. The cloud computing platform provides ubiquitous connectivity between the M2M sensor nodes and the external IP networks through the SD–NFV 6LoWPAN gateway. Fig. 8 illustrates service chaining operations and network connectivity for the developed approach. The cloud front–end API is visible to the user in order to deploy different algorithms and change the network preferences, while the back–end of the network resides in the 6LoWPAN network to provide a global connection via the M2M gateway.

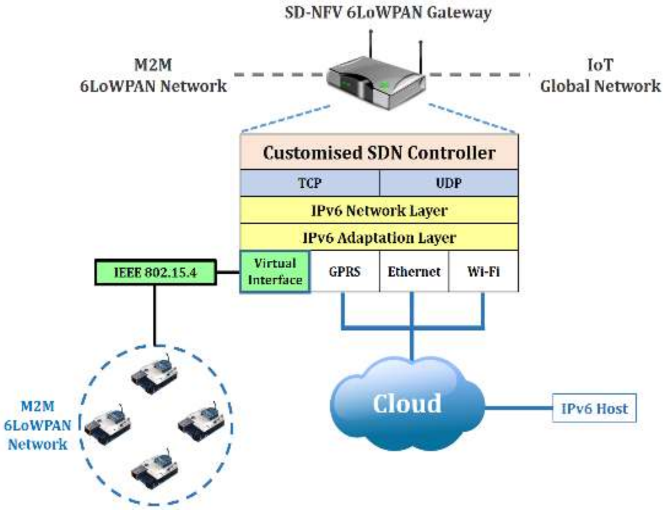
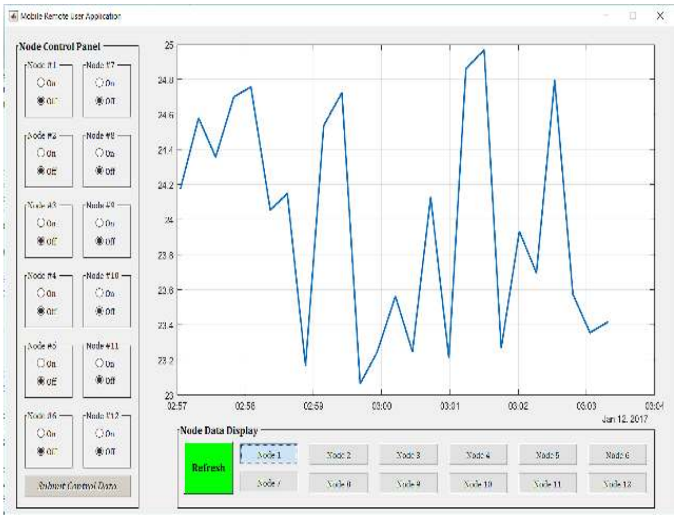


Fig. 8. 6LoWPAN cloud connectivity through SD–NFV gateway

D. Remote End–User Application

A simple end–user application is built using MATLAB software to emulate external IP access to the 6LoWPAN network through the SD–NFV gateway. The application is used to retrieve the data from the cloud and to analyse it on a remote PC. In addition, the remote application is used to send control commands to the M2M sensor nodes by turning the attached LED on/off to verify the IP connectivity and network heterogeneity. Fig. 9 shows the Graphical User Interface (GUI) of the end–user or remote application. The remote application reads the data from the data channels of the cloud platform (ThingSpeak) and sends the network preferences to the 6LoWPAN gateway using the control channels of the same cloud computing platform.



Node #1 ○ On ● Off
Node #2 ○ On ● Off
Node #3 ○ On ● Off
Node #4 ○ On ● Off
Node #5 ○ On ● Off
Node #6 ○ On ● Off
Node #7 ○ On ● Off
Node #8 ○ On ● Off
Node #9 ○ On ● Off
Node #10 ○ On ● Off
Node #11 ○ On ● Off
Node #12 ○ On ● Off

Fig. 9. GUI of the end–user application

VII. PERFORMANCE EVALUATION RESULTS

This section provides detailed proof–of–concept testbed results for the SDN and NFV integration in a cloud–based 6LoWPAN M2M network. The testbed experiments were carried out in indoor environments. The 6LoWPAN network initiates two successive steps. First, network discovery is performed before carrying out any sensing tasks to make the network topology visible to the customised SDN controller. After the nodes are discovered, the second step is started by executing the sensing application in each M2M node and reporting the sensed data to the gateway prior to their storage it in the cloud platform.

A. Node Discovery Phase

It is hard and resource intensive to discover each alive node in a 6LoWPAN network manually. Consequently, automated node discovery mechanism has been designed to monitor the states of the nodes in the network, which is delegated to the 6LoWPAN coordinator. The network coordinator is integrated with the SDN controller to work in harmony for communication cost reduction between M2M nodes. In addition, integrating PAN coordinator with the SDN controller can achieve high bandwidth utilisation compared to the traditional 6LoWPAN node discovery, where the cost associated with communication is usually more than that of sensing and processing.

Fig. 10 illustrates the node discovery time versus the number of 6LoWPAN nodes. The main objective of the node discovery function is to achieve the lowest number of transmitted packets for a node to still be connected with the customised SDN controller. The proposed topology discovery manager is aimed at making the PAN coordinator with the customised SDN controller responsible for maintaining the status information of all M2M nodes down the hierarchy and reporting the status updates to the SDN controller. According to this information, the SDN controller builds–up two tables: one that contains the connected or alive nodes, and a second that is the flow table where each node is mapped to note IP, cloud APIs, and action.

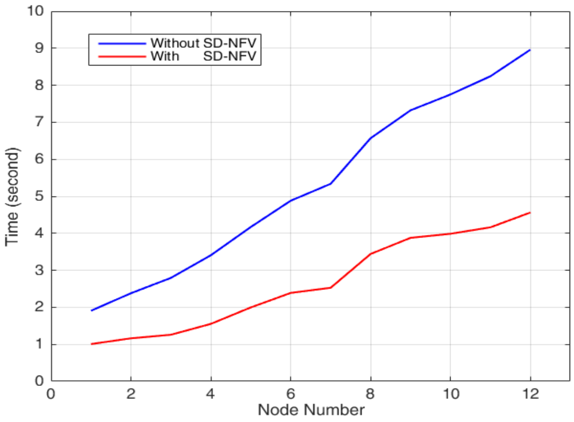


Fig. 10. Node discovery delay

From Fig. 10, it is clear that the SD–NFV enhances the network node discovery process by reducing the topology discovery time by 60% compared to the traditional 6LoWPAN network discovery time. This reduction in network discovery time can be justified due to the decoupling of the control and data planes, whereby the SDN controller will not update the tables frequently. The flow table update takes place when there is no reply from the node, and the corresponding node will be removed from the table or when a new node joins the network and it will be added to both tables.

B. Execution of Sensing Application

When the execution of the sensing application starts, the SD–NFV gateway becomes responsible for filtering the ingress traffic from both the IEEE 802.15.4 and IEEE 802.11 transceivers. It will perform packet fragmentation and packet assembly depending on the destination address of the packet.

The 6LoWPAN nodes are characterised by low–data rates, low–energy consumption, low–cost, and generation of flexible topologies. The traditional 6LoWPAN network will not be able to run different applications but rather it is able to execute a single application. Due to the limited energy source attached to the 6LoWPAN node, the nodes need to use their energies efficiently. The node lifetime is the time span from deployment to the instant when the node is considered non–functional or failed. The analysis focuses on the advanced nodes because these play the role of being a cluster head in hierarchical topology. The proposed SD–NFV approach enhanced the advanced node's lifetime by approximately 65% compared to the traditional 6LoWPAN networks without this approach, as illustrated in Fig. 11. The 6LoWPAN node joining the SD–NFV gateway will not deplete its energy more quickly, because the unnecessary IPv6 packets transmission is eliminated (i.e. IPv6 headers and fragmentation).

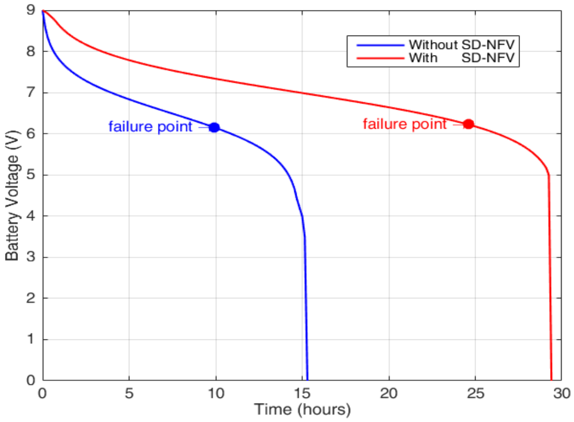


Fig. 11. 6LoWPAN node lifetime

Virtualising the network and adaptation layers of the 6LoWPAN protocol stack in the SD–NFV gateway enables the M2M node to perform low–energy sleep mode in order to conserve energy for a long period. The explanation of how the adaptation and network layers in the 6LoWPAN protocol stack work is out of the paper scope; however, detailed explanation can be found in [9] and [37]. Fig. 12 shows the current drawn from the Arduino Uno board of the advanced node for the first six seconds of network initialisation. The node joining the SD–NFV gateway will not need to have the communication antenna to be on all the time; it will turn on its communication antenna when sending and receiving the packet.

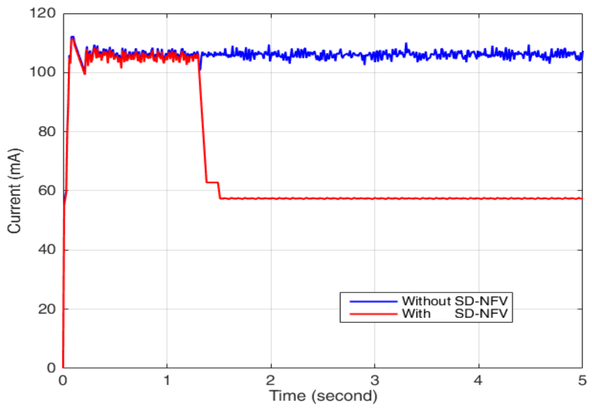


Fig. 12. First six seconds of the network initialization phase

Fig. 13 shows the current drawn from the node's battery under a periodic traffic scenario. The spikes in the red curve represent the time instances when the sensed data has been transmitted to the SD–NFV gateway.

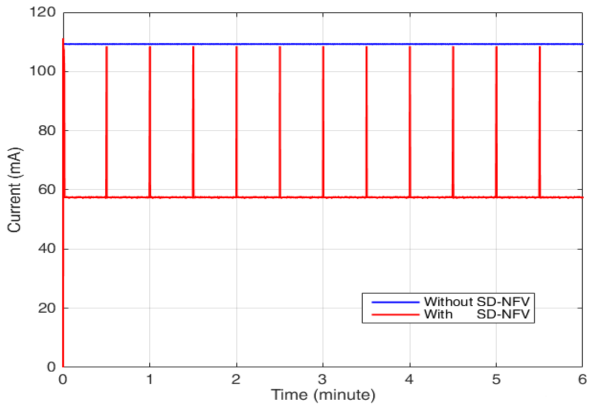
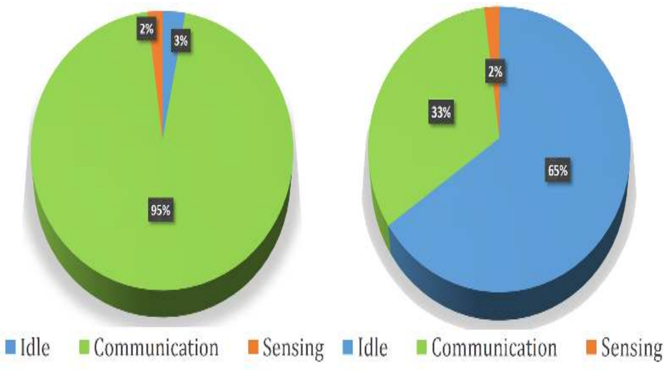


Fig. 13. Current drawn from 6LoWPAN node under periodic traffic condition

Fig. 14 illustrates the advanced 6LoWPAN sensor node activities as a percentage of the sensor node's residual energy. The experimental testbed results indicate that in a traditional 6LoWPAN network, the node spends all its energy listening to the channel and transmitting a fragmented large packet size of IPv6 (1280 byte). As a result, it depletes its energy more quickly, whilst transmitting fragmented IPv6 datagrams over the LoWPAN links efficiently. While in the proposed SD–NFV approach, IP connectivity at the node level is not necessary, because the customised SDN controller has a virtualisation manager with SFV feature, which abstracts IP connectivity from node's protocol stack to the SD–NFV gateway protocol stack. Accordingly, the advanced 6LoWPAN node only sends IEEE 802.15.4 packets (127 byte) and performs sleep mode by turning off its communication antenna. The SD–NFV gateway performs the fragmentation and assembly of IPv6 packets on behalf of the 6LoWPAN nodes. Accordingly, the node can conserve its residual energy, which is powered by batteries only and hence, prolong its lifetime. Finally, the customised SDN controller conserves nodes' energies by an indirect load balancing mechanism.



(a) Traditional 6LoWPAN node (b) SD–NFV / 6LoWPAN node

Fig. 14. 6LoWPAN node activity in relation to the node energy

The Arduino board draws significantly high current compared to other existing microcontroller boards, and hence it works for a day not for months or even a year. However, Arduino boards have been chosen in order to investigate the effects of the SD–NFV approach in very short running time.

VIII. CONCLUSION

This paper developed a proof–of–concept testbed for the SDN and NFV approaches in cloud–based 6LoWPAN gateway. Also, the implemented testbed can be viewed as a first attempt to analyse the challenges of integrating SDN and NFV together in the IEEE 802.15.4–based network, which is characterised by low–power and low–data rate sensor nodes. Currently, the implemented 6LoWPAN testbed has been tested for a specific solution which is called SD–NFV.

The implemented architecture achieves a good performance in terms of node discovery function in the gateway for global topology construction. The node discovery time has been reduced by 60% compared to the traditional 6LoWPAN node discovery time. Also, the proposed SD–NFV approach is aimed at abstracting the most energy harvesting layers from the 6LoWPAN node and makes them virtualised among all the other nodes in the network through the SDN controller. The virtualisation approach enhances the network lifetime; the node joined by the SD–NFV gateway is able to enhance its lifetime by 65% in comparison to the existing 6LoWPAN node joined by traditional 6LoWPAN gateway or edge router.

The SDN offers a new way to design, deploy, and manage IoT devices by improving the interaction between the customised SDN controller and the network infrastructure. While the NFV reshapes the current network services and makes the IoT service function chaining more agile. The proposed SD–NFV approach is quite suitable for constrained networks where energy and processing efficiency is the major concern. Furthermore, the SD–NFV gateway can handle bidirectional communication between 6LoWPAN nodes and a remote user.

ACKNOWLEDGMENT

The corresponding authors are grateful to the Iraqi Ministry of Higher Education and Scientific Research for financially supporting the current research.

REFERENCES

[1] R. van der Meulen, "Analysts to Explore the Value and Impact of IoT on Business at Gartner Symposium / ITxpo 2015, November 8-12 in Barcelona, Spain," http://www.gartner.com, 2015, [Accessed on January 2017].

[2] B. C. Villaverde, R. D. P. Alberola, A. J. Jara, S. Fedor, S. K. Das, and D. Pesch, "Service Discovery Protocols for Constrained Machine-to-Machine Communications," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 41–60, First 2014.

[3] P. K. Verma, R. Verma, A. Prakash, A. Agrawal, K. Naik, R. Tripathi, M. Alsabaan, T. Khalifa, T. Abdelkader, and A. Abogharaf, "Machine-to-Machine (M2M) Communications: A Survey," *Journal of Network and Computer Applications*, vol. 66, pp. 83 – 105, 2016.

[4] 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.

[5] 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.

[6] I. Park, D. Kim, and D. Har, "MAC Achieving Low Latency and Energy Efficiency in Hierarchical M2M Networks With Clustered Nodes," *IEEE Sensors Journal*, vol. 15, no. 3, pp. 1657–1661, March 2015.

[7] Y. J. Chen, Y. H. Shen, and L. C. Wang, "Traffic-Aware Load Balancing for M2M Networks Using SDN," in *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*, Dec 2014, pp. 668–671.

[8] 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.

[9] 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 January 2017].

[10] 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.

[11] 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.

[12] 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.

[13] H. Kim, J. Kim, and Y. B. Ko, "Developing a Cost-Effective OpenFlow Testbed for Small-Scale Software Defined Networking," in *Advanced Communication Technology (ICACT), 2014 16th International Conference on*, Feb 2014, pp. 758–761.

[14] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SDN-WISE: Design, Prototyping and Experimentation of a Stateful SDN Solution for WIreless SEnsor Networks," in *Computer Communications (INFOCOM), 2015 IEEE Conference on*, April 2015, pp. 513–521.

[15] S. Han and S. Lee, "Implementing SDN and Network-Hypervisor Based Programmable Network Using Pi Stack Switch," in *Information and Communication Technology Convergence (ICTC), 2015 International Conference on*, Oct 2015, pp. 579–581.

[16] F. Olivier, G. Carlos, and N. Florent, "SDN Based Architecture for Clustered WSN," in *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2015 9th International Conference on*, July 2015, pp. 342–347.

[17] M. Ariman, G. Seçinti, M. Erel, and B. Canberk, "Software Defined Wireless Network Testbed Using Raspberry Pi of Switches with Routing Add-on," in *Network Function Virtualization and Software Defined Network (NFV-SDN), 2015 IEEE Conference on*, Nov 2015, pp. 20–21.

[18] S. Salsano, P. L. Ventre, F. Lombardo, G. Siracusano, M. Gerola, E. Salvadori, M. Santuari, M. Campanella, and L. Prete, "Hybrid IP/SDN Networking: Open Implementation and Experiment Management Tools," *IEEE Transactions on Network and Service Management*, vol. 13, no. 1, pp. 138–153, March 2016.

[19] R. Vilalta, A. Mayoral, R. Muñoz, R. Casellas, and R. Martínez, "The SDN/NFV Cloud Computing Platform and Transport Network of the ADRENALINE Testbed," in *Network Softwarization (NetSoft), 2015 1st IEEE Conference on*, April 2015, pp. 1–5.

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

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

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

[23] A. Sabbeh, Y. Al-Dunainawi, H. S. Al-Raweshidy, and M. F. Abbod, "Performance Prediction of Software Defined Network Using an Artificial Neural Network," in *2016 SAI Computing Conference (SAI)*, July 2016, pp. 80–84.

[24] P. A. Morreale and J. M. Anderson, *Software Defined Networking: Design and Deployment*. Boca Raton, FL, USA: CRC Press, Inc., 2014.

[25] D. F. Macedo, D. Guedes, L. F. M. Vieira, M. A. M. Vieira, and M. Nogueira, "Programmable Networks - From Software-Defined Radio to Software-Defined Networking," *IEEE Communications Surveys Tutorials*, vol. 17, no. 2, pp. 1102–1125, Secondquarter 2015.

[26] 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.

[27] 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.

[28] 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.

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

[30] K. Hwang, J. Dongarra, and G. C. Fox, *Distributed and Cloud Computing: From Parallel Processing to the Internet of Things*, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011.

[31] Arduino, https://www.arduino.cc/, [Accessed on January 2017].

[32] J. Batalle, J. F. Riera, E. Escalona, and J. A. Garcia-Espin, "On the Implementation of NFV over an OpenFlow Infrastructure: Routing Function Virtualization," in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*, Nov 2013, pp. 1–6.

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

[34] IETF, "Neighbor Discovery for IP version 6 (IPv6)," https://tools.ietf.org/html/rfc4861, 2007, [Accessed on January 2017].

[35] IETF, " Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)," https://tools.ietf.org/html/rfc6775, 2012, [Accessed on January 2017].

[36] Internet-of-Things Platform, https://www.thingspeak.com/, [Accessed on January 2017].

[37] J. W. Hui, "An Extended Internet Architecture for Low-Power Wireless Networks - Design and Implementation," Ph.D. dissertation, EECS Department, University of California, Berkeley, Sep 2008. [Online]. Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-116.html

Bilal R. Al-Kaseem received the BSc degree and the MSc degree in Computer Engineering from Nahrain University, Baghdad, Iraq, in 2006 and 2010 respectively. He is currently pursuing the PhD degree in wireless network engineering at Brunel University London, United Kingdom. His research interests include design and implementation of smart sensors, WSN, SDN, NFV, M2M communication, cloud computing, and intelligent IoT applications.

Hamed S. Al-Raweshidy awarded BEng and MSc from University of Technology, Baghdad in 1977 and 1980 respectively. He completed his Post Graduate Diploma from Glasgow University, Glasgow, UK in 1987. He awarded his PhD in 1991 from Strathclyde University in Glasgow, UK. He has worked with Space and Astronomy Research Centre (Iraq), PerkinElmer (USA), British Telecom (UK), Oxford University, Manchester Metropolitan University, Kent University and currently he is the Director of the Wireless Network Communications Centre (WNCC), Brunel University London, United Kingdom.