Computer Communications 184 (2022) 160–173

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

SDN-based gateway architecture for electromagnetic nano-networks

Akram Galal [a,*], Xavier Hesselbach [a], Wouter Tavernier [b], Didier Colle [b]

[a] Department of Network Engineering, Universitat Politècnica de Catalunya (UPC) BARCELONATECH, 08034 Barcelona, Spain
[b] Internet Technology and Data Science Lab (IDLab), Department of Information Technology (INTEC), Ghent University - imec, Belgium

ARTICLE INFO

Keywords:
Nano-network
Software defined networking
Internet of nano-things
Micro/nano-gateway

ABSTRACT

Electromagnetic nano-communication has increasing attention in recent years. Several developments have been achieved in the fabrication, communication and management of various nano-network devices serving potential applications ranging from software-defined metamaterials, wireless robotic materials and body-centric communication. Such applications need uplink and downlink communication between the deployed nano-network and the external macro-world or the Internet through nano-interfaces. As a result, heterogeneous nano-network devices and their interoperability in different Internet of nano-things applications become new challenges for nano-network communication. In this regard, dynamic, flexible and distributed micro/nano-gateways can accommodate such sustainable issues and make the nano-network fully operational, regardless of the adopted application domain or the protocols used in communication. Network functions virtualization and software-defined networking technologies altogether can overcome these challenges. This article proposes SDN-based architecture and software module for the micro/nano-gateway. The proposed software module converts data formats and protocols between nano-network and traditional network domains allowing the nano-devices to be linked to the Internet. A prototype of the module is built, and the performance of the proposed algorithm is evaluated based on two communication scenarios; single tenant and multitenant. The result shows the effect of the total number of connected nano-devices and the number of packets sent by each device on the total average round-trip processing delay and the overall throughput of the micro/nano-gateway.

1. Introduction

Network virtualization is the logical abstraction of the network hardware appliance that converts its functionality into software implementation. This abstraction decouples the control from hardware and makes it more manageable to modify, maintain and upgrade. Software-Defined Networking (SDN) and Network Function Virtualization (NFV) are the main pillars to provide virtualization in current telecommunication networks. SDN decouples the control plane from the forwarding plane and makes it directly programmable and centralized, while NFV is the mechanism of abstracting functions from dedicated hardware to a virtualized environment. Both technologies are complementary, as they are used together to improve and facilitate the functionality of different communication systems [1]. One of the most promising communication systems that attract attention in recent years is nano-network, which is a communication network between very tiny devices, whose dimensions are measured in micro/nano-meters [2]. This communication enables different advanced applications of nano-technology in many fields such as biomedical, environmental and industrial fields. Among these various promising applications that are enabled by nano-networks are software-defined metamaterials, wireless robotic materials, body-centric communication and wireless networks-on-chip. Each application domain needs certain requirements and features to operate efficiently starting from different protocols in the network layer, data link layer and physical layer up to network size, nano-nodes density, latency, throughput, nodes mobility and addressing schemes [3].

SDN and NFV technologies can be used together to manage and control nano-networks, which have very limited resources and computing capabilities. A hybrid SDN/NFV architecture for the Internet of Nano-Things (IoNT) is used to provide reliable communication for nano-networks, where nano-machines are connected with SDN/NFV system using micro/nano-gateways [4]. These gateways are smart hybrid devices that can communicate in both nano-scale and classical communication paradigms in micro/macro-communication networks, hence linking the legacy communication networks with the IoT and IoNT network domains [5]. The compatibility problem of different protocols between different domains is considered an important concern to create a fully functional nano-network. Moreover, IoNT devices are usually heterogeneous, i.e., each device has its own communication protocols, data formats and applications. As a result, the choice of micro/nano-gateway depends on the application domain, the suitable protocol stack and the required task needed from the nano-network

* Corresponding author.
E-mail addresses: akram.galal@upc.edu (A. Galal), xavier.hesselbach@upc.edu (X. Hesselbach), wouter.tavernier@UGent.be (W. Tavernier), Didier.Colle@UGent.be (D. Colle).

https://doi.org/10.1016/j.comcom.2021.12.017
Received 14 March 2021; Received in revised form 6 August 2021; Accepted 21 December 2021
Available online 27 December 2021

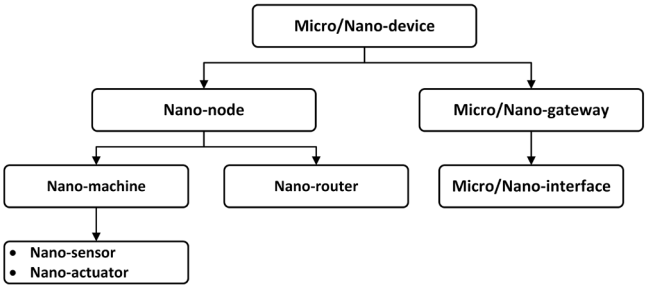

Fig. 1. Schematic classification of nano-devices [8].

application [6]. Hence, to address the heterogeneity of these devices, micro/nano-gateways are required to solve this problem efficiently.

The main contribution of this article is to propose a micro/nano-gateway architecture and software module based on SDN technology to provide an access to the nano-network from the Internet by affording a protocol conversion mechanism to accommodate different data formats coming from various nano-nodes, therefore developing different external applications. The proposed module allocates main information from each packet received from nano-domain protocols and converts it to a suitable data format to enable linking the nano-network with an external high-performance computing system. The control protocol between the SDN controller and the proposed module is based on the Programming Protocol-independent Packet Processors (P4) language, as it is not tied to a specific protocol, and it provides its own control interface rather than depending on OpenFlow control-based interface. The proposed architecture is implemented as a proof of concept, and its performance is evaluated using some evaluation metrics following single-tenant and multitenant communication scenarios.

The rest of the paper is organized as follows. The literature review and related work are presented in Section 2. The proposed micro/nano-gateway module is presented in Section 3. The implementation of the proposed module is illustrated in Section 4. The performance evaluation is illustrated in Section 5. The qualitative analysis is provided in Section 6, followed by the conclusion and future work in Section 7.

2. Literature review and related work

In this section, we briefly discuss the recent relevant work in the literature for some approaches that are related to the scope of the paper such as electromagnetic nano-network domain, nano-network simulators, IoT and IoNT gateways. Also, we discuss how the next-generation network technologies can provide an added value to the nano-network context. Moreover, we provide the research methodology used to develop our proposal.

2.1. Electromagnetic nano-network domain

Despite the evolution of the nano-technological components, nano-devices still do not have sufficient computing capabilities that allow them to work effectively [7]. Several contributions in the literature discuss electromagnetic nano-communication, which uses the properties of EM waves like amplitude and phase as information carriers. In general, the nano-network architecture consists of nano-machines, nano-routers and nano-interfaces. Nano-machines are small nano-sensors/actuators have limited energy, computational and storage capabilities. Nano-routers have relatively powerful computational resources compared with nano-machines. Nano-routers are responsible for data aggregation to/from a group of nano-machines and sending it to an associate micro/nano-gateway, which acts as a nano-interface between the nano-domain and macro-scale world [8]. Fig. 1 illustrates a schematic classification of the corresponding nano-devices.

The heterogeneity of connected nano-sensors and their interoperability produce data in different formats in various domains, which use different data models and communication protocols [9]. As a result, the micro/nano-gateways appear on the scene to provide multiple functionalities such as the interoperability between different communication protocols/domains of heterogeneous networks, encapsulation/decapsulation of packets, translation of addresses, providing traffic prioritization and forwarding packets to allow the communication with things and devices within different technologies [7].

The current state-of-the-art of THz nano-communication proposes different contributions for the protocol stack of nano-network, which consists of three layers: network, link, and physical layers combined with a high-level application layer that runs multiple application domains. Each application domain that operates in the THz band has specific constraints from the underlying nano-network such as node density, delivery latency, throughput, reliability of communication, addressing and routing protocols [3].

The network layer is responsible for data communication between nano-nodes. In order to allow such communication, nano-nodes depend on intermediate hops to forward information because of the particular nature of nano-nodes that lacks computing capabilities and energy storage. There are multiple routing protocols/algorithms are proposed in the literature to secure suitable routing strategies for the nano-network [3]. These routing protocols are classified based on two main principles: network structure and protocol operation. Based on network structure, routing protocols are classified into flat-based, hierarchical-based and location-based routing protocols. Based on the protocol operation, routing protocols are classified into path-based, query-based and negotiation-based [10].

The link layer is responsible for enabling direct communication between a pair of nano-machines or between a nano-machine and a more powerful device such as a nano-router or a macro/nano-interface. The main function of the link layer protocols is channel access, which is performed by the classical Media Access Control (MAC) protocol. Due to the peculiarities of the THz propagation and constraints of nano-machines, traditional MAC protocol cannot be directly adopted and applied in the link layer of nano-network. Accordingly, the research community provided several link layer protocols that are suitable to work for THz band nano-communication. These link layer protocols are grouped into hierarchical protocols that assume the availability of more powerful nano-controllers and distributed protocols, in which all nano-nodes have the same functionality and capability [3].

The physical layer describes the methods of transmitting raw bits over a wireless communication link between two nano-nodes. In particular, the physical layer is responsible for modulation, coding and error control, besides other methods that determine the data rate and error rate. Different protocols for the physical layer in EM nano-networks have been discussed in the literature, besides the channel characterization and modeling, which differ for each application domain [3]. For presentation clarity and consistency, Table 1 provides some examples of the aforementioned communication protocols for each layer in the nano-network protocol stack.

2.2. Electromagnetic nano-network simulators

There are several tools for simulating nano-networks such as Vouivre [31], BitSimulator [32] and TeraSim [33]. The most popular simulator is NanoSim [19,20], which is an event-based NS-3 simulator for modeling electromagnetic nano-network communications. In this simulator, the nano-network consists of nano-nodes, nano-routers and nano-interfaces. While the communication architecture consists of a network layer, a link layer and a physical layer composed of a message processing unit, which is responsible for generating and processing messages. So, this simulator represents the real network architecture and the protocol stack of EM nano-networks.

The link layer in the NanoSim simulator is composed of a simple MAC protocol, and the network layer is composed of a routing

Table 1
Summary of nano-network protocol stack.

Layer	Application/Protocol
Application	Software-defined metamaterials, Wireless robotic materials, Body-centric communication and WNoCs [3].
Network	E3A [5], CORONA [11], SLR [12], DEROUS [13], MHTD [14] and ECR [15].
Link	PHLAME [16], DRIH-MAC [17], TCN [18], Smart-MAC [19] [20], EEWNSN-MAC [21], BRS-MAC [22] and Dynamic MAC [23].
Physical	TS-OOK [24], PHLAME [16], SRH-TSOOK [25], ASRH-TSOOK [26], Multi-band OOK [27], DAMC [28], MEC [29] and SBN [30].

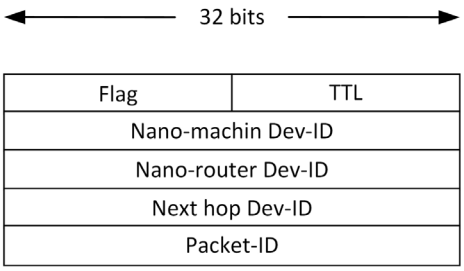

Fig. 2. Nano-network message header format [19].

module based on the selective flooding strategy. Once a new message is created, the message process unit sends it through the protocol stack to the physical interface. Since nano-machines communicate at the physical layer with very limited transmission ranges, it should be essential to establish a multi-hop path between the sender and the receiver. Hence, the routing algorithm should efficiently handle multi-hop connections. To limit the complexity of routing operations, the nano-network shall be organized following a hierarchical architecture starting from nano-machines, each one having its reference nano-router, which forwards nano-machines measurements/data towards the nano-interface, i.e., micro/nano-gateway [19,20].

In the NanoSim simulator, any message generated from the nano-network domain starts with a fixed header structure. This structure has five fields, and it is independent of the routing strategy used in the nano-network. The flag field has been added to indicate if the packet is forwarded to the micro/nano-gateway or a nano-router, the source Dev-ID, the destination Dev-ID, the packet-ID, and the Time To Live (TTL). Source and destination Dev-IDs are set to the Dev-ID of the local nano-machine and the receiver, respectively. Dev-ID is a unique identifier, and it is assigned to nano-devices, e.g., nano-gateway, nano-router and nano-machine. The structure of the header added by the network layer, as well as the value and the length of each field, can be modified depending on the use case. The packet ID is assigned by the message process unit in a sequential manner, whereas the TTL is set to a value between (100–1000), and it is decreased by one after each hop. The high value of TTL is considered under the assumption of a message in a wireless nano-network can reach its destination after traversing hundreds of nano-devices [19,20]. Fig. 2 illustrates the nano-network header packet blueprint that has been discussed in the NanoSim simulator.

Table 2 summarizes a comparison between electromagnetic nano-network simulators [3]. At this point, the scalability of the discussed simulators in the literature is not defined. Also, their reliability and usability for different communication scenarios are still open research areas. So, more cooperation between research communities is encouraged in this domain to enhance the experimental results obtained from these simulators [3].

2.3. IoT and IoNT gateways

There are some similarities in the challenges that face the gateway design of both IoT and IoNT, such as devices interoperability, addressing the heterogeneity of different protocols, the heterogeneity of connected things and the scalability of their networks [34]. The interoperability of heterogeneous devices produces data in different formats and patterns to realize and support services in various domains, which use different data models and communication protocols. As a result, interoperability shall be supported in the network for devices, data, and communication protocols [9].

In the massive IoT deployment, the heterogeneity of IoT devices entails different communication protocols and data formats, which result in a lack of interoperability, undesirable packet delays and network congestion [35]. Most IoT services are provided by different platforms. Each platform has a certain application domain, and it is associated with several sensors, gateways and computing platforms, which are built by different service providers. These devices and their communication infrastructure cause a lack of interoperability and several limitations when they are connected to the Internet because service providers need to fabricate the devices and configure the communication protocols for each application and manage redundant data generated from hundreds of sensors [36,37].

Several works have been presented in the literature discussing IoT gateway architecture [7]. These gateways have to ensure interoperability between different communication protocols and communication domains between heterogeneous networks. Encapsulation/decapsulation

Table 2
Comparison between electromagnetic nano-network simulators [3].

Simulator	Criteria								
	Development	Protocol stack (Layers implementation)			Hierarchical architecture			Mobility support	Energy harvesting capability
		Physical layer	Link layer	Network layer	Nano-nodes	Nano-routers	Nano-interfaces		
Vouivre	It is C++ THz nano-wireless simulation library, and it is developed as an extension for Dynamic Physical Rendering Simulator (DPRSim).	Yes	No	No	Yes	No	No	No	No
BitSimulator	It is developed in C++, and it utilizes a discrete event model.	Yes	No	Yes	Yes	No	No	Yes	No
NanoSim	An event-based NS-3 module.	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
TeraSim	A newer alternative to the NanoSim simulator, and it is developed on top of NS-3.	Yes	Yes	No	Yes	No	No	Yes	Yes

of packets, translation of addresses, enabling data storage, providing traffic prioritization and forwarding packets are the main functions of these gateways to allow the communication with things and devices that belong to different technologies [7].

Authors in [36] proposed an IoT architecture that enables sharing various resources including devices, data and software between many applications at different levels. Their architecture supports data interoperability in the network efficiently.

Authors in [38] studied the complex situation of different devices and multiple transmission protocols in the application of industrial IoT. They proposed an architectural platform for a unified gateway of industrial IoT.

Authors in [35] identified the main problems that arise in the management of thousands of IoT devices of different technologies and vendors. Also, they proposed an open-source software solution architecture to orchestrate infrastructure enabling interoperability and management of industrial IoT devices from multiple vendors.

SDN is used in IoT applications to provide programmability of the IoT network without changing the architecture of the existing implementation [39]. In the earlier stages of SDN-enabled IoT platforms, a centralized controller was used to manage the whole network. Later, multiple controllers have been distributed to minimize the packet loss. In the meantime, a combination of Blockchain, SDN and IoT technologies is being studied to ensure privacy and security management [40].

Authors in [6] proposed an architecture for a distributed gateway suitable for an IoT application called disaster management. Their proposal used NFV and SDN technologies and the functions of the gateway were provided as VNFs taking into consideration the mobile scenario of IoT sensor network in the disaster management application.

Authors in [7] proposed architecture for IoT using SDN, NFV and fog computing technologies. The gateway in their architecture ensures interoperability between different communication protocols and heterogeneous networks, also it supports a standard northbound interface such as Representation State Transfer (REST) with the feature of dynamic updating of the gateway, however this gateway does not support distributed architecture.

Authors in [9] investigated SDN technology when designing IoT gateways and proposed an IoT architecture that supports standard northbound interfaces using JavaScript Object Notation (JSON), however their proposed gateway does not support a distributed manner, also it does not enable the same gateway to be used by more than one application.

Authors in [34] proposed NFV/SDN-IoT architecture aiming at addressing the scalability and mobility of IoT networks. They proposed SDN-based gateways and implemented the functionalities of the gateways as VNFs supporting heterogeneous IoT devices. The gateway in their proposal can be updated dynamically, however their proposed architecture cannot be deployed over ad-hoc networks, and it does not support distributed architecture.

It has been noticed that none of the reviewed papers suggested adopting the gateway architecture for the IoNT platform, which has different devices pattern fabricated to operate in the THz band, besides their different communication protocols, packet formats and computing capabilities [41]. Hence, IoNT gateways are not an extension of traditional IoT gateways at the nano-scale. They have the following differences:

- The gateway in IoT operates with devices in the GHz band, while the gateways in the IoNT operate with devices in the THz band.
- The gateway of IoT operates in system architecture composed of application layer, management layer, network construction layer and perception layer. While the gateways of IoNT operate in a system architecture composed of an application layer, network layer, link layer and physical layer.

Table 3
Main differences between IoT and IoNT gateways.

Category	IoT gateway	IoNT gateway
Communication range	GHz band	THz band
System architecture	Application layer Management layer Network construction layer Perception layer	Application layer Network layer Link layer Physical layer
Nodes classification	Sensors/Actuator	Nano-machines Nano-routers
Computing capability	Moderate	Limited
Energy conservation	High	Low

- The nodes in IoNT fabricated in nano-scale, and IoNT gateways communicate with nano-nodes that are classified to nano-routers and nano-machines, i.e., nano-sensors/actuators. While IoT gateways collect a stream of data coming to/from devices fabricated on the macro-scale such as sensors and actuators.
- The computing capabilities of IoNT gateways are confined considering that they have limited memory, storage and processing. Accordingly, they are not capable of executing complex computations. While the IoT gateways have relevant powerful computing capabilities.
- When IoNT gateways process information coming to/from different nano-machines, considering thousands or millions of them, the harvested energy of the gateway will be rapidly consumed. While the IoT gateways have a higher conservation possibility.

As a result, IoT gateways are not suitable to be used in nano-network communication. Therefore, specific interfaces such as micro/nano-gateways need to be evolved, and this is the main motivation behind our proposed work. We summarize the main differences between IoNT and IoT gateways in Table 3 [41].

2.4. Next generation network technologies with nano-network

SDN coupled with NFV making a critical enabling technology to change the way the network operates. An NFV/SDN system becomes the enablers of newly added value services on top of the existing infrastructure of IoT and IoNT, which have a tremendous impact on modern life [34]. NFV can serve SDN by virtualizing the SDN controller function to be run on the cloud allowing dynamic migration of the controllers to external locations out of the hardware appliance. While SDN can serve NFV by providing programmable network connectivity between different VNFs to achieve an optimized traffic engineering [34].

SDN technology facilitates building network services. The logically centralized controller and the standardized southbound interfaces provide a basic and powerful architecture to support common functions and interfaces that can be developed rapidly. Moreover, interoperability of data in different formats and models can also be achieved easily in a dynamically controlled way. Accordingly, applications in different domains can be supported quickly and efficiently by the same system. It is worth noting that the centralization of the control may cause scalability limitations. Therefore, a distributed controller scheme with east/west interfaces will overcome this issue [9]. OpenFlow is the most well-known SDN protocol for the southbound interface. It is a standard communications protocol defined between the control layer and the infrastructure layer of the SDN architecture. It allows direct access to the forwarding plane of network devices such as OpenFlow switches and routers [42].

P4 is a high-level language for programming packet processors. It raises the level of abstraction for programming the network and serves as a general interface between the SDN controller and the network devices. P4 is a protocol-independent that works in conjunction with SDN controllers. The objective of designing P4 is the continuous increase of

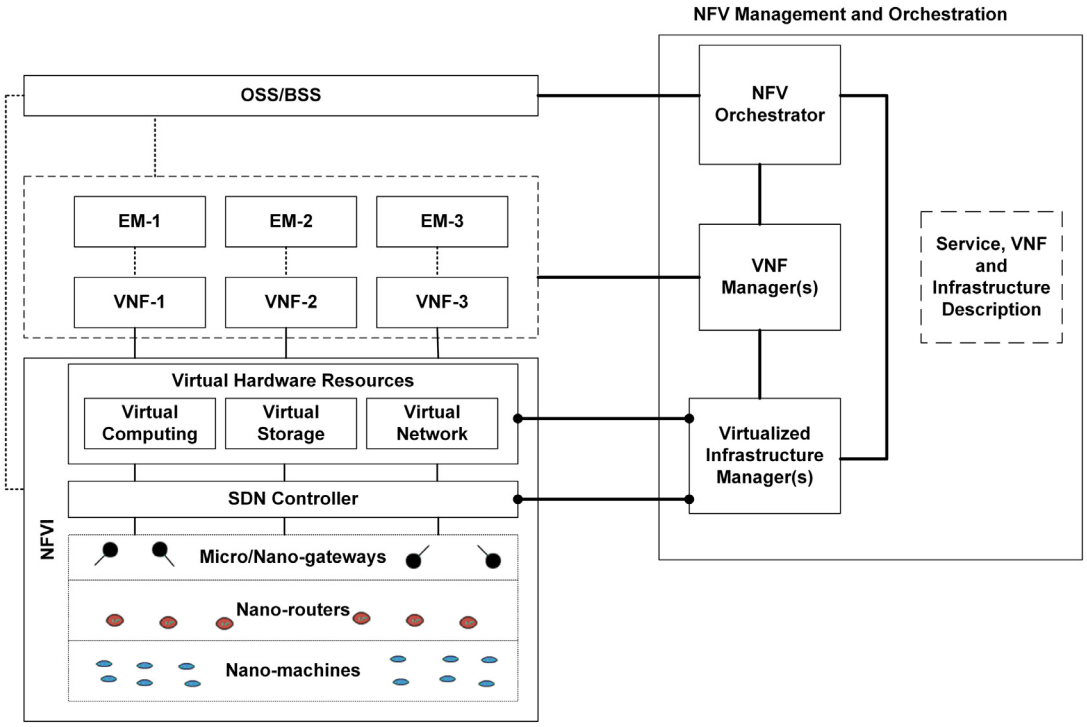

Fig. 3. General architecture combining NFV and SDN with IoNT.

new header fields. For example, the original version of the OpenFlow protocol has several updates to support more than 40 protocols due to the packet encapsulation that requires different processing by the network devices. For instance, data center network operators increasingly want to apply new forms of packet encapsulation to deploying software switches that are easier to extend with new functionalities. Rather than frequently extending the supported fields in the OpenFlow protocol, an open high-level flexible interface is developed to define different mechanisms for parsing packets and matching header fields [43].

Designing P4 has three main goals: (i) Reconfigurability in the field: the controller should be able to redefine the packet parsing and processing in the field. (ii) Protocol independence: switches should not be tied to any specific network protocols or packet formats. Instead, the controller should be able to specify a packet parser for extracting header fields with particular names and types and a collection of match/action tables that process these headers. (iii) Target independence: the controller should not need to know the details of the underlying switch. Instead, a compiler should make the switch's capabilities into account when turning a target-independent description into a target-dependent program [43].

SDN and NFV can serve nano-network properly to overcome its limitations, based on the fact that nano-machines suffer from limited computing resources and capabilities due to their nano-scale size. Authors in [4] proposed hybrid computational system architecture for nano-network communication with a layered approach combining SDN and NFV technologies. In their architecture, the SDN layer is responsible for the intelligent delivery of data between devices, while the NFV layer is responsible for the global computations and decision-making process. Their proposed architecture has provided the possibility of managing and controlling a group of nano-nodes adding an ample level of flexibility and reconfigurability for the nano-networks. While authors in [8] proposed a routing protocol for electromagnetic nano-network based on SDN/NFV architecture, where the routing intelligence, the routing decisions and the complex computations are decoupled from the nano-routers to be fully compiled externally on the computational SDN/NFV architecture. However, both previous works in the literature [4,8] did not provide a detailed illustration about the architecture design of the micro/nano-gateway that allows a group of nano-machines/sensors to interact and communicate with each other or with the Internet. Fig. 3 shows a general architecture of SDN and NFV communicates with a group of nano-sensors/machines and nano-routers, which are part of a larger IoNT network [4].

2.5. Summary

NFV and SDN technologies play a vital role in the nano-network communication architecture. Since TCP/IP protocol suite cannot be directly adopted into EM nano-network, because of their constraints in computing and resources, a simple prototype protocol stack is considered for EM nano-network as a starting point for future optimized solutions. This protocol stack consists of an application, network, link and physical layers, and it is modeled by different nano-network simulators. The design of the entire EM nano-communication architecture should be carried out taking into account the specific scope of the nano-network application. Hence, *one size fits all solutions* cannot be adopted, because each application domain in the nano-network has different requirements [19]. To the best of our knowledge, no studies have been conducted in the literature discussing SDN interlock with micro/nano-gateways, which represents the main motivation behind this paper.

2.6. Research methodology

The scope of this study is to answer the question of how the nano-network will be connected to the Internet. Accordingly, our proposal is motivated based on the fact that the heterogeneity of IoNT devices and their different requirements lead to an issue of data transformation between different protocols in the nano-network protocol stack. So, a heterogeneous SDN-based micro/nano-gateway is required to solve the problem of different technologies interaction on physical, data link, network and application layers by affording a protocol conversion mechanism to accommodate the huge amount of traffic coming from various smart nano-devices/sensors.

In our study, the data has been collected by primary means. Firstly, we proposed the micro/nano-gateway architecture and illustrate the

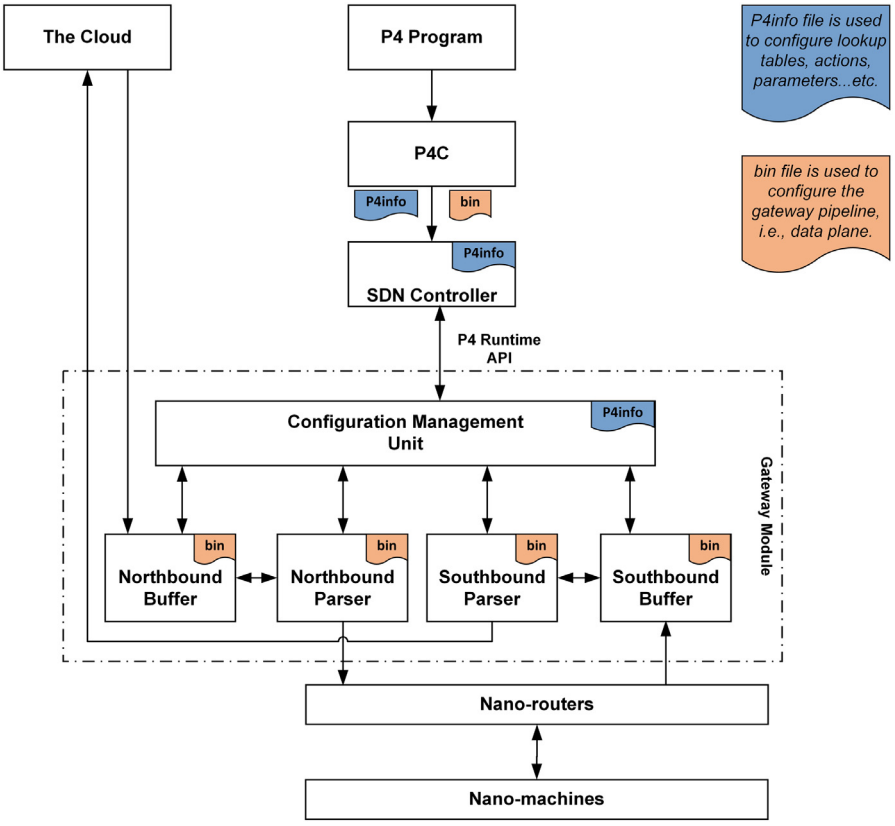

Fig. 4. Schematic diagram of the proposed micro/nano-gateway module.

block diagram of its components, describing their functionality and interaction in Section 3. Moreover, we illustrated two communication scenarios that explain nano-network communication; a single tenant and a multitenant in Section 3. Secondly, we implemented the micro/nano-gateway software module in Section 4, which consists of three components: the nano-router packet generator, the micro/nano-gateway module and the cloud application server.

In this study, our research was a quantitative experimental by controlling and manipulating the total number of connected nano-devices and the number of packets sent by each device, and observing their effects on the total average round-trip processing delay and the overall throughput of the micro/nano-gateway, which has been demonstrated in Section 5.

3. The proposed micro/nano-gateway module

In this section, we propose the micro/nano-gateway architecture and illustrate the block diagram of its components, describing their functionality and their interaction at a high level. In addition, we will propose two use cases that explain nano-network communication scenarios through the micro/nano-gateway.

3.1. Micro/nano-gateway architecture

A simple protocol conversion technique is used to convert the data stream between the nano-network domain and the traditional network domain. In order to achieve this goal, the micro/nano-gateway shall support multi-mode communication interfaces to link different wireless EM nano-networks, perform data analysis and data format conversion to achieve the integration of heterogeneous nano-network and promote the nano-network extensibility to the traditional network. Micro/nano-gateway acts as an interface between nano-nodes and the Internet by converting packet formats from different protocols in the nano-domain to the conventional network protocols, such as Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) and vice-versa. The proposed architecture of the gateway module consists of five components: configuration management unit, northbound buffer, northbound parser, southbound parser and southbound buffer. Fig. 4 illustrates a schematic diagram of the proposed micro/nano-gateway module.

3.1.1. Configuration management unit

The first component of the architecture is the configuration management unit, which is responsible for all configuration and management operations of the micro/nano-gateway. It represents the control plane of the module, as it manages the flow control between all the building blocks of the module, besides the communication with one or more SDN controllers. This unit is controlled by the P4 Runtime application programming interface (API) to translate commands coming from the SDN controller and the P4 compiler (P4C), which configure the data plane pipelines of the micro/nano-gateway based on the instructions coming from the P4 program. The P4C generates two files: the "P4info" file and the "bin" file. The "P4info" file captures the P4 program attributes, e.g., lookup tables, actions and parameters, while the "bin" file is used to configure the gateway pipeline/data plane. The micro/nano-gateway needs to learn some information from the SDN controller, such as the IP address of the cloud application server and its TCP/UDP port number. The SDN controller will pass these parameters to the micro/nano-gateway via the "P4info" file. Meanwhile, when there are multiple nano-routers connected to the micro/nano-gateway and multiple TCP/UDP tunnels are connected to different application servers, the SDN controller guides the configuration management unit for which port is connected to which tunnel endpoint. Hence the micro/nano-gateway can transparently forward the message of the nano-router towards the application. Besides, this unit handles communications with the data plane pipeline components, i.e., northbound/southbound parsers and buffers via the "bin" file. It provides scaling up/down

for the buffer size and assists southbound/northbound parsers in the protocol conversion process by acting as a database for all connected nano-routers by storing their corresponding device IDs, which will be used in the socket binding process with the cloud application server to receive the reply message on each connected nano-router. For that reason, the nano-router device ID is used as a listening port number from the cloud application server to such a nano-router. This binding guarantees to enable multithreading for different nano-routers for each received packet without data conflicts.

3.1.2. The southbound buffer

The second component is the southbound buffer, which represents the storage used to temporarily store packets coming from nano-routers while it is being processed and moved from the nano-network domain to the traditional network domain.

3.1.3. The southbound parser

The third component is the southbound parser, which is responsible for converting data format from the nano-network domain to the traditional network domain. When receiving a packet from a nano-router, the southbound parser will transform it into a more readable packet format that can be easily read and understood by the remote application server hosted on the could. It is worth noting that the southbound parser handles the incoming byte stream from different nano-routers and extracts their corresponding headers following the rules that are provided from the configuration management unit with the aid of the SDN controller. A simplistic illustration would be the southbound parser filters the nano-network packet by stripping off its header and keep the payload. Then, it checks the flag field of the nano-network packet header to confirm the corresponding communication scenario. Moreover, it performs a further extraction on the nano-router packet header by extracting the source nano-router ID and send it to the configuration management unit. After that, the southbound parser encapsulates the nano-router payload into a TCP/UDP packet using the destination IP address and port number that were provided from the configuration management unit and sends the nano-router packet to the cloud application server.

3.1.4. The northbound parser

The fourth component is the northbound parser, which is responsible for converting data format from the traditional network domain to the nano-network domain. When receiving a packet from the cloud application server, the northbound parser will transform it into a more readable packet format that can be easily read and understood by the nano-routers. The northbound parser handles the incoming byte stream from the cloud application server following the rules provided by the configuration management unit with the aid of the SDN controller. A simplistic illustration would be the northbound parser filters the cloud application server packet by stripping off its TCP/UDP header and keep the reply message. Then, it creates a nano-network packet header and adjusts the next-hop address field in the new packet header to be the source device ID of the nano-router that initially generates the packet. This step is achieved by recalling the source device ID value from the configuration management unit. After that, the northbound parser encapsulates the reply message from the server as a payload into the newly generated nano-packet header and sends the complete reply packet to the corresponding nano-router.

3.1.5. The northbound buffer

The fifth component is the northbound buffer, which represents the storage used to temporarily store packets coming from the cloud application server while it is being processed and moved from the traditional network domain to the nano-network domain.

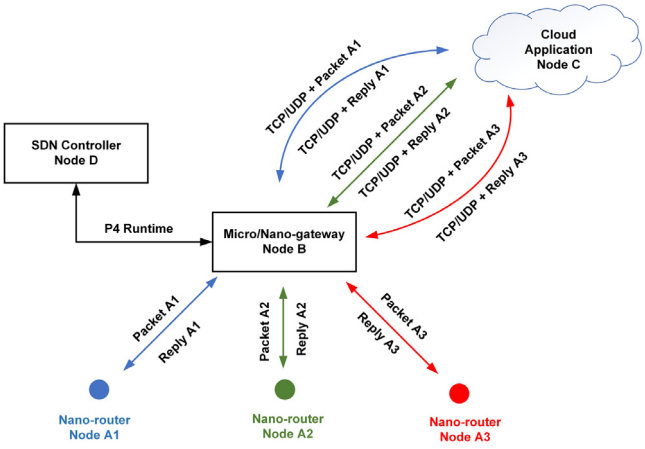

Fig. 5. High-level schematic diagram for the nano-network communication use cases.

3.2. The micro/nano-gateway communication use cases

Essentially, the communication to the micro/nano-gateway from nano-routers is established through its southbound interface using different routing protocols in the nano-domain. While from its northbound interface, it is controlled with P4 Runtime by the SDN controller as shown in Fig. 5. Nodes (A1), (A2) and (A3) represent regular nano-routers in the nano-network domain, while node (B) represents the micro/nano-gateway. Node (C) represents the application server of the nano-network hosted on the cloud that acts as an application enablement platform for the IoNT application, and node (D) represents the SDN controller.

Apparently, there are two possible communication use cases: nano-router to nano-router communication and nano-router to the cloud application communication. The micro/nano-gateway differentiates between these two types of communication using the flag field of the nano-network packet header. For instance, when the flag field set to "0" means nano-domain communication, i.e., the first use case. While, if the flag field set to "1" means a cloud application communication, i.e., the second use case.

3.2.1. Nano-router to nano-router communication (A1-B-A2) use case

In this use case, a certain nano-router is sending a packet to another nano-router, where (A1) sends a nano-network packet to (B). Node (B) will rewrite the header, i.e., check the corresponding device ID of (A2), decreases the TTL value and sends the packet to (A2). Accordingly, the message flow across the nodes will be (A1-B-A2). The P4 program runs on top of the SDN controller (D), and it is independent of the message flow. This communication use case is established under some constraints that the micro/nano-gateway (B) shall be aware of the maximum supported packet size and the maximum energy consumption of nano-routers (A1) and (A2). It is worth noting that it is not mandatory to involve the gateway in a nano-router to nano-router communication. To the best of our knowledge, multiple routing protocols in the nano-domain can forward the data as a multi-hop communication between nano-routers without the use of an external gateway [3,8]. Accordingly, the most promising communication use case for the gateway functionality is when the nano-router communicates with the Internet.

3.2.2. Nano-router to the cloud application (A1-B-C) use case

In this use case, the nano-router (A1) is sending a packet to the cloud application server (C) through the micro/nano-gateway (B). The message flow path is (A1-B-C). Node (A1) sends a nano-network packet to node (B), where node (B) strips off the nano-network packet header and keeps the payload to be encapsulated into TCP/UDP packet and send to the cloud application server (C). The same role of node (B)

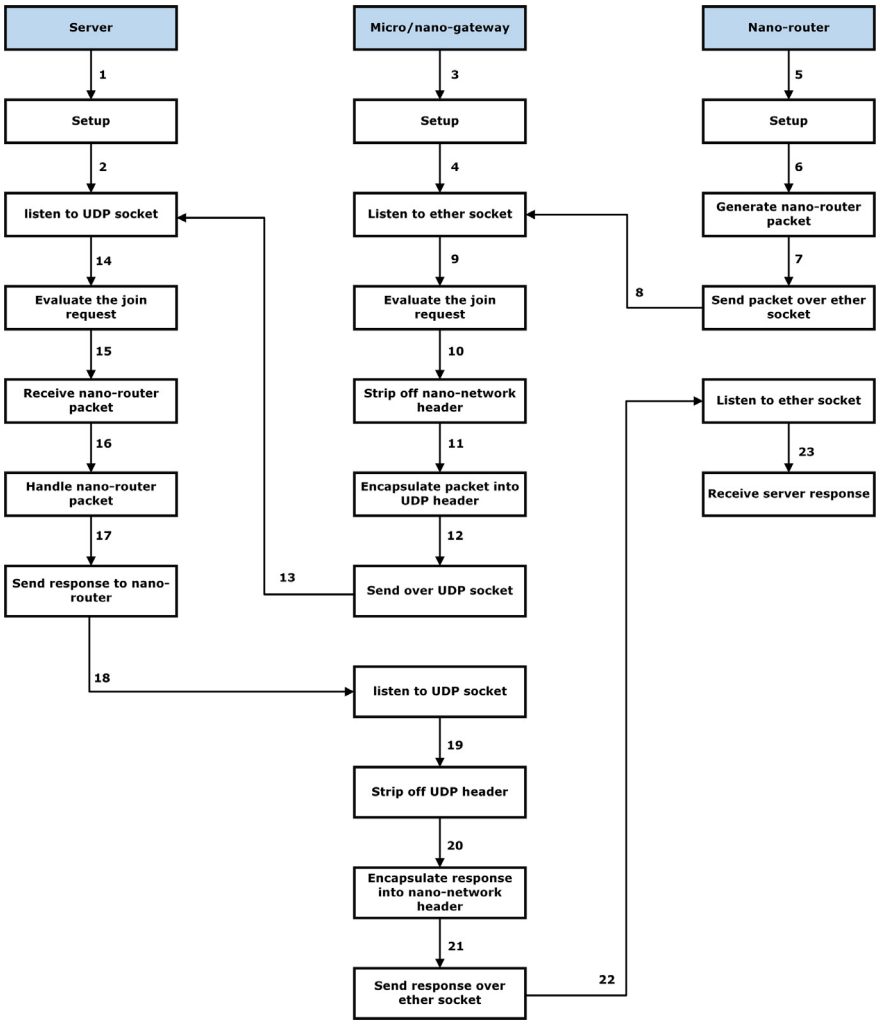

Fig. 6. The sequence diagram of the simulation mechanism.

will be repeated in the opposite communication direction for the reply message coming from the server following a path (C-B-A1). Node (B) will strip off TCP/UDP header and encapsulate the payload into the nano-network packet header and forward it to the nano-router (A1). This communication use case will be established under some constraints from both northbound and southbound interfaces of the micro/nano-gateway.

From the northbound interface.

- The micro/nano-gateway shall know the full identification of the cloud application server, e.g., IP address and port number.
- The role of the SDN controller is to instruct the micro/nano-gateway on which the physical nano-network port connects to which TCP/UDP endpoint.
- The cloud application server shall be able to understand the core functionality of the nano-router packet.

From the southbound interface.

- The micro/nano-gateway shall be aware of the maximum supported packet size and the maximum energy consumption of nano-routers.
- When multiple requests are coming from the nano-router, the SDN controller will guide the micro/nano-gateway about which port is connected to which tunnel endpoint.

4. Implementation of the proposed module

Based on the architecture of the micro/nano-gateway module proposed in Section 3, this section discusses the implementation of its algorithm. The proposed micro/nano-gateway module is implemented based on a software mechanism that integrates various heterogeneous wireless nano-network protocols and integrates them with traditional communication networks. Our focus is to design the micro/nano-gateway module. The simulation has been performed using Python 3.8.1 running on a machine with a 3.50 GHz Intel Core i7 processor and 32 GB RAM. The proposed software consists of three structures: the nano-router packet generator, the micro/nano-gateway module and the UDP cloud application server. Fig. 6 illustrates the sequence diagram of the simulation mechanism from both sides, e.g.; upstream transmission from the nano-router towards the cloud application server and downstream transmission from the cloud application server towards the nano-router.

The first structure is the nano-router packet generator, where the proposed software simulates a light version of the nano-network simulator NanoSim discussed in [19,20]. In our proposed software, the message processing unit creates periodical packets with a packet size that can be tuned by the user. The network layer adds the network header independently on the used routing strategy, while the MAC layer provides a simple asynchronous strategy that has been designed without executing any kind of control, as the packet is transmitted from the network layer to the physical interface of the nano-router

directly based on the fact that all information that is useful for delivering messages to the destination node is already included within the header added at the network layer. This technique is called transparent-MAC. Hence, the generated byte stream includes the packet header associated with the encapsulated payload of a nano-router, which is transmitted through the physical interface of the nano-router towards the micro/nano-gateway.

The second structure is the micro/nano-gateway module, which is up and waiting to sniff different packets coming from the nano-router packet generator. Upon receiving a certain packet, the gateway starts a simple evaluation process by checking the packet header to confirm the communication direction is upstream transmission towards the cloud application server by checking the flag field inside the packet header. Also, it checks the next-hop address to confirm that it is its device ID. After that, the micro/nano-gateway strips of the nano-packet header and saves the source nano-router device ID in its configuration management unit, which uses this value to bind the listening socket with the cloud application server. Whereas the micro/nano-gateway gets the IP address and port number of the remote server from the configuration management unit, it encapsulates the payload into the UDP packet and sends it over UDP/IP socket to the cloud application server. Meanwhile, the configuration management unit creates multithreading in case multiple nano-routers are connected. This technique will execute multiple versions of the gateway code in parallel for all nano-routers intents. Hence, other nano-routers can connect simultaneously to the micro/nano-gateway.

The third structure is the UDP cloud application server, which is up and running tending on its listening port to receive connection requests from the micro/nano-gateway. After receiving a connection, the server evaluates the join request to confirm the correct destination IP address and port number. After such confirmation, the server program will accept the nano-router packet and add the newly connected nano-router device ID to the connection list. Meanwhile, the server program will run multithreading to allow multiple nano-routers connected from the micro/nano-gateway at the same time. After the server handles the nano-router packet, it sends the corresponding response to the micro/nano-gateway, which is listening for the reply message on its communication port that is mapped with a port number that has the same value as the source nano-router device ID. Accordingly, the micro/nano-gateway decapsulates the reply message from the UDP header and creates a nano-network packet header by adjusting the next-hop address field in the new packet header to be the source device ID of the nano-router that initially send the packet. Then, the micro/nano-gateway sends the reply message to the corresponding nano-router. Fig. 7 illustrates the flowchart of the methodology used by the micro/nano-gateway algorithm.

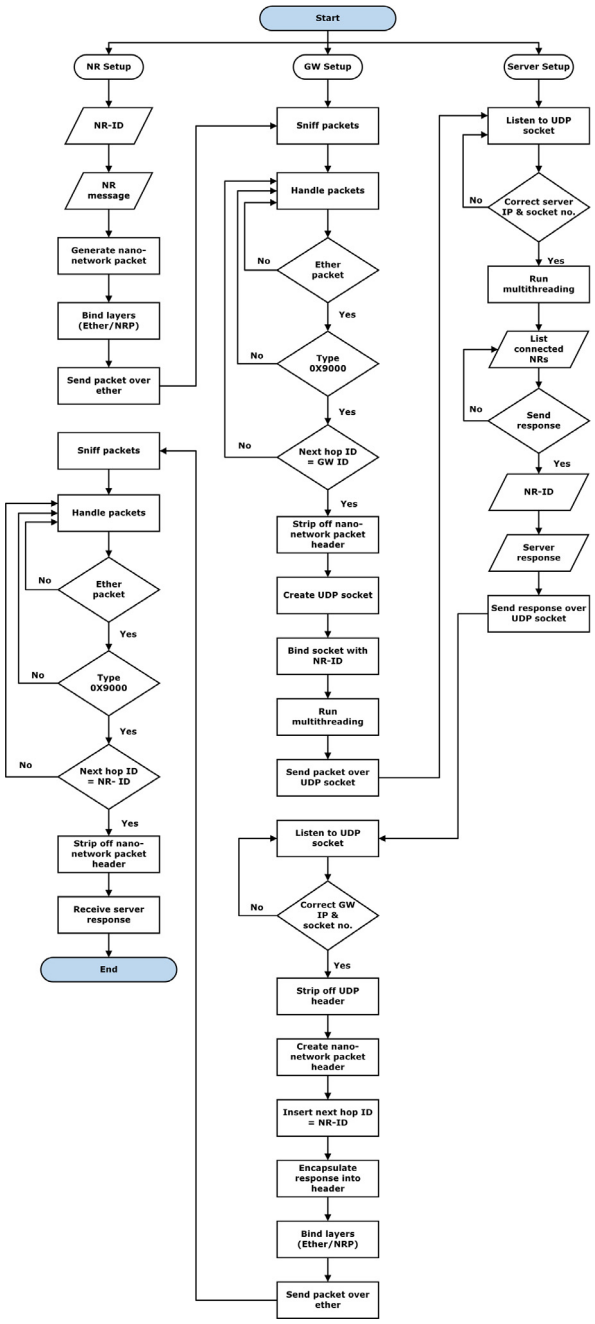

Fig. 7. Flowchart represents the general structure of the algorithm used by the micro/nano-gateway module.

5. Performance evaluation

In this section, we evaluate the performance of the proposed micro/nano-gateway algorithm based on two communication scenarios; single-tenant and multitenant by manipulating the total number of connected nano-routers and the number of packets sent by each nano-router, taking into consideration the following evaluation metrics:

- Upstream processing delay: it is the execution time that the micro/nano-gateway takes to decapsulate the nano-packet header, encapsulate the payload into the UDP header and send the packet to the cloud application server.
- Downstream processing delay: it is the execution time that the micro/nano-gateway takes to decapsulate the UDP header, encapsulate the payload into a nano-header and send the reply back to the corresponding nano-router.
- Average processing delay: it is the round-trip time that the micro/nano-gateway takes to process the nano-packet while traveling in both upstream and downstream directions.
- Upstream throughput: it is the total number of packets per second that the micro/nano-gateway can handle in the upstream direction from the nano-router to the cloud application server.
- Downstream throughput: it is the total number of packets per second that the micro/nano-gateway can handle in the downstream direction from the cloud application server to the nano-router.

5.1. Single tenant scenario

A single tenant scenario, in which one nano-router is connecting to the micro/nano-gateway and sending multiple packets. In this scenario, we will evaluate the average processing delay and throughput of the

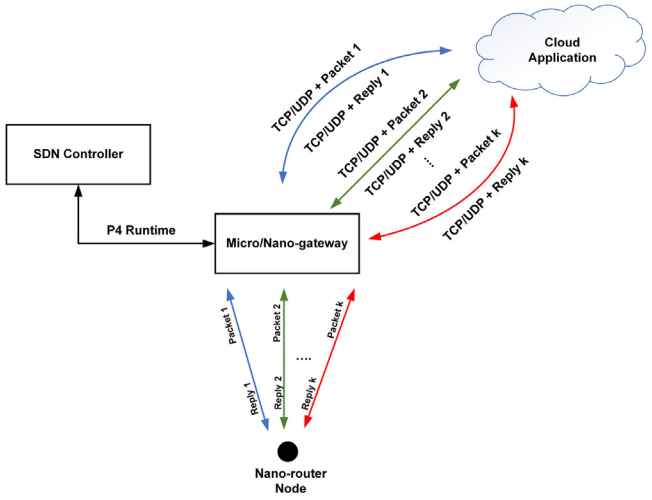

Fig. 8. A schematic diagram for the single tenant communication scenario.

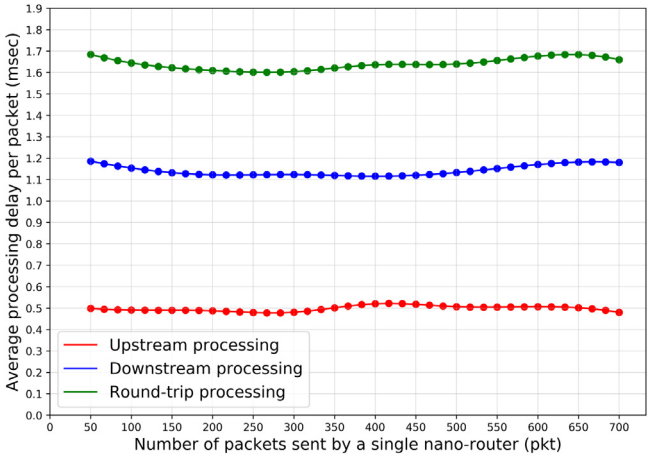

Fig. 9. The relation between average processing delay and number of packets in a single tenant scenario.

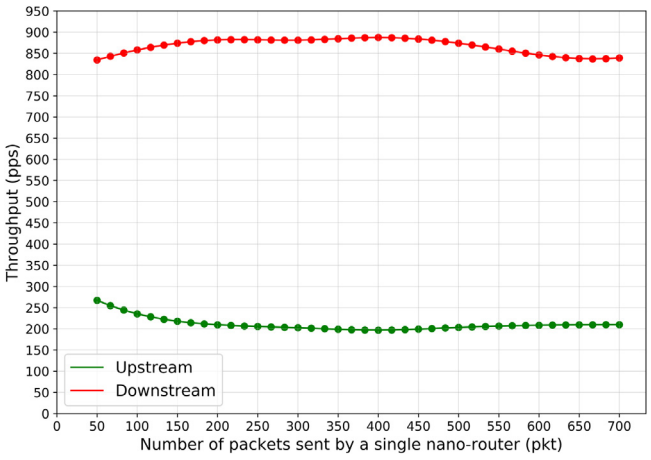

Fig. 10. The relation between throughput and number of packets in a single tenant scenario.

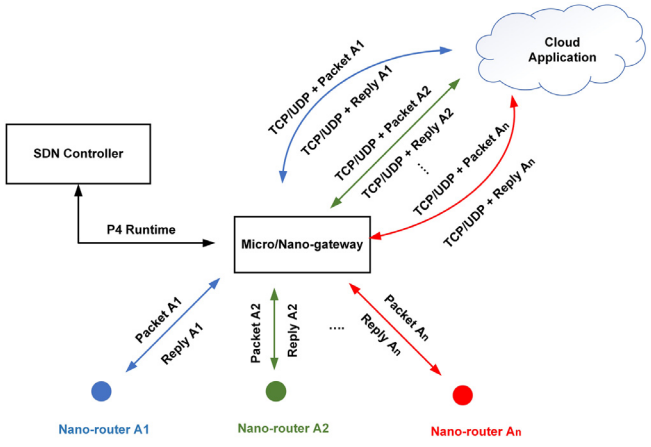

Fig. 11. A schematic diagram for the multitenant-single packet use case.

micro/nano-gateway based on manipulating the number of simultaneous packets sent by the nano-router. Fig. 8 illustrates the schematic diagram of the single tenant communication scenario.

We have tested the performance of the micro/nano-gateway by changing the number of packets sent by the nano-router from 50 to 700 packets, and check the average processing delay of the upstream, the downstream and the round-trip of the micro/nano-gateway, besides its upstream and downstream throughput. Fig. 9 shows that the micro-nano/gateway takes a shorter time to process packets in the upstream direction, while it takes a longer time to process packets in the downstream direction. It means that the packet decapsulation from a nano-packet and encapsulation into the UDP header takes lower processing time than decapsulating the payload from the UDP header and encapsulating it into the nano-packet header. The results show that the upstream processing delay oscillates around 0.5 ms, while the upstream processing delay oscillates between 1.1 ms and 1.2 ms. This is due to the packet structure format and the header size of the messages originating from the nano-router is smaller than the header size of the messages originating from the server. Moreover, increasing the total number of packets sent by the nano-router does not have a high effect on the total average round-trip processing delay of the micro/nano-gateway, as it oscillates between 1.6 ms to 1.7 ms.

Fig. 10 depicts the upstream and downstream throughput of the micro/nano-gateway. It shows that downstream throughput is almost four times higher than upstream throughput. Also, as the total number of packets sent by the nano-router increases, the upstream and downstream throughput of the micro/nano-gateway decreases slightly.

5.2. Multitenant scenario

A multitenant scenario in which multiple nano-routers are connecting and sending packets simultaneously to the micro/nano-gateway. In this scenario, we will study two use cases; multitenant-single packet and multitenant-multiple packets.

5.2.1. Multitenant-single packet use case

In a multitenant-single packet use case, multiple nano-routers are connected simultaneously. Each nano-router sends one packet to the micro/nano-gateway. Fig. 11 illustrates the schematic diagram of the multitenant-single packet use case.

We have tested the performance by manipulating the total number of connected nano-routers from 50 to 700 nano-routers. Each nano-router sends a single packet to the micro/nano-gateway. Fig. 12 shows that the micro-nano/gateway takes a longer time to process packets in the downstream direction, while it takes a very short time to process packets in the upstream direction. This means that the packet decapsulation from a nano-packet and encapsulation into the UDP packet takes a very low processing time in a multitenant-single packet scenario. While decapsulating the payload from the UDP packet and encapsulating it into a nano-packet takes a longer time. This is due

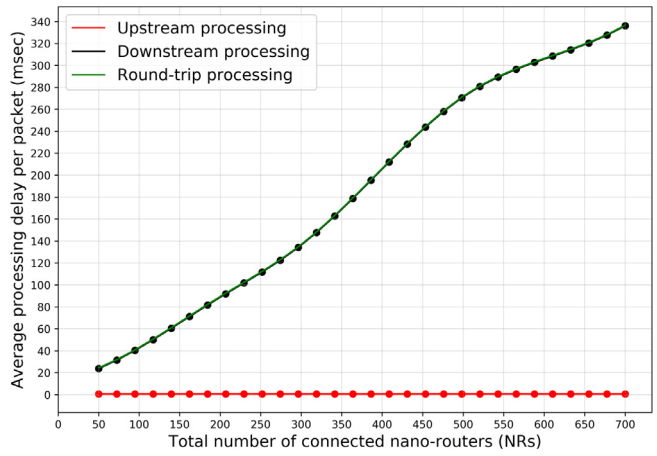

Fig. 12. Average processing delay of the micro/nano-gateway in multitenant-single packet use case.

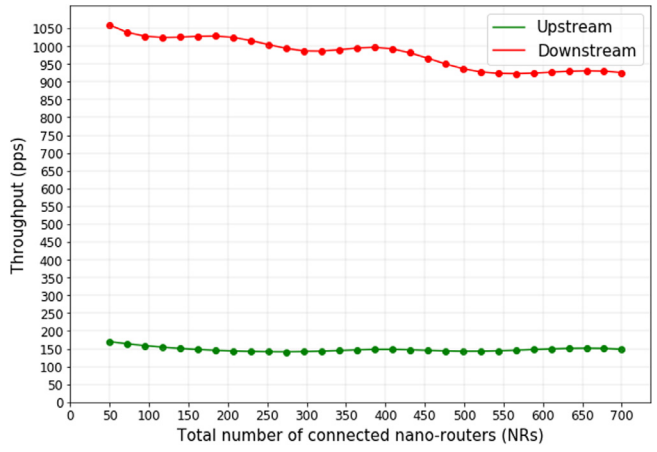

Fig. 13. Downstream and upstream throughput of the micro/nano-gateway in multitenant-single packet use case.

to the packet structure format and the header size of the messages originating from the nano-router is smaller than the header size of the messages originating from the server. Moreover, as the total number of connected nano-routers increases, the upstream processing delay is almost constant for different numbers of connected nano-routers, while the downstream processing time increases. Accordingly, the total average round-trip processing delay of the micro/nano-gateway increases as well. This means the total number of connected nano-routers and the downstream processing time have a direct effect on the scalability performance of the gateway in the multitenant-single packet use case.

Fig. 13 describes the relationship between the micro/nano-gateway downstream and upstream throughput and the total number of connected nano-routers. It should be pointed that as the total number of connected nano-routers increases, the upstream throughput of the micro/nano-gateway will decrease slightly. This is due to the increased number of packets generated by multiple nano-routers will increase the amount of data injected into the micro-nano-gateway. This data injection leads to a decrease in upstream throughput. The same concept with the downstream throughput, as the total number of packets generated from the cloud application server increases, the data injected into the micro/nano-gateway will increase, which will lead to a decrease in the downstream throughput.

5.2.2. Multitenant-multiple packets use case

In the multitenant-multiple packets use case, multiple nano-routers are connected simultaneously. Each nano-router sends multiple packets to the micro/nano-gateway. Fig. 14 illustrates the schematic diagram of the multitenant-multiple packet use case.

We have tested the performance of the average processing delay and throughput of the micro/nano-gateway by manipulating the total number of connected nano-routers from 5 to 95 nano-routers with different packet rates sent by each nano-router to the micro/nano-gateway. We have tested the scenario based on that each nano-router sends 1, 2, 4 and 8 packets.

Fig. 15 shows that the micro-nano/gateway takes a long time to process packets in the downstream direction, while it takes a very short time to process packets in the upstream direction. This means that the upstream and downstream processing delays have the same effect as the multi tenants-single packet use case. However, more importantly, as the packet rate increases or the total number of connected nano-routers increases, the total average round-trip processing delay of the micro/nano-gateway increases. Because the micro-nano/gateway consumes more processing time handling concurrent multithreading and performs sockets binding for different connected nano-routers in addition to the increased number of packets sent by each nano-router. So, the total number of connected nano-routers and the packet rate of each nano-router have a direct effect on the scalability performance of the micro/nano-gateway in the multitenant-multiple packet use case.

Fig. 16 describes the relationship between the upstream throughput and the total number of connected nano-routers for different packet rates. As the total number of connected nano-routers increases, the upstream throughput will decrease, besides as the total number of packets sent by the nano-router increases, the upstream throughput will decrease as well. Hence, both the total number of connected nano-routers and the packet rate of each nano-router have a dramatic effect on the upstream throughput of the micro/nano-gateway. This is because increasing both the total number of connected nano-routers and the packet rate of each one means increasing the amount of data pushed into the micro/nano-gateway, accordingly, the upstream throughput decreases. Also, it is obvious that for a small number of connected nano-routers, for instance, less than 15 nano-routers, the upstream throughput decrease of small packet rates (1–2 packets) is higher than the upstream throughput decrease of the higher packet rates (8 packets). While for a large number of connected nano-routers, for instance, 90 nano-routers, the upstream throughput decrease still exists, but with some sort of convergent differences.

On the other hand, Fig. 17 depicts the relation between the downstream throughput and the total number of connected nano-routers for different packet rates. The total number of connected nano-routers has not had a massive effect on the downstream throughput of the micro/nano-gateway. While the total number of packets sent by the cloud application server increases, the downstream throughput will decrease. Hence, the cloud application server packet rate has a direct effect on the downstream throughput of the micro/nano-gateway in the downstream direction.

6. Qualitative analysis

In this section, we provide a qualitative comparison between the proposed micro/nano-gateway and other related IoT gateways that have been discussed in the literature. The qualitative analysis has been achieved based on the following criteria:

- Infrastructure architecture: it describes the actual components and their design of the system used. In the IoNT, data flows from nano-sensors/actuators towards a set of nano-routers, which aggregate the data and forward it to the micro/nano-gateway. While in the IoT, data flows from regular sensors/actuators towards the gateway interface.
- Technology used in the architecture: different technologies ranging from SDN and NFV to fog computing and multi-core processing have been adopted in the design and implementation of both the IoT and IoNT gateways.

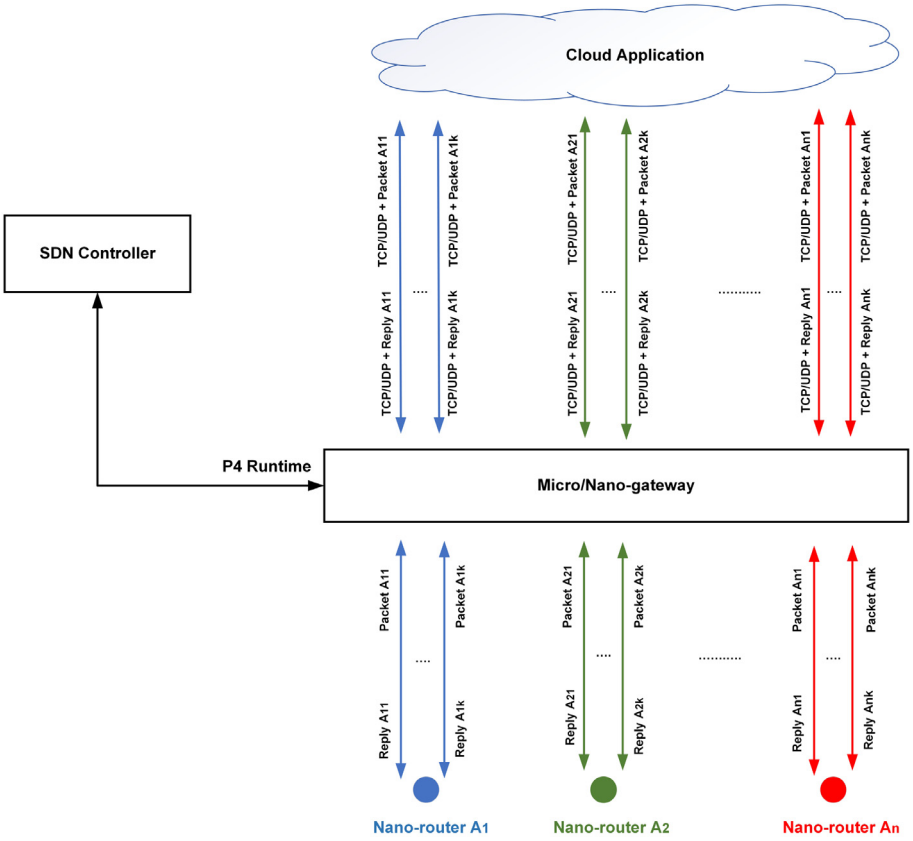

Fig. 14. A schematic diagram for the multitenant-multiple packet use case.

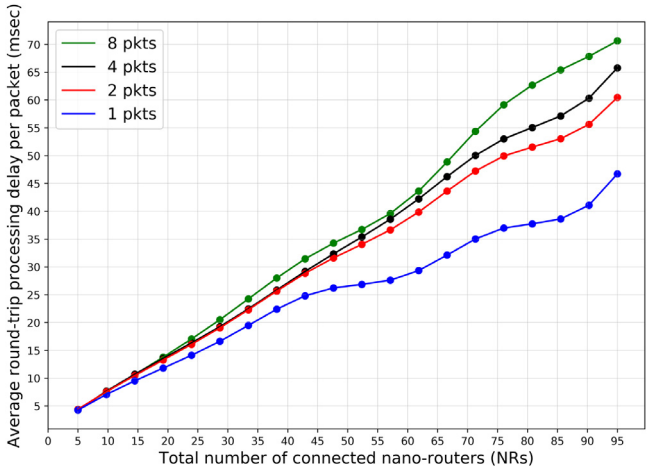

Fig. 15. Average processing delay of the micro/nano-gateway in multitenant-multiple packets use case.

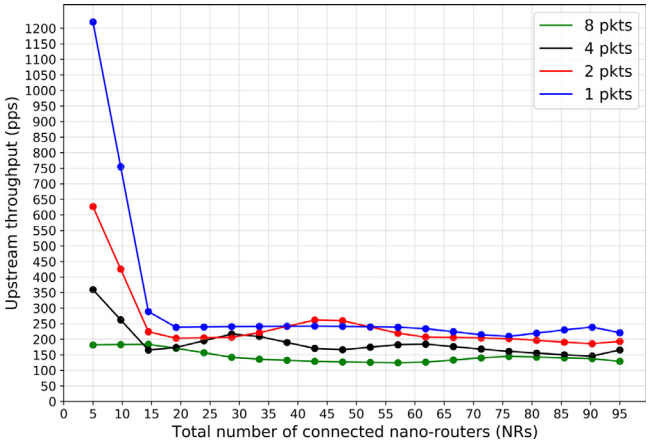

Fig. 16. Upstream throughput of the micro/nano-gateway in multitenant-multiple packets use case.

- Interoperability: the ability of the gateway and its associated software to exchange data between different domains and operate with various communication protocols.
- Standard interface: the use of standard interfaces that support service definitions for creating APIs to access, exchange and analysis data flow and diagnostic information.
- Centralized vs. distributed: it is the gateway system capability to operate in a centralized or distributed manner.

Table 4 provides a comparison between the proposed micro/nano-gateway and other related work of IoT gateways proposed in the literature. Our proposed SDN-based micro/nano-gateway is presented to work with the IoNT infrastructure using an SDN controller based on P4 language. The proposed micro/nano-gateway architecture overcomes the heterogeneity and interoperability problems of the IoNT domain. It affords upstream and downstream communication between the nano-network and the Internet, despite the unconventional communication protocols used in the nano-network protocol stack, the various packet formats, or the limited computing/processing capabilities of several nano-devices. The qualitative comparison identifies the lack of use of the gateway architecture for the IoNT paradigm. Also, it emphasizes the fact that future research is needed to develop micro/nano-gateway interfaces enabling nano-networks to be fully operational and communicated with the Internet.

Table 4
Comparison between the proposed micro/nano-gateway and other related work of IoT gateways proposed in the literature.

The gateway	Infrastructure	Technology	Interoperability	Standard interface	Distributed
Proposed in [6]	IoT	SDN/NFV	No	Yes	Yes
Proposed in [7]	IoT	SDN/NFV/Fog computing	Yes	Yes	No
Proposed in [9]	IoT	SDN	N/A	Yes	No
Proposed in [34]	IoT	SDN/NFV	No	Yes	No
Proposed in [35]	IoT	SDN	Yes	Yes	N/A
Proposed in [38]	IoT	Multi-core processing	Yes	Yes	No
Proposed in [36]	IoT	SDN/OpenFlow	Yes	Yes	Yes
Proposed in this paper	IoNT	SDN/P4	Yes	Yes	Yes

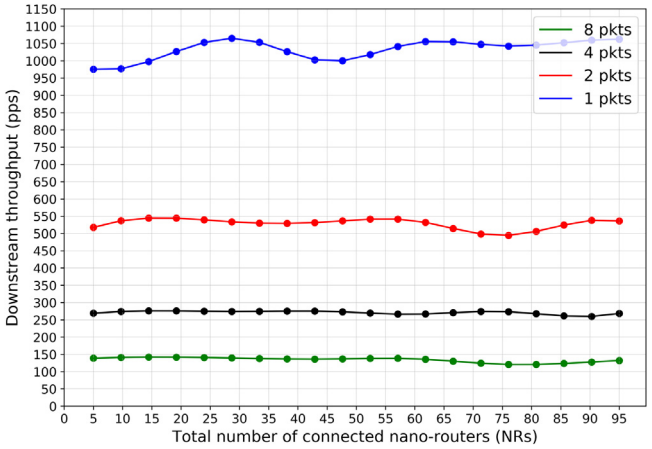

Fig. 17. Downstream throughput of the micro/nano-gateway in multitenant-multiple packets use case.

7. Conclusion and future work

Interoperability and heterogeneity are the main features of IoNT. Exchanging different data formats between several communication protocols in diverse nano-network applications leads to a lack of cooperation and capability mismatch between devices, which affects the overall performance of the nano-network. SDN approach can bring flexibility by allowing different nano-devices to be connected to heterogeneous nano-networks and communicate in different application domains thanks to the micro/nano-gateways. The use of an SDN-enabled gateway offers good solutions for the aforementioned issues, as it provides an IoNT flexible environment to manage a huge amount of data formats and various available protocols. In this article, we propose a micro/nano-gateway architecture and a software module based on SDN technology. The proposed software module interacts between the nano-network domain and the traditional network domain by providing a protocol mapping mechanism that allows an intermediary data format conversion for various protocols of the nano-network domain to the traditional domain and vice versa. Therefore, providing an access to the nano-network from the Internet and facilitate the development of different applications. A prototype of the proposed architecture is provided with a detailed description of the functionality behind each building block. Moreover, a set of experiments are conducted to evaluate the performance of the proposed algorithm based on two main scenarios as a single tenant and a multitenant. The evaluation results show the direct effect of the total number of connected nano-routers and the number of packets sent by each nano-router on the total average round-trip processing delay of the micro/nano-gateway and its overall throughput.

We envision some future work to extend the functionality of the micro/nano-gateway module to be provisioned as virtual network functions and chain them together to form a complete network service. It will allow the main algorithm of the micro/nano-gateway to be embedded as virtual network functions that can be chained together to form a Service Function Chaining (SFC), while SDN establishes the paths between these VNFs and provisions the network traffic providing a dynamic and flexible softwarized gateway for the nano-network. Assuming the micro/nano-gateway cannot do the protocol conversion itself over a hardware appliance due to limited computing resources, its functionality can be virtualized. So, the micro/nano-gateway will send the received nano-network packet transparently to a cloud application that represents its virtual network function. Hence, there will be two applications hosted on the cloud, one application represents the core functionality of the virtualized micro/nano-gateway, and another application represents the cloud application server. Both applications will communicate together with a suitable API.

Furthermore, fog computing is a potential analysis that can be considered in our future work by studying the connection possibility of the gateway module with a fog node closer to the gateway appliance and check how the RTT will change.

Another open research challenge is the scalability of the micro/nano-gateway, which is defined as the capability to handle a large number of connected nano-routers. Moreover, the performance of the proposed module can be evaluated with different metrics such as the nano-packet size, the buffer size of the micro/nano-gateway and the energy consumption rate providing a study on how these metrics affect the overall performance of the micro/nano-gateway.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been supported by the “Agencia Estatal de Investigación” of Spain under project PID2019-108713RB-C51/ AEI/ 10.13039/501100011033 and “Agència de Gestió d’Ajuts Universitaris i de Recerca” (AGAUR), Spain of the “Generalitat de Catalunya” under FI-AGAUR grant.

References

[1] I. Alam, K. Sharif, F. Li, Z. Latif, M. Monjurul Karim, B. Nour, S. Biswas, Y. Wang, IoT virtualization: A survey of software definition & function virtualization techniques for internet of things, 2019, arXiv preprint arXiv:1902.10910.

[2] I.F. Akyildiz, J.M. Jornet, M. Pierobon, Nanonetworks: A new frontier in communications, Commun. ACM 54 (11) (2011) 84–89, http://dx.doi.org/10.1145/2018396.2018417, URL http://doi.acm.org/10.1145/2018396.2018417.

[3] F. Lemic, S. Abadal, W. Tavernier, P. Stroobant, D. Colle, E. Alarcón, J. Marquez-Barja, J. Famaey, Survey on terahertz nanocommunication and networking: A top-down perspective, IEEE J. Sel. Areas Commun. 39 (6) (2021) 1506–1543, http://dx.doi.org/10.1109/JSAC.2021.3071837.

[4] A. Galal, X. Hesselbach, Nano-networks communication architecture: Modeling and functions, Nano Commun. Netw. 17 (2018) 45–62, http://dx.doi.org/10.1016/j.nancom.2018.07.001, URL http://www.sciencedirect.com/science/article/pii/S1878778918300164.

[5] F. Al-Turjman, A cognitive routing protocol for bio-inspired networking in the internet of nano-things (IoNT), Mob. Netw. Appl. (2017) 1–15, http://dx.doi.org/10.1007/s11036-017-0940-8.

[6] C. Mouradian, N.T. Jahromi, R.H. Glitho, NFV and SDN-based distributed IoT gateway for large-scale disaster management, IEEE Internet Things J. 5 (5) (2018) 4119–4131, http://dx.doi.org/10.1109/JIOT.2018.2867255.
[7] O. Salman, I. Elhajj, A. Kayssi, A. Chehab, Edge computing enabling the internet of things, in: 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), 2015, pp. 603–608, http://dx.doi.org/10.1109/WF-IoT.2015.7389122.
[8] A. Galal, X. Hesselbach, Probability-based path discovery protocol for electromagnetic nano-networks, Comput. Netw. 174 (2020) 107246, http://dx.doi.org/10.1016/j.comnet.2020.107246, URL http://www.sciencedirect.com/science/article/pii/S1389128619308801.
[9] Y. Li, X. Su, J. Riekki, T. Kanter, R. Rahmani, A SDN-based architecture for horizontal Internet of Things services, in: 2016 IEEE International Conference on Communications, ICC, 2016, pp. 1–7, http://dx.doi.org/10.1109/ICC.2016.7511053.
[10] X.-W. Yao, Y.-C.-G. Wu, W. Huang, Routing techniques in wireless nanonetworks: A survey, Nano Commun. Netw. 21 (2019) 100250, http://dx.doi.org/10.1016/j.nancom.2019.100250, URL http://www.sciencedirect.com/science/article/pii/S1878778919300195.
[11] A. Tsioliaridou, C. Liaskos, S. Ioannidis, A. Pitsillides, CORONA: A coordinate and routing system for nanonetworks, in: Proceedings of the Second Annual International Conference on Nanoscale Computing and Communication, NANOCOM' 15, ACM, New York, NY, USA, 2015, pp. 18:1–18:6, http://dx.doi.org/10.1145/2800795.2800809, URL http://doi.acm.org/10.1145/2800795.2800809.
[12] A. Tsioliaridou, C. Liaskos, L. Pachis, S. Ioannidis, A. Pitsillides, N3: Addressing and routing in 3D nanonetworks, in: 2016 23rd International Conference on Telecommunications, ICT, 2016, pp. 1–6, http://dx.doi.org/10.1109/ICT.2016.7500372.
[13] C. Liaskos, A. Tsioliaridou, S. Ioannidis, N. Kantartzis, A. Pitsillides, A deployable routing system for nanonetworks, in: 2016 IEEE International Conference on Communications, ICC, 2016, pp. 1–6, http://Dx.Doi.Org/10.1109/ICC.2016.7511151.
[14] M. Pierobon, J.M. Jornet, N. Akkari, S. Almasri, I.F. Akyildiz, A routing framework for energy harvesting wireless nanosensor networks in the terahertz band, Wirel. Netw. 20 (5) (2014) 1169–1183, http://dx.doi.org/10.1007/s11276-013-0665-y, URL http://dx.doi.org/10.1007/s11276-013-0665-y.
[15] F. Afsana, M. Asif-Ur-Rahman, M.R. Ahmed, M. Mahmud, M.S. Kaiser, An energy conserving routing scheme for wireless body sensor nanonetwork communication, IEEE Access 6 (2018) 9186–9200, http://dx.doi.org/10.1109/ACCESS.2018.2789437.
[16] J.M. Jornet, J.C. Pujol, J.S. Pareta, PHLAME: A physical layer aware MAC protocol for electromagnetic nanonetworks in the terahertz band, Nano Commun. Netw. 3 (1) (2012) 74–81, http://dx.doi.org/10.1016/j.nancom.2012.01.006, URL http://www.sciencedirect.com/science/article/pii/S1878778912000075.
[17] S. Mohrehkesh, M.C. Weigle, S.K. Das, DRIH-MAC: A distributed receiver-initiated harvesting-aware MAC for nanonetworks, IEEE Trans. Mol. Biol. Multi-Scale Commun. 1 (1) (2015) 97–110, http://dx.doi.org/10.1109/TMBMC.2015.2465519.
[18] S. D'Oro, L. Galluccio, G. Morabito, S. Palazzo, A timing channel-based MAC protocol for energy-efficient nanonetworks, Nano Commun. Netw. 6 (2) (2015) 39–50, http://dx.doi.org/10.1016/j.nancom.2015.01.005, pervasive and Ubiquitous Environment Interactions with Nano Things, URL http://www.sciencedirect.com/science/article/pii/S187877891500006X.
[19] G. Piro, L.A. Grieco, G. Boggia, P. Camarda, Nano-Sim: Simulating Electromagnetic-Based Nanonetworks in the Network Simulator, vol. 3, ACM, 2013, http://dx.doi.org/10.4108/icst.simutools.2013.251699.
[20] G. Piro, L.A. Grieco, G. Boggia, P. Camarda, Simulating wireless nano sensor networks in the NS-3 platform, in: 2013 27th International Conference on Advanced Information Networking and Applications Workshops, 2013, pp. 67–74, http://dx.doi.org/10.1109/WAINA.2013.20.
[21] N. Rikhtegar, M. Keshtgari, Z. Ronaghi, EEWNSN: Energy efficient wireless nano sensor network MAC protocol for communications in the terahertz band, Wirel. Pers. Commun. 97 (1) (2017) 521–537, http://dx.doi.org/10.1007/s11277-017-4517-4, URL http://dx.doi.org/10.1007/s11277-017-4517-4.
[22] A. Mestres, S. Abadal, J. Torrellas, E. Alarcón, A. Cabellos-Aparicio, A MAC protocol for reliable broadcast communications in wireless network-on-chip, in: Proceedings of the 9th International Workshop on Network on Chip Architectures, NoCArc'16, Association for Computing Machinery, New York, NY, USA, 2016, pp. 21–26, http://dx.doi.org/10.1145/2994133.2994137, URL http://dx.doi.org/10.1145/2994133.2994137.
[23] F. Lemic, R.U. Akbar, J. Marquez-Barja, J. Famaey, Assessing the reliability of energy harvesting terahertz nanonetworks for controlling software-defined metamaterials, in: Proceedings of the Sixth Annual ACM International Conference on Nanoscale Computing and Communication, NANOCOM '19, Association for Computing Machinery, New York, NY, USA, 2019, http://dx.doi.org/10.1145/3345312.3345467, URL http://dx.doi.org/10.1145/3345312.3345467.
[24] J.M. Jornet, I.F. Akyildiz, Femtosecond-long pulse-based modulation for terahertz band communication in nanonetworks, IEEE Trans. Commun. 62 (5) (2014) 1742–1754, http://dx.doi.org/10.1109/TCOMM.2014.033014.130403.
[25] H. Mabed, Enhanced spread in time on-off keying technique for dense Terahertz nanonetworks, in: 2017 IEEE Symposium on Computers and Communications, ISCC, 2017, pp. 710–716, http://dx.doi.org/10.1109/ISCC.2017.8024611.
[26] H. Mabed, J. Bourgeois, A flexible medium access control protocol for dense terahertz nanonetworks, in: Proceedings of the 5th ACM International Conference on Nanoscale Computing and Communication, NANOCOM '18, Association for Computing Machinery, New York, NY, USA, 2018, http://dx.doi.org/10.1145/3233188.3233198, URL http://dx.doi.org/10.1145/3233188.3233198.
[27] X. Yu, J. Baylon, P. Wettin, D. Heo, P.P. Pande, S. Mirabbasi, Architecture and design of multichannel millimeter-wave wireless NoC, IEEE Des. Test 31 (6) (2014) 19–28, http://dx.doi.org/10.1109/MDAT.2014.2322995.
[28] C. Han, A.O. Bicen, I.F. Akyildiz, Multi-wideband waveform design for distance-adaptive wireless communications in the terahertz band, IEEE Trans. Signal Process. 64 (4) (2016) 910–922, http://dx.doi.org/10.1109/TSP.2015.2498133.
[29] M. Kocaoglu, O.B. Akan, Minimum energy channel codes for nanoscale wireless communications, IEEE Trans. Wireless Commun. 12 (4) (2013) 1492–1500, http://dx.doi.org/10.1109/TWC.2013.022113.2190.
[30] M.A. Zainuddin, E. Dedu, J. Bourgeois, SBN: Simple block nanocode for nanocommunications, in: Proceedings of the 3rd ACM International Conference on Nanoscale Computing and Communication, NANOCOM'16, Association for Computing Machinery, New York, NY, USA, 2016, http://dx.doi.org/10.1145/2967446.2967452, URL http://dx.doi.org/10.1145/2967446.2967452.
[31] N. Boillot, D. Dhoutaut, J. Bourgeois, Scalable simulation of wireless electromagnetic nanonetworks, in: 2015 IEEE 13th International Conference on Embedded and Ubiquitous Computing, 2015, pp. 83–89, http://dx.doi.org/10.1109/EUC.2015.38.
[32] D. Dhoutaut, T. Arrabal, E. Dedu, Bit simulator, an electromagnetic nanonetworks simulator, in: Proceedings of the 5th ACM International Conference on Nanoscale Computing and Communication, in: NANOCOM '18, Association for Computing Machinery, New York, NY, USA, 2018, http://dx.doi.org/10.1145/3233188.3233205, URL http://dx.doi.org/10.1145/3233188.3233205.
[33] Z. Hossain, Q. Xia, J.M. Jornet, TeraSim: An ns-3 extension to simulate Terahertz-band communication networks, Nano Commun. Netw. 17 (2018) 36–44, http://dx.doi.org/10.1016/j.nancom.2018.08.001, URL http://www.sciencedirect.com/science/article/pii/S1878778918300772.
[34] M. Ojo, D. Adami, S. Giordano, A SDN-IoT architecture with NFV implementation, in: 2016 IEEE Globecom Workshops, GC Wkshps, 2016, pp. 1–6, http://dx.doi.org/10.1109/GLOCOMW.2016.7848825.
[35] J. Romero-Gázquez, M.V. Bueno-Delgado, Software architecture solution based on SDN for an industrial IoT scenario, Wirel. Commun. Mob. Comput. 2018 (2018) 2946575, http://dx.doi.org/10.1155/2018/2946575, URL http://dx.doi.org/10.1155/2018/2946575.
[36] Y. Li, X. Su, J. Riekki, T. Kanter, R. Rahmani, A SDN-based architecture for horizontal Internet of Things services, in: 2016 IEEE International Conference on Communications, ICC, 2016, pp. 1–7, http://dx.doi.org/10.1109/ICC.2016.7511053.
[37] F. Tietz, D. Brandão, L.F. Alves, Development of an internet of things gateway applied to a multitask industrial plant, in: 2018 13th IEEE International Conference on Industry Applications, INDUSCON, 2018, pp. 917–923, http://dx.doi.org/10.1109/INDUSCON.2018.8627273.
[38] L. Shimei, Z. Jianhong, L. Enfeng, H. Gang, Design of industrial internet of things gateway with multi-source data processing, in: 2020 International Conference on Computer Engineering and Application, ICCEA, 2020, pp. 232–236, http://dx.doi.org/10.1109/ICCEA50009.2020.00058.
[39] A. Rahman, M.K. Nasir, Z. Rahman, A. Mosavi, S. S., B. Minaei-Bidgoli, DistBlockBuilding: A distributed blockchain-based SDN-IoT network for smart building management, IEEE Access 8 (2020) 140008–140018, http://dx.doi.org/10.1109/ACCESS.2020.3012435.
[40] A. Rahman, M.J. Islam, A. Montieri, M.K. Nasir, M.M. Reza, S.S. Band, A. Pescape, M. Hasan, M. Sookhak, A. Mosavi, SmartBlock-SDN: An optimized blockchain-SDN framework for resource management in IoT, IEEE Access 9 (2021) 28361–28376, http://dx.doi.org/10.1109/ACCESS.2021.3058244.
[41] M.H. Miraz, M. Ali, P.S. Excell, R. Picking, Internet of nano-things, things and everything: Future growth trends, Future Internet 10 (8) (2018) http://dx.doi.org/10.3390/fi10080068, URL https://www.mdpi.com/1999-5903/10/8/68.
[42] O.N. Fundation, Software-defined networking: The new norm for networks, 2012.
[43] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, D. Walker, P4: Programming protocol-independent packet processors, SIGCOMM Comput. Commun. Rev. 44 (3) (2014) 87–95, http://dx.doi.org/10.1145/2656877.2656890, URL http://dx.doi.org/10.1145/2656877.2656890.