2021 IEEE International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo) | 978-1-6654-2652-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/UkrMiCo52950.2021.9716691

Construction of the SDN Control Level Based on ONOS

Oleksandr Romanov
Institute of Telecommunication System
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine
a_i_romanov@ukr.net

Nadiia Korniienko
Institute of Telecommunication Systems
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine
nkornienko2000@ukr.net

Ivan Obod
dept. of Microprocessor Technologies and System
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
ivan.obod@nure.ua

Iryna Svyd
dept. of Microprocessor Technologies and System
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
iryna.svyd@nure.ua

***Abstract*—Possibilities of construction of logically centralized control plane in SDN networks on the basis of open network operating system ONOS are considered. The structure of the controller and its main functional blocks are considered, which provide the collection of information about the state of network elements, the solution of the main control tasks, the interaction of control systems built on different technological bases. The role and place of open network operating system in the controller structure are shown, the description of ONOS multilevel architecture in the form of a set of functional modules is given, the purpose and functions of ONOS subsystems are analyzed, protocols and interfaces that allow to present SDN network as a model are described. The peculiarity of the model is that the managed network can be represented as a set of virtual network functions. Therefore, the control process becomes independent of which vendor's equipment was used to build the network, as well as whether the network is built on real physical elements or virtual ones.**

Keywords—Open Network Operating System, ONOS, Controller, SDN, Control Plane SDN, Data Plane SDN.

I. Introduction

One of the most promising areas of development of modern telecommunications is SDN technology. Issues of building networks based on this technology are in the center of attention of representatives of research organizations, universities and mobile operators.

The architecture of the software-configured network consists of three levels: applications, management and infrastructure, interconnected through open API-interfaces (Fig. 1). The application layer contains functional blocks with a set of software that can solve individual management tasks, provide modern services to users, process statistics on the state of network elements, provide automated change of protocols when configuring the network, virtualize network functions, load balancing and more.

The management level is based on the controller, which is a centralized network management body. This level provides policy and network traffic management. The infrastructure layer contains both physical and virtual network elements that have only executive functions. All management tasks are solved by the controller, which brings these decisions to the level of infrastructure.

Today, research is underway and proposals are being developed for the practical implementation of this technology in a number of projects. These projects include the creation of next-generation broadband access infrastructure, the construction of a service platform, a platform for the construction of SD-RAN and Core RAN networks of mobile operators 5G / 6G. Representatives of the Open Networking Foundation (ONF) consortium make the greatest contribution to the development of this area. ONF is at the forefront of all SDN construction projects and is involved in accelerating the processes of their practical implementation [1].

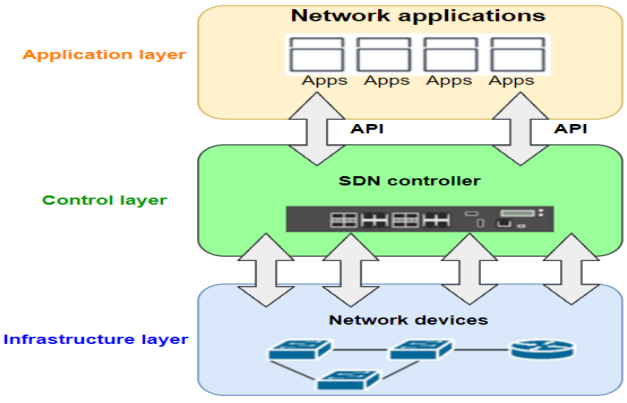

Fig. 1. General SDN architecture.

A number of documents describing the principles of construction and operation of SDN networks have been published. In works [3, 4, 5, 6] the general requirements, system approaches and the generalized architecture of SDN networks are considered. In works [2, 7, 8, 11] the tasks and features of the protocols used in solving various tasks in SDN networks are considered. In works [9, 10, 12] the features of construction of SDN network elements and the order of their interaction in the process of servicing information flows are described. In works [13, 14, 15, 16, 17] the principles of construction of the optical transport

network are considered and recommendations for ensuring the safety of their operation are given. The most complete and systematic material is presented in works [18, 19, 20, 21]. Articles [22, 23, 24, 25, 26] explore various aspects of information flow maintenance and focus on the need to meet network security requirements.

The most difficult tasks in SDN networks are the level of management. The solution of these problems is entrusted to the controller, the main element of which is the network operating system. Consider the features of building an open network operating system (ONOS), the functional composition of its elements, protocols and interfaces that allow you to represent the SDN network as a model.

II. Functional Structure of SDN Controller

The controller provides traffic maintenance in accordance with the policy set by the network operator. By removing the control plane from the network equipment, the controller implements a centralized management system, simplifies automatic network management and provides integration and administration of business applications. [27, 28, 29].

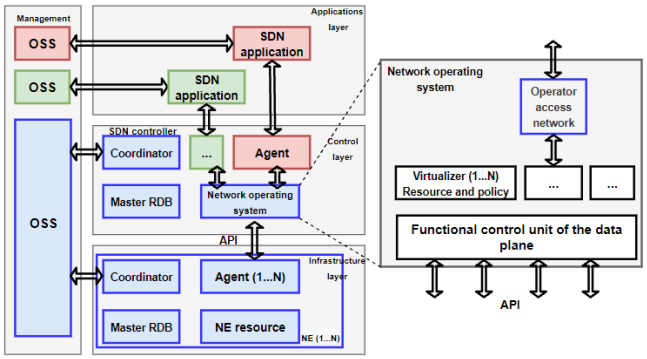

Fig. 2. Functional resources of the controller.

The SDN architecture (Figure 2) does not define the internal design or implementation of the SDN controller. It may be:

- one monolithic process;
- confederation of identical processes organized to distribute the load or protect each other from failure;
- a set of individual functional components that work according to a certain set of rules;
- the SDN controller can use external services to perform some functions, for example, to calculate the path.

Any combination of these alternatives is allowed: the SDN controller is considered a black box that performs certain functions. Controller components can be hosted on arbitrary computing platforms, including computing resources in data centers.

Just as OSS manages resources and states, the controller is subject to the same coordination requirement with any SDN controllers involved for sharing. Several components of a manager or controller can share network resources, but to meet SDN requirements, they must be:

- set up to manage extraordinary sets of resources;
- synchronized with each other so that there are never contradictions or conflicting commands.

The functional plane control unit of the data plane effectively has the resources available to it and is managed according to the instructions of the OSS/coordinator or virtualizer. Resources are recorded as an instance of the information model. The model is accessed through an agent at the subordinate level. Because the scope of the SDN controller may cover multiple network elements or even multiple virtual networks, the data plane control unit must ensure coordinated operation. This function is usually called orchestration.

The Resource Database (RDB) simulates the current instance of the information model and the required capabilities.

The coordinator is a functional component that acts on behalf of the human operator. It provides information on network service policies. It transmits control information for network elements: data models, control planes and application planes. Therefore, the functional blocks of the coordinator are located everywhere.

In the SDN architecture, virtualization is the allocation of virtual abstract resources for specific applications. The SDN controller offers application services in the form of an instance of an information model that abstractly describes the available resources and the policy of their use.

A virtualizer is a functional object that stores a copy of the information model of resources and the policy of their use.

The virtualizer is created by the OSS/coordinator for each client application. The OSS/coordinator allocates resources and determines the policies that the virtualizer should use in the process of providing services with applications through the API. Next, the virtualizer creates an agent in which it writes its copy of the information model of available resources and the policy of its use to provide service to a particular program.

Any protocol must end with a functional object. The controller-agent model is suitable for the relationship between the controlled and the controlling object. The controlled object is assigned by the agent, a functional component that represents the resources and capabilities of the client in the server environment. The agent in the N-level SDN controller represents the resources and actions available with the N + 1 application.

III. Architecture of the Open Network Operating System

Let's consider the principles of building the architecture of the open network operating system ONOS and its features. It should be noted that ONOS is the main functional block of the SDN controller. Therefore, in the scientific and technical literature, ONOS is often identified with the SDN network controller. It should be borne in mind that this is not entirely true. The controller consists of a number of functional blocks, one of which is ONOS. However, due to the fact that ONOS is responsible for solving all the intellectual tasks of network management, its architecture uses the approaches, descriptions and levels that were used both when describing the SDN architecture and when describing the controller of this network.

It should be noted that the ONOS development is a separate open source ONF project released under the Apache 2.0 license [30]. In addition, the ONOS system description is publicly available [31].

ONF believes the ONOS platform is being designed to meet the needs of carriers to create carrier-grade solutions that provide the flexibility to create and deploy new dynamic network services with simplified APIs. ONOS must provide support for both configuration and real-time network management. This approach will eliminate the need to run routing and switching control protocols within the operator's network infrastructure.

One possible way to further improve management efficiency is to move ONOS to the cloud. This will allow end users to create new network applications without having to change the data plane.

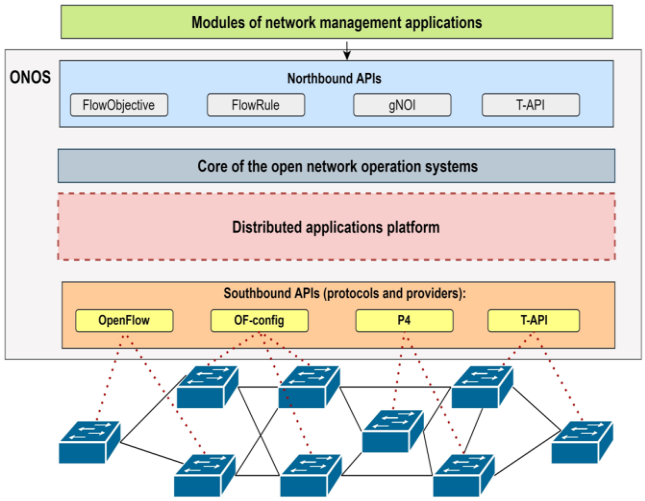

Fig. 3. ONOS architecture.

A possible version of the ONOS architecture is shown in Fig. 3. As shown in Figure 3, the ONOS architecture is modular in design. This approach allows you to ensure optimal resource use, since for each specific deployment, the optimal number of the required subset of modules will be determined. The ONOS architecture includes:

- The core of the open operating system, which is responsible for solving all the intellectual tasks of network management.
- A distributed application platform, which is a set of applications for solving network management problems. This platform is built as an extensible modular system. Moreover, each modular functional block solves a specific control problem. As the list of control problems to be solved increases, the number of modules in the distributed application platform will increase.
- A set of open network interfaces. At the same time, just as in the SDN architecture, the use of two types of interfaces is proposed: open southbound interface (SBI), which is a set of modules for interacting with the data layer;open northbound interface (NBI), which is a set of modules for interacting with functional blocks of the application level.

It should be noted that the southbound interface includes a set of protocols and interfaces such as OpenFlow, OF-CONFIG, P4, T-API and others. The tasks that these elements of the system solve and their importance are determined by the conditions for the functioning of the network. For example, consider the operation of the OF-CONFIG and OpenFlow protocols.

OF-CONFIG (OpenFlow Management and Configuration Protocol) is a protocol for configuring and managing the operational context of OpenFlow switches.

And the OpenFlow protocol is a protocol for controlling the processing of data transmitted over a data transmission network by routers and switches, which implements software-defined network technology.

The OpenFlow protocol assumes that an OpenFlow switch (for example, an Ethernet switch that supports OpenFlow) has been configured with various artifacts, such as the IP addresses of OpenFlow controllers and other network elements. That is, the switch is pre-configured.

OF-CONFIG protocol is designed for remote configuration of OpenFlow switches. It runs in a slower mode compared to the OpenFlow protocol.

For example, the OF-CONFIG protocol solves the problem of creating routing matrices, which will later be solved by the OpenFlow protocol in real time when processing incoming packets. Another example of the OF-CONFIG protocol can be enabling / disabling a port, which is also not related to real-time packet processing.

OF-CONFIG introduces the OpenFlow switch as an abstraction called the OpenFlow logical switch (virtual switch). The OF - CONFIG protocol allows the configuration of the OpenFlow Logical Switch parameters so that the OpenFlow Controller can communicate and control the OpenFlow Logical Switch via the OpenFlow Protocol.

OF-CONFIG allows you to dynamically associate resources with specific OpenFlow logical switches. This protocol can make several virtual switches from one physical switch and assign a certain resource to each. ONOS uses these protocols to interact with the data plane in order to implement network management tasks.

Recently, P4, T-API are starting to be used. They are increasingly used to build network models for solving control problems. At the same time, P4 shows greater efficiency than OpenFlow.

It is advisable that the ONOS set of open network interfaces and protocols be universal for all network segments and elements. This would make it possible to build homogeneous, scalable, universal systems with a high level of interaction of all elements.

This ONOS architecture will allow:

- Simplify the process of reinstalling and upgrading software on network elements.
- Reduce the cost of introducing new technologies and services throughout the operator's network.
- A step-by-step increase in the number of control problems to be solved by adding new functional blocks as part of the distributed application platform.
- Create a horizontally scalable system that provides a high level of fault tolerance, which is very important

to ensure that the specified requirements for the reliability of the functioning of centralized control systems are met.

Based on the above, it is possible to formulate the requirements for the ONOS architecture. First, the horizontal plane of the NBI must be large enough. This is because any access to the underlying hardware will be done through ONOS. Therefore, the aggregate of all northbound APIs should be sufficient for configuring, maintaining, and managing the network.

In addition, there should be a multilevel system of applications and services running on ONOS. Depending on the priority of the task solved by the application and the frequency of its use, a level should be determined that determines the efficiency of the task. For example, applications that define the main control function block in the event of a failure of network elements must have a direct connection with ONOS. And applications that provide the installation of new software, certificates, configuration parameters and in their work use the OF-CONFIG protocol can be located at a level with a high latency.

Another requirement for ONOS is to provide simultaneous two-way information exchange via NBI and SBI. As can be seen from Fig. 3, applications use ONOS through NBI to manage the network, and through SBI, southbound modules transmit information about the state of the underlying network to the ONOS core.

The interaction between the ONOS core and network devices is provided by a set of protocols and interfaces, such as OpenFlow, OF-CONFIG, P4, T-API, which take on the granularity of interaction with devices, thereby isolating the ONOS core and applications running on top of it from the details of diversity network devices.

The ONOS core consists of a number of subsystems, each of which is responsible for a specific aspect of the functioning of the network. Each subsystem maintains its own service abstraction, which is responsible for propagating network state parameters across the cluster.

ONOS services are built using distributed tables, which are implemented using a distributed key / value store. ONOS uses Atomix for storage. It is a Java based system that includes: distributed data structure; description of the algorithm for direct message exchange between elements; description of coordination of interaction, including blocking of unwanted functions and election of a leader; group membership management.

An important feature of Atomix is the coordination of all ONOS instances. Two tasks are solved here:

- The number of ONOS instances running at any given time depends on the workload and the number of replications required to ensure availability in the event of a failure. The Atomix group membership primitive is used to define the set of ONOS instances available. This allows you to track healthy ONOS instances and failed ONOS instances.
- The main task of each ONOS instance is to monitor and maintain a subset of physical switches in the network where it is elected as a leader. The definition of a leader is carried out using the Atomix function block. All ONOS instances can monitor the status of the switches. However, only the leader can manage the switches. If an ONOS instance fails, Atomix ensures that a new leader is selected for the switches. The same approach is used when connecting a new switch to the network.

IV. ONOS Subsystems and Services

Although ONOS relies heavily on standard protocols and models, such as OpenFlow, NETCONF, OpenConfig, its system architecture is not directly tied to them. ONOS modular operating system, it consists of separate subsystems (Fig. 4).

The ONOS application management subsystem is responsible for distributing application artifacts across the cluster. It ensures that all nodes work with the same software. The basic ONOS distribution contains more than 175 applications that fall into numerous categories, such as traffic management programs, device drivers, utilities, monitoring programs, ready-made YANG models.

In order to interact with the outside world, ONOS has a graphical interface and a number of external adapters, such as the REST API, CLI and an extensible dynamic web interface. Open source gRPC interface. Uses HTTP / 2 for transport. It provides features such as authentication, bidirectional streaming, flow control, cancellation and timeouts.

ONOS applications act as core extensions, they can be protocol libraries, drivers or pre-compiled models.

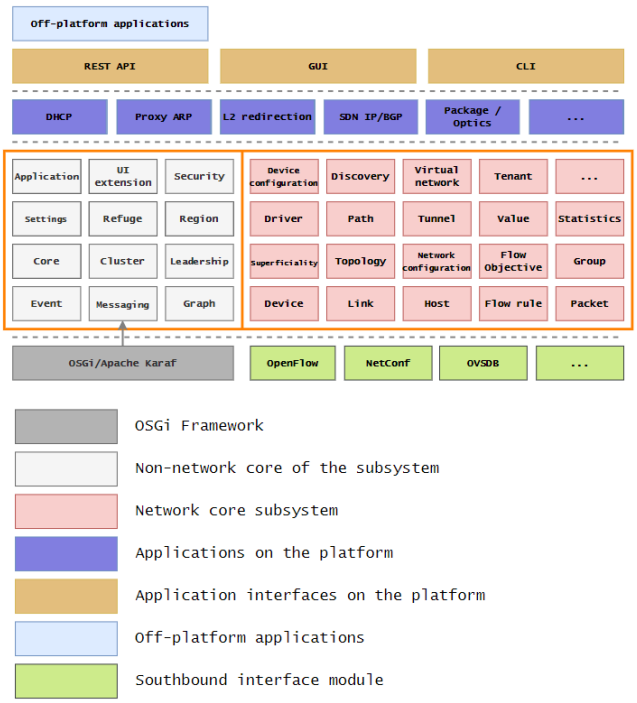

Fig. 4. ONOS subsystems.

The ONOS core consists of two parts: network and non-network. The network part supports the following functions: collection of statistics, topology analysis, device configuration, creation of virtual networks, web groups, in general, support for services to work on the Internet. The non-network part performs the functions necessary to control the physical part of the SDN controller, namely messaging between controllers, device memory management, setting the

operating system locally, support for a graphical interface for simplified system management.

Regarding the purpose of some ONOS services, the most typical of them are:

Host: records the end systems connected to the network. This uses ARP, NDP, or DHCP packet interception.

Device: records information about infrastructure devices such as switches, routers, ROADM, including a description of their ports.

Links: Writes link attributes that connect device/ infrastructure port pairs, using LLDP packet interception.

Topology: represents the network as a whole using a graph. The service is based on the Device and Link services. Provided in the form of an agreed graph.

Wizard: Defines the leader using Atomix. Selects which ONOS instance in the cluster should be the master for each infrastructure device. In the event of a failure of the ONOS instance, it guarantees the timeliness of selecting a new wizard for all devices left without it.

Cluster: Controls the configuration of the ONOS cluster. It provides information about all peer ONOS nodes in Atomix. Atomix nodes perform clustering on ONOS nodes with master definition. In this case, the ONOS nodes themselves are actually simple clients that are used to scale the control logic and I/O for network devices.

Network configuration: describes meta information about networks, such as network devices, their ports, hosts, and links. Provides information to external users about how the network is served by the ONOS kernel and its applications.

Component Config: manages configuration settings for various software components in the ONOS kernel and applications. Operates the following parameters: rules of external flow processing; device address or DHCP server; survey frequency, etc.

Allows you to solve software configuration problems. Installed by the operator according to deployment needs.

Packet: Allows basic services and applications to intercept incoming packets, analyze them, and send them back to the network. This uses well-known methods for detecting nodes and channels, such as ARP, DHCP, LLDP.

The above services are used by many applications because they provide information about network devices and their topologies. However, there are many more services. Including those that allow applications to program network behavior using different constructs and different levels of abstraction. These include:

Route: Defines a prefix for matching the next step. Installed either by the control application or manually configured by the operator.

Mcast: determines the IP address of the group, the location of the source and receiver. Installed by the control application or manually by the operator.

Group: combines ports or actions in a device. Stream records can point to a specific group. This allows: use complex means of forwarding, such as load balancing between ports in a group; switching between ports in the group; multicast transmission to all ports specified in the group.

The group can also be used to aggregate the joint actions of different threads. In this case, you only need to change one record in order to change the maintenance order in the whole group.

Counter: Expresses the speed limit to ensure the quality of service for the selected network traffic processed by the device.

Flow rule: Provides a device-oriented match/action pair to program the forwarding behavior at the device data level. It requires that flow rule entries be compiled according to the conveyor structure and device capabilities.

Stream target: Used to program the device's forwarding behavior in a conveyor-independent manner.

Intention: Provides a topology-independent way to set network processing rules. Specifications may indicate various restrictions for the use of the through path; type of traffic; source and target hosts; input and output ports; request types and connection capabilities.

The service provides connection support in appropriate ways. And then continuously monitors the network, changing paths over time to ensure compliance with the objectives proposed by the intention, in a changing network situation.

Each service has its own distributed database and messaging capabilities. Some programs are free to expand this set with their own services [32].

V. Conclusion

1. ONOS is a user-friendly open source platform that allows different teams of developers to jointly participate in projects to upgrade and improve the management system.

2. Using ONOS allows you to build a logical centralized control plane in SDN networks.

3.The existing set of functional modules, services and interfaces in the ONOS, allows you to perform network management tasks.

4. For further development of ONOS it is necessary to develop mathematical models and methods of optimal solution of control problems in different operating conditions, which in the future will become software modules at the application level.

References

[1] Open Network Foundation. Accelerating the Adoption of SDN & NFV, 2021.

[2] "Open Network Operating System (ONOS) SDN Controller for SDN/NFV Solutions", *Open Networking Foundation*, 2021. [Online]. Available: https://opennetworking.org/onos.

[3] "What is SDN controller (software-defined networking controller)? - Definition from WhatIs.com", *SearchNetworking*, 2021. [Online]. Available: https://searchnetworking.techtarget.com/definition/SDN-controller-software-defined-networking-controller.

[4] "ONOS - Wikipedia", *En.wikipedia.org*, 2021. [Online]. Available: https://en.wikipedia.org/wiki/ONOS.

[5] "Apache License, Version 2.0 | Open Source Initiative", *Opensource.org*, 2021. [Online]. Available: https://opensource.org/licenses/Apache-2.0.

[6] ONF TR-525 SDN Interoperability Event Technical Issues Report AppFest 2015.

[7] K. Pentikousis, IETF RFC 7426. Request for Comments: 7426. ISSN: 2070-1721 EICT.

[8] O. I. Romanov, M. V. Oryschuk and Y. S. Hordashnyk, "Computing of influence of stimulated Raman scattering in DWDM telecommunication systems," *2016 International Conference Radio Electronics & Info Communications (UkrMiCo)*, 2016, pp. 1-4, doi: 10.1109/UkrMiCo.2016.7739622.

[9] K. Pentikousis, *ONOS. Security and Performance*. Analysis: Report No. 1. September 19, 2017.

[10] O. Romanov and V. Mankivskyi, "Optimal Traffic Distribution Based on the Sectoral Model of Loading Network Elements," *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T)*, 2019, pp. 683-688, doi: 10.1109/PICST47496.2019.9061296.

[11] O. Lemeshko and O. Yeremenko, "Linear optimization model of MPLS Traffic Engineering Fast ReRoute for link, node, and bandwidth protection," *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, 2018, pp. 1009-1013, doi: 10.1109/TCSET.2018.8336365.

[12] O. Romanov, M. Nesterenko, L. Veres, R. Kamarali and I. Saychenko, "Methods for Calculating the Performance Indicators of IP Multimedia Subsystem (IMS)", *Advances in Information and Communication Technology and Systems*, pp. 229-256, 2020. doi: 10.1007/978-3-030-58359-0_13.

[13] O. Lemeshko, J. Papan, O. Yeremenko, M. Yevdokymenko and P. Segec, "Research and Development of Delay-Sensitive Routing Tensor Model in IoT Core Networks", *Sensors*, vol. 21, no. 11, p. 3934, 2021. doi: 10.3390/s21113934.

[14] C. C. O'Connor, T. Vachuska, and B. Davie, “Software-Defined Networks: A Systems Approach”, 2021, p. 152.

[15] K. Phemius, M. Bouet and J. Leguay, "DISCO: Distributed multi-domain SDN controllers," *2014 IEEE Network Operations and Management Symposium (NOMS)*, 2014, pp. 1-4, doi: 10.1109/NOMS.2014.6838330.

[16] J. Lam, S. Lee, H. Lee and Y. Oktian, "Securing SDN Southbound and Data Plane Communication with IBC", *Mobile Information Systems*, vol. 2016, pp. 1-12, 2016. doi: 10.1155/2016/1708970.

[17] O. I. Romanov, D. M. Fediushyna and T. T. Dong, "Model And Method Of Li-Fi Network Calculation With Multipath Light Signals," *2018 International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo)*, 2018, pp. 1-4, doi: 10.1109/UkrMiCo43733.2018.9047550.

[18] I. Obod, I. Svyd, O. Maltsev, G. Zavolodko, D. Pavlova and G. Maistrenko, "Fusion the Coordinate Data of Airborne Objects in the Networks of Surveillance Radar Observation Systems", *Data-Centric Business and Applications*, pp. 731-746, 2020. doi: 10.1007/978-3-030-43070-2_31.

[19] O. Romanov, E. Siemens, M. Nesterenko and V. Mankivskyi, "Mathematical description of control problems in SDN networks", *International Conference on Applied Innovations in IT (ICAIIT)*, pp. 33-39, 2021. Available: 10.25673/36582.

[20] I. Svyd, I. Obod, O. Maltsev, T. Tkachova and G. Zavolodko, "Optimal Request Signals Detection in Cooperative Surveillance Systems," *2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON)*, 2019, pp. 1-5, doi: 10.1109/UKRCON.2019.8879840.

[21] I. Obod, I. Svyd, O. Maltsev, G. Maistrenko, O. Zubkov and G. Zavolodko, "Bandwidth Assessment of Cooperative Surveillance Systems," *2019 3rd International Conference on Advanced Information and Communications Technologies (AICT)*, 2019, pp. 1-6, doi: 10.1109/AIACT.2019.8847742.

[22] D. Sanvito, D. Moro, M. Gullì, I. Filippini, A. Capone and A. Campanella, "ONOS Intent Monitor and Reroute service: enabling plug&play routing logic," *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, 2018, pp. 272-276, doi: 10.1109/NETSOFT.2018.8460064.

[23] D. Comer and A. Rastegarnia, "Externalization of Packet Processing in Software Defined Networking," in *IEEE Networking Letters*, vol. 1, no. 3, pp. 124-127, Sept. 2019, doi: 10.1109/LNET.2019.2918155.

[24] O. Romanov, M. Nesterenko and V. Mankivskyi, "The Method of Redistributing Traffic in Mobile Network", *Data-Centric Business and Applications*, pp. 159-182, 2021. doi: 10.1007/978-3-030-71892-3_7.

[25] "GitHub - OpenNetworkingFoundation/TAPI: ONF Transport API Repository (TAPI)", *GitHub*, 2021. [Online]. Available: https://github.com/OpenNetworkingFoundation/tapi.

[26] I. Svyd, I. Obod, O. Maltsev, T. Tkachova and G. Zavolodko, "Improving Noise Immunity in Identification Friend or Foe Systems," *2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON)*, 2019, pp. 73-77, doi: 10.1109/UKRCON.2019.8879812.

[27] "TAPI Overview - Open Transport Configuration & Control - Confluence", *Wiki.opennetworking.org*, 2021. [Online]. Available: https://wiki.opennetworking.org/display/OTCC/TAPI+Overview.

[28] P. Littlewood, F. Masood, E. Follis. *Optical transport network*. Hannover: Ciena, 2014.

[29] "TAPI v2.1.3 Reference Implementation Agreement. TR-547. Version 1.0. July 2020", *Opennetworking.org*, 2018. [Online]. Available: https://opennetworking.org/wp-content/uploads/2020/08/TR-547-TAPI-v2.1.3-Reference-Implementation-Agreement-1.pdf.

[30] "Open Network Operating System (ONOS) SDN Controller for SDN/NFV Solutions", *Open Networking Foundation*, 2021. [Online]. Available: https://opennetworking.org/onos/.

[31] "Software-Defined Networks: A Systems Approach — Software-Defined Networks: A Systems Approach Version 2.1-dev documentation", *Sdn.systemsapproach.org*, 2021. [Online]. Available: https://sdn.systemsapproach.org/index.html.

[32] "Chapter 6: Network OS — Software-Defined Networks: A Systems Approach Version 2.1-dev documentation", *Sdn.systemsapproach.org*, 2021. [Online]. Available: https://sdn.systemsapproach.org/onos.html.