Network Device Emulation Method for Implementation of Network Functions

Olena Starkova
Department of network and Internet technologies,
Taras Shevchenko National University of Kyiv,
Kyiv, Ukraine
elesta.tcs@gmail.com

Kostiantyn Herasymenko
Department of network and Internet technologies,
Taras Shevchenko National University of Kyiv,
Kyiv, Ukraine
c.herasymenko@gmail.com

Kyryl Nikolchev
Department of network and Internet technologies,
Taras Shevchenko National University of Kyiv,
Kyiv, Ukraine
nikolchev.kyryl@gmail.com

Yurii Ahafonov
Department of network and Internet technologies,
Taras Shevchenko National University of Kyiv,
Kyiv, Ukraine
storminside4766@gmail.com

Viktoriia Ivannikova
Air Transportation Management Department,
National Aviation University,
Kyiv, Ukraine
victoriia.ivannikova@gmail.com

2021 IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT) | 978-1-6654-3847-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/ATIT54053.2021.9678587

***Abstract*—This work is devoted to creating a network device emulation method for testing self-created network protocols (particularly routing protocols). It can be run on single device without virtualization or containerizing. The method was developed using the concept of software-defined networks (SDN), as well as the Network functions virtualization (NFV) architectural framework created by ETSI. When developing the method, the means of mathematical and simulation modeling were used.**

Keywords— network device, dynamic routing protocol, test environment, Python, SDN, network functions virtualization.

I Smart internet devices

In the days when network technology was still in its infancy, every recipient on the network received all transmissions, and one real recipient determined that the data was meant for him, while others simply discarded the data.

To date, it is clear that such a scheme in today's environment of mistrust is dangerous and ineffective. You cannot send your two-way authentication data to all devices on the network.

Technically, the above scheme was in the era of hubs, which were not intelligent devices that directed traffic from one interface to all others.

As network technology became more sophisticated, smart devices were able to identify the correct recipient and route traffic to their destination accordingly. This refers to switches and routers, which are now usually combined into a single device.

II Network Devices Functions

There are a lot of the functions that perform network devices. For example, the router performs many tasks [1-3]:

- Connecting local networks to the global network.
- Network segmentation into separate broadcast domains, which increases the security, performance and controllability of such networks.
- Finding the best route for packet delivery over the network by composing routing tables. Dynamic routing protocols of different types are used to find the best route to the destination according to different parameters.
- To create easy to access network infrastructure, routers can serve as different servers, such as a DNS or DHCP server.
- Routers can establish encrypted tunnels for secure data transmission to remote network.
- Firewall and Intrusion Prevention System (IPS).

III SDN As A Solution For Problems of Modern Networks

At the moment, there are two trends in the development of networks:

- optimization of existing technologies, protocols, mechanisms, algorithms in accordance with the requirements that are presented to modern networks;
- automation of management of network devices;
- virtualization of network functions.

One of the most promising methods that can help to solve listed problems is the Software Defined Networking (SDN) model [4], which provides for the separation of traffic transmission functions and control functions, including control of both the traffic itself and the devices that transmit it. According to the SDN concept, all control logic is located in controllers that are able to monitor the operation of the entire network using special protocols (for example, OpenFlow), which operate on the concept of flows and can perform various actions with them (allow, deny, redirect, edit fields in packages, etc.). The advantages of a software-defined network are centralized management, simplification of network maintenance and modernization.

Network functions virtualization (NFV) is an architectural framework created by the European Telecommunications Standards Institute (ETSI) that defines standards to decouple network functions from proprietary hardware-based appliances and have them run in software on standard x86 servers. Some of the benefits of NFV are similar to the benefits of server virtualization and cloud environments [5]:

- Reduced capital expenditure (capex) and operational expenditure (opex) through reduced equipment costs and efficiencies in space, power, and cooling.
- Faster time to market (TTM) because VMs and containers are easier to deploy than hardware.
- Improved return on investment (ROI) from new services.
- Ability to scale up/out and down/in capacity on demand (elasticity).
- Openness to the virtual appliance market and pure software networking vendors.
- Opportunities to test and deploy new innovative services virtually and with lower risk.

NFV Architecture Framework, that was developed by ETSI showed on Fig.1.

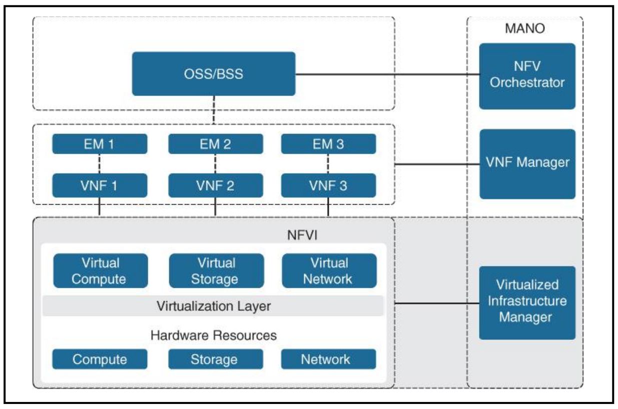


Fig. 1. ETSI NFV Architectural Framework

IV Concept of Test Environment

There is no simple test environment for creating new dynamic routing protocols [5-9]. There are several projects for modeling a network on a regular computer, but they do not involve significant protocol changes and load the system with things that are not needed in routing testing, such as the second layer. Although mostly already existing projects use containerizing or must be deployed on different physical devices or virtual systems using virtualization.

Therefore, it was decided to create a simple system exclusively for testing dynamic routing protocols. The system works on one computer and allows you to quickly and easily create and modify various routing protocols with their subsequent testing, as well as in the future to implement the developed routing methods that are more efficient than existing ones.

V Why using Python

Python was the language of choice due to its high prevalence, especially among network professionals. Although using Python as a development language will allow the application to be as cross-platform as possible [10-20].

VI How second network layer works in application

Instead of emulating the second network layer which is not needed for developing and testing dynamic routing protocols our utility uses sockets. When using sockets there are no MAC addresses and ARP packets but ports. So to emulate different interfaces on one computer we need a network port for every interface. Although we need to connect interfaces to send data from one to another. So we use something like broadcast ports. In one simulation interfaces with one broadcast port will hear each other's packets. So we can use broadcast connection to send something like ARP packets and connect interfaces in something like socket connection.There is two ways connecting interfaces:

- Manual connecting

1. ***connect_to_router*** function must be executed at first routers interface that want to be connected

2. ***accept_connection*** function must be executed at second routers interface that want to be connected

- Automatic connection

1. There is must be only two interfaces in broadcast connection (only 2 interfaces must have same broadcast port)
2. From one interface to broadcast must be send message ***"Hello, lets connect"***
Example of router connections in simulation will be displayed in the next image.

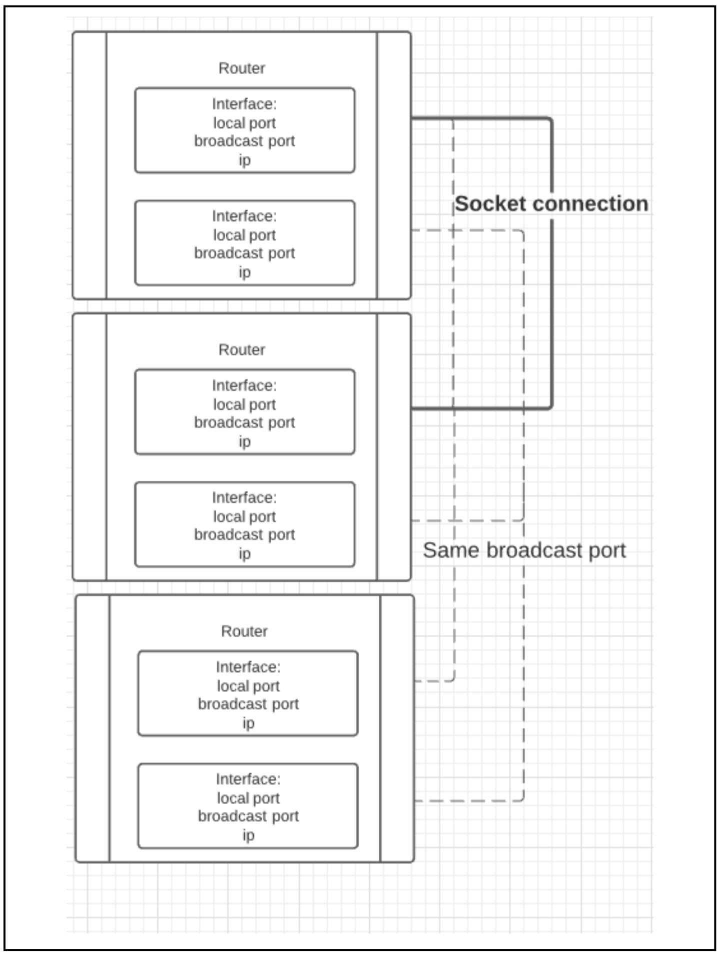

Router
Interface: local port broadcast port ip
Interface: local port broadcast port ip
Router
Interface: local port broadcast port ip
Interface: local port broadcast port ip
Router
Interface: local port broadcast port ip
Interface: local port broadcast port ip
Socket connection
Same broadcast port

Fig. 2. Example of router connections in simulation.

VII How routing works in application

After creating interfaces and their connection we need to create basic static routing logic. Interfaces are always listening for new packets and depending on the destination IP-address field in the packet decide what to do with it. If the received packet destination is appointed to one of routers interfaces it will just say that receive packet. If the router knows that destination IP-address is one of IP-addresses his interfaces connected directly to. It will just send this packet from this interface [21-25]. If the packet destination IP-address is in one of the networks that is in routers routing table router will send packet to interface appointed to this network in routing table. Although if the packet is not in some network router will send it to the next hop interface (0.0.0.0 network in routing table). And finally if the router can't find any route it will drop the packet.

Next will be displayed routers working schema.

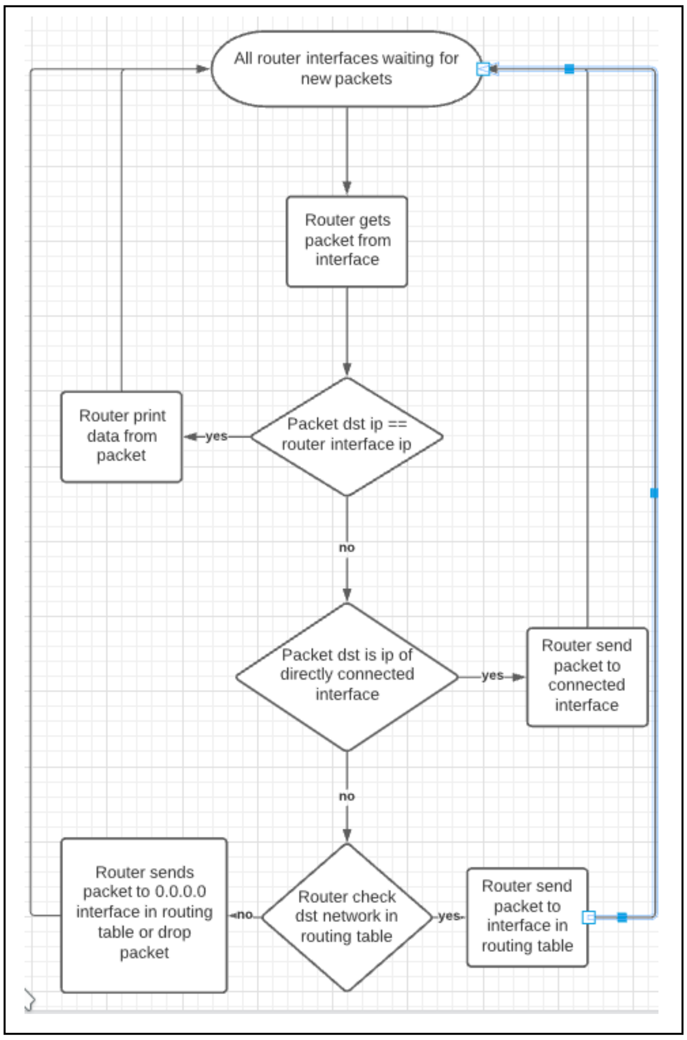
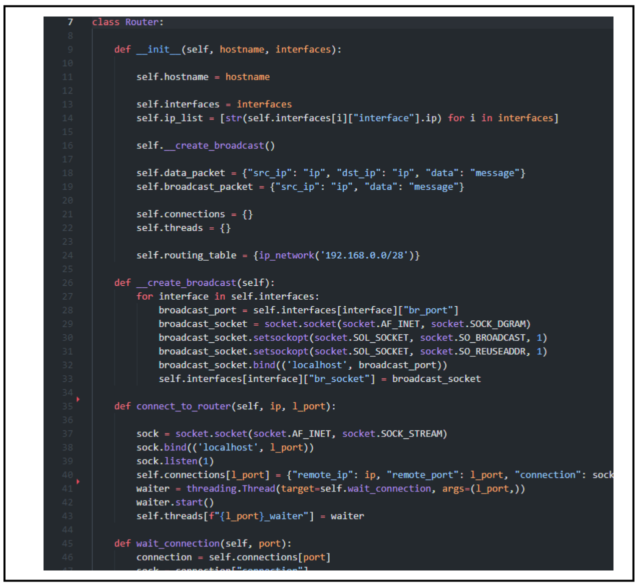

All router interfaces waiting for new packets
Router gets packet from interface
Packet dst ip == router interface ip
yes
Router print data from packet
no
Packet dst is ip of directly connected interface
yes
Router send packet to connected interface
no
Router check dst network in routing table
yes
Router send packet to interface in routing table
no
Router sends packet to 0.0.0.0 interface in routing table or drop packet

Fig. 3. Routers working schema

VIII How dynamic routing protocols are implemented in application.

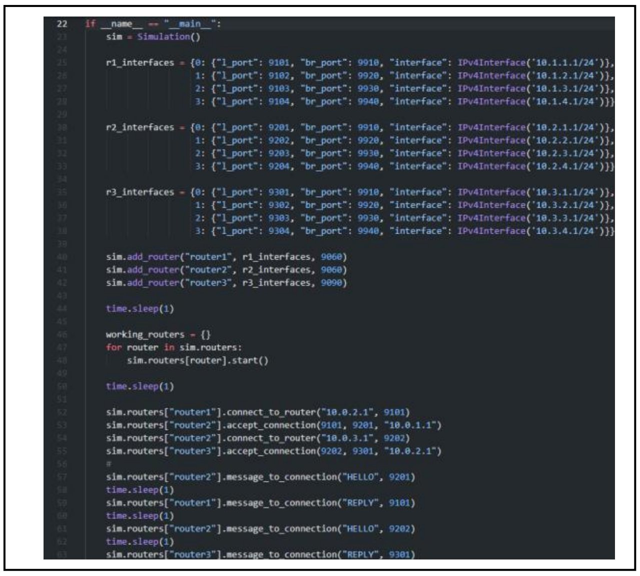
To create your own dynamic routing protocol, you need to create a new class that inherits the ***RoutingProtocol*** class. You can also copy and change the already created example class. Any additional data arrays can be created inside, for example, information about delays to each destination node. The dynamic routing protocol changes the main routing table in the router with new information. To find new routes, you can use not only messages on the router interfaces, but also broadcasts. In the dynamic routing protocol there is a function with some trigger. Trigger can be a time passed from the last main function run or function call etc. Trigger runs the main dynamic protocols function which defines the best routes and changes routers routing table [26-33].

We have made a simple router simulation using Python and sockets, simulating a very simple network with a single server and multiple clients [9-10]. The server shall be sending some data to the router, and the router will have functionality to decide which client to deliver the data to (Fig. 4-5).



```python
class Router:

    def __init__(self, hostname, interfaces):

        self.hostname = hostname

        self.interfaces = interfaces
        self.ip_list = [str(self.interfaces[i]["interface"].ip) for i in interfaces]

        self.__create_broadcast()

        self.data_packet = {"src_ip": "ip", "dst_ip": "ip", "data": "message"}
        self.broadcast_packet = {"src_ip": "ip", "data": "message"}

        self.connections = {}
        self.threads = {}

        self.routing_table = {ip_network('192.168.0.0/28')}

    def __create_broadcast(self):
        for interface in self.interfaces:
            broadcast_port = self.interfaces[interface]["br_port"]
            broadcast_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
            broadcast_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
            broadcast_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
            broadcast_socket.bind(('localhost', broadcast_port))
            self.interfaces[interface]["br_socket"] = broadcast_socket

    def connect_to_router(self, ip, l_port):

        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.bind(('localhost', l_port))
        sock.listen(1)
        self.connections[l_port] = {"remote_ip": ip, "remote_port": l_port, "connection": sock
        waiter = threading.Thread(target=self.wait_connection, args=(l_port,))
        waiter.start()
        self.threads[f"{l_port}_waiter"] = waiter

    def wait_connection(self, port):
        connection = self.connections[port]
```

Fig. 4. A simple router simulation using Python. Sockets



```python
if __name__ == "__main__":
    sim = Simulation()

    r1_interfaces = {0: {"l_port": 9101, "br_port": 9910, "interface": IPv4Interface('10.1.1.1/24')},
                     1: {"l_port": 9102, "br_port": 9920, "interface": IPv4Interface('10.1.2.1/24')},
                     2: {"l_port": 9103, "br_port": 9930, "interface": IPv4Interface('10.1.3.1/24')},
                     3: {"l_port": 9104, "br_port": 9940, "interface": IPv4Interface('10.1.4.1/24')}}

    r2_interfaces = {0: {"l_port": 9201, "br_port": 9910, "interface": IPv4Interface('10.2.1.1/24')},
                     1: {"l_port": 9202, "br_port": 9920, "interface": IPv4Interface('10.2.2.1/24')},
                     2: {"l_port": 9203, "br_port": 9930, "interface": IPv4Interface('10.2.3.1/24')},
                     3: {"l_port": 9204, "br_port": 9940, "interface": IPv4Interface('10.2.4.1/24')}}

    r3_interfaces = {0: {"l_port": 9301, "br_port": 9910, "interface": IPv4Interface('10.3.1.1/24')},
                     1: {"l_port": 9302, "br_port": 9920, "interface": IPv4Interface('10.3.2.1/24')},
                     2: {"l_port": 9303, "br_port": 9930, "interface": IPv4Interface('10.3.3.1/24')},
                     3: {"l_port": 9304, "br_port": 9940, "interface": IPv4Interface('10.3.4.1/24')}}

    sim.add_router("router1", r1_interfaces, 9060)
    sim.add_router("router2", r2_interfaces, 9060)
    sim.add_router("router3", r3_interfaces, 9090)

    time.sleep(1)

    working_routers = {}
    for router in sim.routers:
        sim.routers[router].start()

    time.sleep(1)

    sim.routers["router1"].connect_to_router("10.0.2.1", 9101)
    sim.routers["router2"].accept_connection(9101, 9201, "10.0.1.1")
    sim.routers["router2"].connect_to_router("10.0.3.1", 9202)
    sim.routers["router3"].accept_connection(9202, 9301, "10.0.2.1")
    #
    sim.routers["router2"].message_to_connection("HELLO", 9201)
    time.sleep(1)
    sim.routers["router1"].message_to_connection("REPLY", 9101)
    time.sleep(1)
    sim.routers["router2"].message_to_connection("HELLO", 9202)
    time.sleep(1)
    sim.routers["router3"].message_to_connection("REPLY", 9301)
```

Fig. 5. A simple router simulation using Python. Interfaces

IX Conclusions

Considering limitations and trends in modern networks the existing standards the best decision is to use NFV. Architectural Framework of NFV was developed by ETSI.

Emulation of network devices and their functions gives an opportunity to develop and modify our own standards, which gives us more flexibility while designing the corporate and global networks. In this work a method for developing new and modifying existing network functions was created and environment for testing self-created dynamic routing protocols was tested.

References

[1] Network routing: algorithms, protocols, and architectures / Medhi D., Ramasamy K. San Francisco: Kaufmann Publishers is an imprint of Elsevier, 2007.-824 p.

[2] ISO/IEC 10589 Information technology — Telecommunications and information exchange between systems — Intermediate System to Intermediate System intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473), Second edition 2002-11-15.

[3] RFC 2328 OSPF Version 2, April 1998.

[4] Software-defined networking (SDN): a survey. [Electronic resource] - Access mode: https://onlinelibrary.wiley.com/doi/epdf/10.1002/sec.1737

[5] Network Functions Virtualisation (NFV). [Electronic resource] - Access mode: https://www.etsi.org/technologies/nfv

[6] Use Containerlab to emulate open-source routers. [Electronic resource] - Access mode:
https://www.brianlinkletter.com/2021/05/use-containerlab-to-emulate-open-source-routers/

[7] Creating a simple router simulation using Python and sockets. [Electronic resource] - Access mode: https://medium.com/swlh/creating-a-simple-router-simulation-using-python-and-sockets-d6017b441c09

[8] Exploring the Functions of Routing. [Electronic resource] - Access mode: https://www.learncisco.net/courses/icnd-1/lan-connections/functions-of-routing.html

[9] Use Containerlab to emulate open-source routers. [Electronic resource] - Access mode: https://www.brianlinkletter.com/2021/05/use-containerlab-to-emulate-open-source-routers/

[10] Kravchenko, Y., Dakhno, N., Leshchenko, O., Tolstokorova, A. “Machine learning algorithms for predicting the results of COVID-19 coronavirus infection”, International conference Information Technology and Interactions, IT&I-2020, CEUR Workshop Proceedings, 2021, 2845, pp. 371–381.

[11] Kravchenko, Y., Afanasyeva, O., Tyshchenko, M., Mykus, S. “Intellectualisation of Decision Support Systems For Computer Networks: Production-Logical F-Inference”, International conference Information Technology and Interactions, IT&I-2020, CEUR Workshop Proceedings, 2021, 2845, pp. 117–126.

[12] Mashkov O. A. , Murasov R. K. , Kravchenko Y. V. , Dakhno N. B., Leschenko O. A., Trush O. V. Optimal forecast algorithm based on compatible linear filtration and extrapolation. Mathematical Modeling And Computing, 2021, 8(2), pp. 157–167.

[13] Mashkov, V.Task allocation among agents of restricted alliance. In Proc. of the 8th IASTED International Conference on Intelligent Systems and Control, ISC, pp. 13-18, 2005.

[14] Mashkov, V. Restricted alliance and coalitions formation. In Proc. of IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2004, pp. 329-332, 2004.

[15] Yudin, O., Suprun, O., Ziubina, R.,Buchyk, S., Frolov, O., Barannik, N. Efficiency Assessment of the Steganographic Coding Method with Indirect Integration of Critical Information: Proceeding of the International Conference on Advanced Trends in Information Theory (ATIT 2019), Kyiv, Ukraine, pp.36-40.

[16] Barannik, N., Yudin, O., Babenko, M.,.Manakov, V., Yroshenko, V. The method of embedding information in the noise domain based on the modification of elements in the pseudo-spatial domain: Proceeding of the International Conference on Advanced Trends in Information Theory (ATIT 2020), Kyiv, Ukraine, 2020. IEEE Catalog Number: ISBN 978-1-7281-9799-9/20/$31.00©2020 IEEE, pp.77-82.

[17] O. Yudin, O. Suprun S. Buchyk, R. Ziubina, I. Bondarenko Devising a Method of Protection Against Zero-Day Attacks Based on an Analytical Model of Changing the State of the Network Sandbox: Eastern-European Journal of Enterprise Technologies, 2021, 1(9(109)), pp. 50–57.

[18] Kovbasiuk S.V., Rakushev M.Yu., Permiakov O. Yu., Lavrinchuk O. V. Outer Space Monitoring System: Purpose, Tasks, Structure and approaches to trajectory processing. 2019 IEEE International Conference on Advanced Trends in Information Theory, ATIT 2019 Proceedings, pp. 154-160.

[19] Rakushev M.Yu. Numerical method for integrating solution of stochastic differential equations based on differential transformations. Journal of Automation and Information Sciences. Begell House Inc. Vol. 45, Issue 11, 2013. pp.51-62.

[20] Rakushev M.Yu. Numerical method of integrating the variational equations for Cauchy problem based on differential transformations. Journal of Automation and Information Sciences. Begell House Inc. Vol. 47, Issue 9, 2015. pp.63-75.

[21] Savchenko, V., Akhramovych, V., Tushych, A., Sribna, I., Vlasov, I. Analysis of Social Network Parameters and the Likelihood of its Constraction. International Journal of Emerging Trends in Engineering Research. Volume 8 No. 2. 2020. pp. 271–276.

[22] O.Laptiev, V.Savchenko, A.Kotenko, V. Akhramovych, V. Samosyuk, G.Shuklin, A.Biehun Method of Determining Trust and Protection of Personal Data in Social Networks. International Journal of Communication Networks and Information Security (IJCNIS), Vol. 13, No. 1, 2021. pp.15-21.

[23] Ivanova, D., Starkova, O. and Herasymenko, K. Realization of the Remote Power Management System Based on the Concept of Internet of Things. In Proceedings of the IEEE Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T), 2016, Kharkov: IEEE Ukraine Section, pp. 96-98.

[24] Polianytsia, A., Starkova, O. and Herasymenko, K. Survey of Hardware IoT platforms. In Proceedings of the IEEE 4th International Scientific-Practical Conference Problems of Infocommunications Science and Technology, (PIC S and T 2017), Kharkov: IEEE Ukraine Section, 2017, p. 369-371.

[25] Starkova, O., Herasymenko, K. and Babailova, Y. Remote Control Systems of Household Appliances. In Proceedings of the IEEE 4th International Scientific-Practical Conference Problems of Infocommunications Science and Technology, (PIC S and T 2017), Kharkov: IEEE Ukraine Section, 2017, pp. 585-588.

[26] Pliushch, O.G. “Gradient Signal Processing Algorithm for Adaptive Antenna Arrays Obviating Reference Signal Presence,” presented at the IEEE International Scientific-Practical Conference PIC S&T, Kyiv, Ukraine, October 8–11, 2019, p. 190.

[27] O.Maksymuk, V.Sobchuk, I.Salanda, Yu. Sachuk. A system of indicators and criteria for evaluation of the level of functional stability of information heterogenic networks. Mathematical Modeling and Computing. 2020. Vol. 7, No. 2. pp. 285 – 292

[28] Dudnik, A., Daria, P., Kobylchuk, M., Domkiv, T., Dahno, N., Leshchenko, O. (2020, November). Intrusion and Fire Detection Method by Wireless Sensor Network. In 2020 IEEE 2nd International Conference on Advanced Trends in Information Theory (ATIT)pp. 211-215.

[29] Leshchenko, O., Trush, O., Dahno, N., Dudnik, A., Kazintseva, K., & Kovalenko, O. (2020, November). Methods for Predicting Adjustments to the Rates of Modern “Digital Money”. In 2020 IEEE 2nd International Conference on Advanced Trends in Information Theory (ATIT), pp. 222-226).

[30] V. Bondarenko, “Subjective-probability approach to design an expert system for assessment of states of complex systems in conditions of non-regular destructive influences”, 2019 IEEE International Conference on Advanced Trends in Information Theory. 18.12.2019-20.12.2019. Kiev Ukraine. pp. 183-186.

[31] K.Park, K.Lee, S.Park, H.Lee, “Telecommunication node clustering with node compatibility and network survivability requirements” Management Science, vol. 46(3), 2000, pp.363-374.

[32] M. Rakushev,Y. Kravchenko, O. Permiakov, O. Lavrinchuk,V. Bychenkov, V Krainov, “Modeling of solving stabilized differential equations by differential-Taylor transformations”, IEEE 2nd International Conference on Advanced Trends in Information Theory, ATIT`2020, Proceedings, pp. 216–221.

[33] G. Vlasyuk, Y. Kravchenko, O. Starkova, K. Herasymenko, A. Polianytsia, “Implementation of the Internet of things concept for remote power management”, 2nd International Conference on Advanced Information and Communication Technologies, AICT 2017 - Proceedings. pp. 26–30.