BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2019
http://excel.fit.vutbr.cz

Modeling of OSPFv3 and BGPv4 Routing Protocols

Lukáš Galbička, Adrián Novák*

Abstract
This paper deals with modeling and simulation of OSPFv3 and BGPv4 protocols. OSPFv3 and BGPv4 are widely used routing protocols. In their newest version are treated as modern multi-address family protocols, which means they supports both IPv4 and IPv6 routing. The resulting model may be used to demonstrate routing mechanisms in real networks. They are both implemented in OMNeT++ Discrete Event Simulator as a part of ANSA and INET frameworks. A contribution of this work is that no working model of OSPFv3 has been yet implemented in any other simulators that are similar to OMNeT++. BGPv4 is implemented in INET4 for IPv4 network layer protocol support only and there are some issues with the current version. The version of BGPv4 does not support multi address-family routing.

Keywords: OMNeT++ — OSPFv3 — BGPv4— INET — ANSA — Network Simulation

Supplementary Material: Github repository for ANSA including OSPFv3 — Github repository for INET4 with BGPv4 multi address-family support

*xgalbi01@stud.fit.vutbr.cz, xnovak1j@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

Computer networks are growing in modern days to enormous measures. It is impossible to administrate such network just by static routing so new more complex and effective ways were introduced, like dynamic routing. There are two types of dynamic protocols. One for routing inside of autonomous system (AS), and the second one for routing between them. The main representative of the first type is Open Shortest Path First (OSPF) and for the second type, it is Border Gateway Protocol (BGP). There are lots of ways how to create new network topology and choose the most appropriate routing protocol. Create new projects would be very difficult and in case of any misconduct, a price for realization can grow to enormous numbers. This is the reason, why network simulations are widely used and why network simulations are the best practice when new computer networks are created.

Currently, there is support for OSPFv2 for IPv4 and basic BGPv4. The OSPF is ready to be used for simulation of networks. However, OSPFv2 supports only IPv4 network protocol routing and it only supports one OSPF process per router. It also lacks the support of the Not-So-Stubby Area (NSSA) option present in OSPF. Model in its current implementation can create neighborships and hardly fulfill the exchange of Link-State database between routers [1]. The current implementation of BGPv4 works with IPv4 network protocol and support for multi-address family routing is not implemented yet. Simulation models are being implemented in a software called OMNeT++ [2]. OMNeT++ is a discrete event simulator mostly used for simulation of computer networks. The main extension of OMNeT++ is open-source framework INET [3], which provides various models of protocols such as IPv4, IPv6, TCP, UDP or RIP. OSPFv3 as part of this

paper is part of Automated Network Simulation and Analysis (ANSA) [4]. ANSA is a project which aims is to extend the INET framework functionality.

This paper focuses on implementing an extension of existing simulation models for OSPF and BGP protocols. The aim is to create independent simulation modules representing the OSPFv3 and BGPv4 process. The result will be capable of demonstrating how the routing mechanisms in these protocols work. The main advantage of this work is that it will provide a tool to simulate dual-stack routing with both IPv4 and IPv6 on a single device. Since there are not many tools to simulate IPv6 routing, this work will have a significant impact.

The paper is divided into several chapters. At first, there is a necessary theoretical background for dynamic routing and OSPF and BGP protocols. Next, there is described design of both protocols as they are trying to be as accurate to their real functionality as it is possible. After that paper continues with a chapter about the implementation of both protocols in OMNeT++. After implementation comes testing which shows the validity of both models. At the end of the paper is provided short conclusion and references.

2. Theoretical Background

2.1 Dynamic routing

Dynamic routing protocols exchange routing information between routers. A routing protocol is a set of processes and messages that are used to exchange routing information and fill up routing tables with best paths. Dynamic routing protocols are more flexible and provide better scalability, than manual static routing configured by a network administrator. The main tasks of routing protocols are:

- discovering remote networks
- maintaining up-to-date routing information in the routing table
- choosing the best paths to destination networks
- prevent any loops
- reacting to any changes in the network

There are two types of routing protocols. The first one is called Interior Gateway Protocol (IGP) and they are used for distribution of routing information within one AS. They are further divided into more simple Distance-vector protocols, where routers share they routing information only with their neighbors and more complex Link-state routing protocols. Router using link-state routing protocols holds the whole topology in the form of the graph and share this information with every other router within the protocol domain. Every router independently calculates the best path to every device in the network using only its local topology. For the best path calculation in link-state protocols is mostly used Dijkstra algorithm. Two best-known link-state protocols are OSPF and Intermediate System to Intermediate System [5].

The second type is called Exterior Gateway Protocol (EGP) and they distribute routing information between routers from different AS. BGP is a commonly used protocol of this type and is used for routing on The Internet. BGP belongs to Path-vector routing protocols, where Path-vector is represented by alist of AS, which lead to the destination.

2.2 OSPFv2 and OSPFv3

OSPF was designed to work with TCP/IP Internet and is inherently classless. [6] OSPF uses only a small amount of network traffic and is designed to have very short convergence times when a change in the topology occurs. Routing information received from other routers is stored in a link-state database, which is later used by The Shortest Path First (SPF) to calculate the best paths to destination networks.SPF algorithm is used to find the shortest path in a graph by creating a shortest path tree.

The OSPFv3 Process running over IPv4 and IPv6 has the same core algorithms and structures. OSPFv3 does not use own authentication anymore. The authentication fields were left out because OSPFv3 relies on IPsec natively present in IPv6. OSPF uses Link State Advertisements (LSAs) to distribute information about reachable networks. OSPFv3 modifies previous LSAs and introduces new ones. OSPFv3 runs over IPv6 but it is capable of routing IPv4 as well [7]. This feature is implemented in the form of independent instances. Each interface is associated with a single process but each process may have multiple instances. Each instance is identified by instance ID (unique number which identifies this instance across all routers in the OSPFv3 domain) and address family (this may be either IPv6 or IPv4) [8][1].

2.3 BGP

The Border Gateway Protocol is used for routing between ASs [9][10]. The primary function of a BGP router is to exchange network reachability information with other BGP routers. This information includes Path-vector, which is the list of AS numbers that are used for reaching a specific destination. The AS list is good prevention from routing loops too. If border router in one AS receives routing information from another AS and sees its own AS number in the AS list that border router ignores that information because

that may cause routing loops. BGP creates a connection over TCP on port 179. BGP version 4 works with IPv4 network addresses. For routing over IPv6, it is necessary to use Multi-Address Family routing by Multiprotocol Extensions for BGP.

2.3.1 Multiprotocol Extensions for BGP-4

Multiprotocol Extensions [11] allows BGP protocol to carry routing information for multiple network layer protocols. These extensions are backward compatible, that means router without these extensions can communicate with routers with extensions support.

The first part of this extension deals with extending BGP Open message with Capability Advertisement optional parameter, which specifies the Address Family Identifier (AFI) is carried in BGP connection [12]. Either AFI = 1 for IPv4, or 2 for IPv6 [13].

Multiprotocol extensions use two new optional attributes in BGP Update message. Multiprotocol Reachable NLRI (MP_REACH_NLRI) and Multiprotocol Unreachable NLRI (MP_UNREACH_NLRI). The first one (MP_REACH_NLRI) is used to carry the set of reachable destinations together with the next hop information to be used for forwarding to these destinations. The second one (MP_UNREACH_NLRI) is used to carry the set of unreachable destinations. Both of these attributes are optional and non-transitive.

3. Design

Both models are designed to provide the same functionality as OSPFv3 and BGPv4 running on Cisco devices and to be in accordance with related RFC files. OSPFv3 protocol is implemented in *INET 3.3* and *OMNeT++ 5.0*. BGPv4 is created for *INET 4.0* and *OMNeT++ 5.4.1*.

OSPFv3 is designed to support multiple processes, each designed as a separate module, running on a single device as well as two address families per process. Inside the process, there are two instances available, one per each AF. Each router's interface, which participates in the routing process, is associated with exactly one process allowing to run maximum of two instances (one for IPv4 and the other for IPv6).

BGP model creates TCP sessions (each one with specific parameters). One of these parameters is flag which defines what address-family protocol is carried in that session (whether it is IPv4 or IPv6 AF).

4. Implementation

Both models are implemented in OMNeT++, which is based on C++ programming language. OMNeT++ is discrete component-based simulator. Each component is implemented in C++ and models are created from these components defined in NED high-level language.

4.1 OSPFv3

The whole OSPFv3 is implemented as an extension of the INET framework, which completely encapsulates the whole functionality of OSPF. For anyone who would like to use the OSPFv3 routing module, it would appear as a separate module which is capable of simulating OSPFv3 processes. The Splitter module is responsible for the initial creation of each process. RFC 5340 states that there may be multiple instances on a single link. Cisco, on the other hand, allows only a single instance per process and address family. This model meets somewhere in the middle. It allows a maximum of two processes per interface, each for one AF, but each process may have multiple instances.

Each process works with its own list of areas. Each area may be spread across multiple interfaces and each interface may be associated with multiple neighbors,

The main contribution of our work is extending existing OSPF implementation models with SPF calculation for both IPv4 and IPv6. Moreover, we heavily refactorized existing code-base fixing a lot of bugs and divergences from expected behavior, such as incorrectly generating almost every LSA message, redundant Hello packets or long-lasting neighborhood adjacency. All link-state advertisements (LSAs) now obtain capabilities for routing calculation and their aging inside LSDB (except Inter-Area-Prefix LSA). Route calculation is capable of computing routes inside one OSPF domain within one area or split between two or more of them. The protocol now also dynamically react to changes in topology.

Despite huge contribution for OSPFv3, implementation of the protocol does not involve some functionality. There is support for different network types configuration, such as Point-to-point, Non-broadcast-multi-access or Point-to-multipoint network, but no routing between these networks. The implementation also lack inter-autonomous-system routing, checksum calculating or options managing.

4.2 BGPv4

Firstly, there were fixed some issues with BGPv4 implementation for IPv4 support, such as correct creation and processing of BGP Update message, BGP finish state machine consistency, model's reactions on link drops, BGP Update message sending on multi-point segments and the existence of internal peering only.

Next multi address-family support was implemented. The `BGPRouter6` is created in NED language. Scheme of this router is shown in the Figure 1.

`BGPRouter6` extends classic `BGPRouter` and has IPv6 module enabled with specified `Ipv6RoutingTable`.

Then there were created data structures for IPv6 and multi-address support. These structures are – BGP routing table for IPv6, updated BGP Open message with Capability Advertisement optional parameter for identifying address family, new BGP Update message for IPv6 support with Multiprotocol extension.

New methods were implemented for working with these data structures. The XML configuration file was updated too. A new configuration file is more like Cisco configuration of BGP protocol. The part of new configuration file is shown in the Figure 2. Figure shows configuration of interfaces, configuration of the BGP router with multi address-family support and static routes configuration.

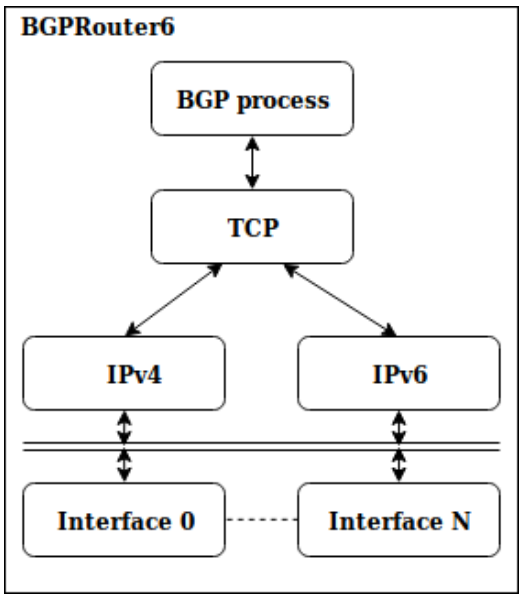

Figure 1. Structure of BGP Router in OMNeT++

5. Testing

This chapter presents some tests, which show, that modules are correctly implemented and can be used for simulation purposes. More information about testing of both models is described in their subsections.

5.1 OSPFv3

For testing of OSPFv3 module, we used real topology consisting of four Cisco routers R1-R4 divided into two areas, interconnected as is shown in Figure 3. R1 is in area 1, R3 in area 0, R2 and R4 are Area Border Routers (ABR). The same topology is built in OMNeT++ where simulation of topology takes place.

Two main features which show the right functionality of simulated OSPFv3 protocol is correctly built LSDB and computed routing table. So the first pair of

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<BGPConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="BGP.xsd">
    <TimerParams>
        <connectRetryTime> 120 </connectRetryTime>
        <holdTime> 180 </holdTime>
        <keepAliveTime> 60 </keepAliveTime>
        <startDelay> 15 </startDelay>
    </TimerParams>

    <Devices>
        <Router name="R2" id="200.0.2.1">
            <Interfaces>
                <Interface id="eth1">
                    <Ipv4 address="10.0.20.1" netmask="255.255.255.252" />
                    <Ipv6 address="fd00:20:20::0/127" />
                </Interface>
                <Interface id="eth0">
                    <Ipv4 address="10.0.12.2" netmask="255.255.255.252" />
                    <Ipv6 address="fd00:12:12::1/127" />
                </Interface>
                <Interface id="lo0">
                    <Ipv4 address="200.0.2.1" netmask="255.255.255.252" />
                    <Ipv6 address="fd00:200:0:200::1/64" />
                </Interface>
            </Interfaces>
            <Bgp as="200">
                <Address-family id="Ipv4">
                    <Neighbor Addr="10.0.12.1" remote-as="100" />
                    <Neighbor Addr="10.0.20.2" remote-as="200" />
                    <Network Addr="200.0.2.0" />
                    <Network Addr="200.0.3.0" />
                </Address-family>
                <Address-family id="Ipv6">
                    <Neighbor Addr="fd00:12:12::0" remote-as="100" />
                    <Neighbor Addr="fd00:20:20::1" remote-as="200" />
                    <Network Addr="fd00:200:0:200::" />
                    <Network Addr="fd00:200:0:300::" />
                </Address-family>
            </Bgp>
            <Route destination='200.0.3.0' netmask='255.255.255.252' interface='eth1'/>
            <Route6 destination='fd00:200:0:300::/64' interface='eth1' nexthop="fd00:20:20::1"/>
        </Router>
    </Devices>
</BGPConfig>
```

Figure 2. New configuration file for BGP router in OMNeT++

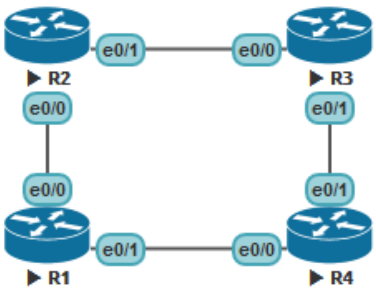

Figure 3. Topology for OSPFv3 testing

the picture shows a comparison of LSBD of R1 built upon real router (Figure 4) and on simulated one (Figure 5). As we can see, LSBD is the same. Testing also showed that topology converged roughly at the same time.

After LSDB is built, Dijkstra's SPF algorithm computes all needed routing table entries, from which are the necessary one picked into the routing table of the router. Figure 6 shows the routing table of R1 inside of real topology and Figure 7 shows the same router inside of a simulation. As we can see, the computed routes are same what means that SPF was calculated right. Differences in administrative distances and metrics are accepted since they are caused by slight configuration discrepancies.

5.2 BGPv4

For testing purposes of BGPv4 module, we create the topology, which consists of four routers R1 – R4. The topology is divided into three AS. R1 is in AS 100 and will create external peering with router R2. R2 and R3 are in AS 200 and they will create internal peering with each other. The last one R4 is in AS

```
          OSPFv3 1 address-family ipv6 (router-id 1.1.1.1)

               Router Link States (Area 1)

ADV Router       Age         Seq#        Fragment ID  Link count  Bits
 1.1.1.1          8          0x80000001  0            2           None
 2.2.2.2          8          0x80000003  0            1           B
 4.4.4.4          8          0x80000005  0            1           B

               Net Link States (Area 1)

ADV Router       Age         Seq#        Link ID    Rtr count
 2.2.2.2          8          0x80000001  3          2
 4.4.4.4          8          0x80000001  3          2

               Inter Area Prefix Link States (Area 1)

ADV Router       Age         Seq#        Prefix
 2.2.2.2          64         0x80000003  2222::/64
 2.2.2.2          8          0x80000001  3333::/64
 4.4.4.4          64         0x80000001  3333::/64
 4.4.4.4          8          0x80000003  2222::/64

               Link (Type-8) Link States (Area 1)

ADV Router       Age         Seq#        Link ID    Interface
 1.1.1.1          64         0x80000003  4          Et0/1
 4.4.4.4          64         0x80000002  3          Et0/1
 1.1.1.1          64         0x80000004  3          Et0/0
 2.2.2.2          64         0x80000003  3          Et0/0

               Intra Area Prefix Link States (Area 1)

ADV Router       Age         Seq#        Link ID    Ref-lstype  Ref-LS
 2.2.2.2          64         0x80000001  3072       0x2002      3
 4.4.4.4          64         0x80000001  3072       0x2002      3
```

Figure 4. OSPFv3 Link-state database of router R1 inside of real topology

Router Link States (Area 0.0.0.1)

ADV Router	Age	Seq#	Fragment ID	Link count	Bits
1.1.1.1	9	0x80000002	0	1	None
2.2.2.2	9	0x80000001	0	1	B
4.4.4.4	9	0x80000001	0	1	B

Net Link States (Area 0.0.0.1)

ADV Router	Age	Seq#	Link State ID	Rtr count
2.2.2.2	9	0x80000001	0.0.0.101	2
4.4.4.4	9	0x80000001	0.0.0.101	2

Inter Area Prefix Link States (Area 0.0.0.1)

ADV Router	Age	Seq#	Prefix
2.2.2.2	49	0x80000001	2222::/64
4.4.4.4	49	0x80000001	3333::/64
4.4.4.4	9	0x80000003	2222::/64
2.2.2.2	9	0x80000003	3333::/64

Link (Type-8) Link States (Area 0.0.0.1)

ADV Router	Age	Seq#	Link State ID	Interface
1.1.1.1	49	0x80000001	0.0.0.101	eth0
2.2.2.2	49	0x80000001	0.0.0.101	eth0
1.1.1.1	49	0x80000001	0.0.0.102	eth1
4.4.4.4	49	0x80000001	0.0.0.101	eth1

Intra Area Prefix Link States (Area0.0.0.1)

ADV Router	Age	Seq#	Link ID	Ref-lstype	Ref-LSID
2.2.2.2	50	0x80000001	0.0.0.2	0x2002	0.0.0.101
4.4.4.4	50	0x80000001	0.0.0.2	0x2002	0.0.0.101

Figure 5. OSPFv3 Link-state database of router R1 inside of simulation

300 and will create external peering with router R3. Each router announces network which is connected to `loopback` interface. Routers in AS 200 sends network of internal peer too. This topology with IP addresses of `loopback` networks is shown in the Figure 8.

There are used two main methods for validating our models.

The first method is to compare the output from real network topology recorded by Wireshark software and

```
1111::/64 [0/0]
 via Ethernet0/0, directly connected
1111::1/128 [0/0]
 via Ethernet0/0, receive
2222::/64 [110/20]
 via FE80::A8BB:CCFF:FE00:200, Ethernet0/0
3333::/64 [110/20]
 via FE80::A8BB:CCFF:FE00:400, Ethernet0/1
4444::/64 [0/0]
 via Ethernet0/1, directly connected
4444::1/128 [0/0]
 via Ethernet0/1, receive
FF00::/8 [0/0]
 via Null0, receive
```

Figure 6. OSPFv3 Routing table of router R1 inside of real topology

```
routeList (IPv6Route *>)
  elements[6] (inet::IPv6Route *)
    [0] = C 1111::/64 is directly connected, eth0
    [1] = C 4444::/64 is directly connected, eth1
    [2] = O 2222::/64 [255/2] via fe80::a8bb:ccff:fe00:200, eth0
    [3] = O 3333::/64 [255/2] via fe80::a8bb:ccff:fe00:400, eth1
    [4] = S fe80::/10 is directly connected, eth0
    [5] = S fe80::/10 is directly connected, eth1
```

Figure 7. OSPFv3 Routing table of router R1 inside of simulation

output from simulation models created in OMNeT++ framework. With this method, it is possible to validate type, values, and order of messages. An example of compared BGP Update message are shown in Figures 9 and 10. This BGP Update message is sent by router R2 to its external peer router R1. The message carries routing information about network announced by router R4 and uses IPv6 network layer protocol.

The second method compares data structures of converged real network topology with one of the simulation models. These data structures are routing tables working with specific routing protocol and primary routing tables. They should contain the same information in the simulation and in the real network topology. An example of compared BGP routing tables for IPv6 is shown in Figures 11 and 12. These IPv6 routing tables contain the same routes for router R1, that means simulation model works as well as a real router.

Compared messages and routing tables contain the same values, so implemented multi address-family extension of BGPv4 protocol is valid and works fine.

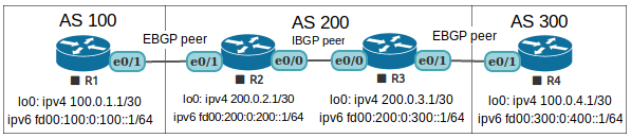

Figure 8. BGP testing topology

Figure 9. BGP Update message for IPv6 in OMNeT++

6. Conclusions

This work focuses on creating and extending simulation models of link-state protocol OSPFv3 and path-vector protocol BGPv4. It is implemented in the OMNeT++ simulation environment. It improves the current OSPF and BGP model presented in the INET framework.

The most important contribution of both models is multi address-family routing. That means models are now able to work with IPv4 and IPv6 network

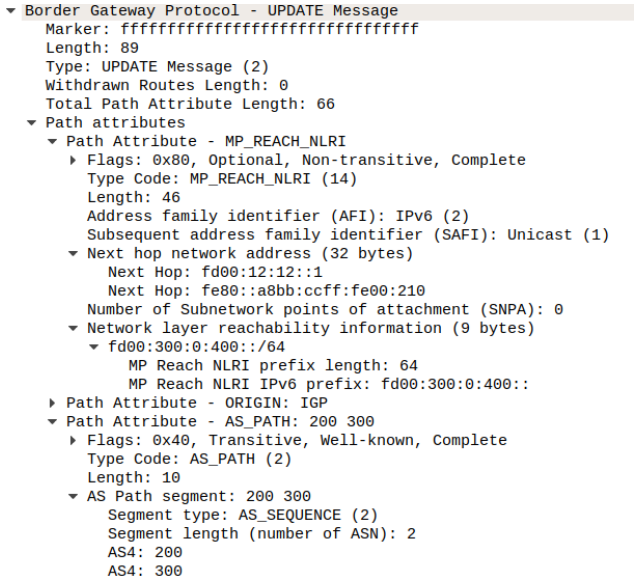

Figure 10. BGP Update message for IPv6 in Wireshark

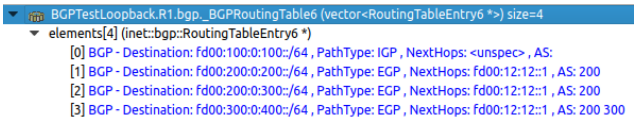

Figure 11. BGP routing table for IPv6 in OMNeT++

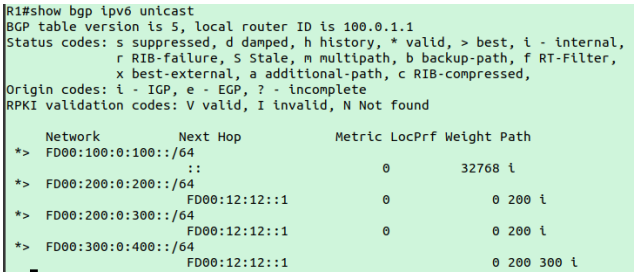

Figure 12. BGP routing table for IPv6 in real network

layer protocol at the same time. Another new feature for BGP is the new configuration file, which allows configuring devices more comfortable than before and brings more Cisco like configuration.

Properly working simulator like this can save an enormous amount of time and resources. These new simulation models can provide an easy way to simulate and validate networks, test ideas and build new topologies using single network simulator. This can be useful for any network architect and internet service provider.

Acknowledgements

We would like to thank our supervisor Ing.Vladimír Veselý Ph.D. for his support and valuable advice.

References

[1] Michal Ruprich. Modelling of ospfv3 link-state routing protocol, 2016.

http://excel.fit.vutbr.cz/submissions/2016/033/33.pdf, [Online; cit. 7.3.2019].

[2] András Varga. *OMNeT++ Discrete Event Simulator*. [Online; cit. 7.3.2019].

[3] *INET Framework*. [Online; cit 7.3.2019].

[4] Brno University of Technology Faculty of Information Technology. *Project ANSA*. [Online; cit. 7.3.2019].

[5] H. Smit and T. Li. Intermediate system to intermediate system (is-is) extensions for traffic engineering (te). RFC 3784, RFC Editor, June 2004.

[6] J Moy. Rfc 2328. *OSPF version 2*, 1998. [Online; cit. 7.3.2019].

[7] R Coltun, D Ferguson, J Moy, and A Lindem. Rfc 5340, ospf for ipv6. *IETF. July*, 24, 2008. [Online; cit. 7.3.2019].

[8] Acee Lindem, Sina Mirtorabi, Michael Barnes, Abhay Roy, and Rahul Aggarwal. Support of address families in ospfv3. 2010. [Online; cit. 7.3.2019].

[9] Yakov Rekhter, Susan Hares, and Tony Li. A Border Gateway Protocol 4 (BGP-4). RFC 4271, 2006. https://rfc-editor.org/rfc/rfc4271.txt, [Online; cit. 7.3.2019].

[10] Brad Edgeworth Vinit Jain. *Troubleshooting BGP: A Practical Guide to Understanding and Troubleshooting BGP*. Cisco Press, 2016.

[11] Tony J. Bates, Ravi Chandra, Yakov Rekhter, and Dave Katz. Multiprotocol Extensions for BGP-4. RFC 4760, 2007. https://rfc-editor.org/rfc/rfc4760.txt, [Online; cit. 7.3.2019].

[12] Ravi Chandra and John G. Scudder. Capabilities advertisement with bgp-4. RFC 4271, 2002. https://tools.ietf.org/html/rfc3392, [Online; cit. 7.3.2019].

[13] Internet Assigned Numbers Authority. *Address Family Numbers*. https://www.iana.org/assignments/address-family-numbers/address-family-numbers.xhtml, [Online; cit. 7.3.2019].