Analyze OSPF Convergence Time in the Presence of Single and Multiple Failures

Cristina-Loredana Duta, Laura Gheorghe, Nicolae Tapus
Department of Computer Science and Engineering
University Politehnica of Bucharest, Bucharest, Romania
Email: cristina.duta.mapn@outlook.com, laura.gheorghe@cs.pub.ro, nicolae.tapus@cs.pub.ro

***Abstract*—Open Shortest Path First (OSPF) is a widely used link-state routing protocol in IP networks. Processing delays in OSPF implementations have an effect on the time necessary for inter-domain and intra-domain routing to re-converge after a topology change. OSPF implements different timers in order to reduce the protocol overhead. These timers ensure that the OSPF network takes several tens of seconds to recover from a failure. The delay that appears in the convergence time is due to failure detection, more specifically, is due to the value of timers and of routing calculation scheduling. In this paper, we evaluate OSPF convergence time in the presence of single or multiple failures using Quagga software routing engine and Mininet simulated network environment. The purpose is to understand the impact of failures on convergence, to observe their effects on end-to-end traffic and to determine what components should be taken into consideration in order to reduce the convergence time in a network topology based on OSPF.**

Keywords-OSPF; failure; convergence; routing software Quagga; Mininet.

I. INTRODUCTION

Nowadays, the popularity of information and communication technologies is increasing as new high bandwidth applications and services based on streaming are emerging. Because of this rapid technological advance, there is a growing demand for high-performance switching and transmission equipment. Compared with Personal Computers (PCs), where standards for development have been defined since the beginning, the field of networking equipment (that of packet switching more specifically), has always supported the development of proprietary architectures.

Routers represent the key component of the Internet infrastructure because they are interconnected through networks or links to form a backbone network which can guarantee communications between Internet users. In general, routing protocols are used in dynamic environments [1] where they have the purpose to constantly monitor any changes of the network or any events that appear [2, 3]. These functions are usually implemented at local level, in routers.

An essential characteristic of a routing protocol, which impacts end-to-end performance, is how fast it converges when topology changes happen. Convergence is when all routers have their routing tables in a state of consistency [4]. The key factor that distinguishes different routing protocols is the convergence time. Based on the speed of convergence, the routing protocols can be evaluated: the faster the convergence is, the better the routing protocol is [5].

Issues regarding convergence have been identified and analyzed at the beginning for Border Gateway Protocol (BGP) [6], but nowadays, OSPF has become widely used in the Internet infrastructure.

The purpose of this paper is to measure and analyze OSPF convergence in the presence of single and multiple failures and their impact on end-to-end traffic. We have created a simple topology and we have investigated the routing convergence under five different situations: two single link failures and three multiple link failures between different routers according to topology which will be presented further on. All the experiments were performed 10 times and each time 50 ping packets were sent.

We present the fundamental concepts regarding OSPF convergence, we analyze the impact of single and multiple failures on convergence dynamics and we describe some methods that can be useful for improving network convergence.

The rest of the paper is organized as follows. The necessary background for our work is presented in Section 2. Related work is presented in Section 3. Section 4 gives an overview of the test scenario created and it also includes a briefly outline about the convergence process and the related timers of OSPF. Section 5 offers details about our implementation, about the analysis we performed using different tests scenarios as well as the experimental results obtained and includes the adjustments of the parameters which we have done with the purpose to minimize the convergence time. Section 6 draws the conclusions for analysis of OSPF convergence behavior in the presence of single or multiple failures.

II. BACKGROUND

In this section, we present some details about OSPF, Mininet, which is the network emulator we have used in our scenarios and Quagga, which is the routing software we have selected for this evaluation.

A. Open Shortest Path First (OSPF)

OSPF [7] is a non-proprietary routing protocol which was developed in 1998 and is widely used in intra-domain Internet Service Provider (ISP) networks. OSPF is a link-state protocol that has the purpose to manage the routing table in order to use the best path to reach destination during packet forwarding.

Link-state feature is related to the functioning mode of OSPF: each OSPF router describes its topological situation,

 ISBN: 978-1-61208-422-0

its active links to all the connected counterparts so that every router knows exactly the entire topology. Link State Advertisements (LSAs) are crucial to OSPF and building the topology. LSAs represent the mean through which routers know about each other's links and who connects to whom across an area.

When the network topology changes (an event was produced – e.g., a link is down), the router communicates with the neighboring routers to determine the state of all adjacencies. The protocol used is the Hello protocol, in order to detect the failure and then generate new LSAs. LSA dissemination is done through a flooding mechanism: when a router receives a new LSA (this notifies a topological change), the LSA is sent through all of the router's interfaces, except the one it has received the new LSA from.

After the LSAs are synchronized through the mechanism previously described, the routers can correctly calculate the routing table for packet forwarding. To compute the shortest path to all destinations the Dijkstra algorithm [8] is applied, having the router as root node – in this way every router will calculate a different shortest path sub-graph.

B. Mininet network emulator

Mininet [9] is a widely used open source network emulator that can simulate a number of end-hosts, switches, routers, and links on a Linux kernel. Mininet offers several advantages such as: speed (a simple network takes only a few seconds to start up), creation of custom topologies, running real programs (anything that runs on Linux can be executed by the user of Mininet too), customization of packet forwarding, ease of use and active development.

Because it is easy to interact with the created network using Mininet Command Line Interface (CLI), to customize it, to deploy it on real hardware and to share it with others, Mininet is very useful for teaching, development and research. Mininet can be very helpful to develop and experiment with OpenFlow and Software-Defined Networking (SDN) systems.

C. Quagga routing software

Quagga is a fork of GNU Zebra Project [10] which started in 1996 by an idea of Kunihiro Ishiguro. It is a routing software package that manages TCP/IP based routing services with routing protocols support such as: BGP, Routing Information Protocol (RIP) v1, RIPv2, RIPng, OSPFv2, and OSPFv3. Quagga, allows the machine of the user to exchange routing information with other routers through specific protocols. The information gathered is used to update the kernel routing table in order to ensure the correct placement of data. Quagga can setup interface's address, flags, static routes and others. There are two modes available: normal mode and enable mode. The first one allows the user to view only the system status and the second one allows him to change the system's configuration.

It is composed of a collection, including different daemons that interact in order to build together the routing table: *RIPD* – which handles RIP protocol; *OSPFD* – which supports OSPF version2; *BGPD* – which hands BGP-4 protocol; *ZEBRA* – allows establishing communication between underlying Linux kernel and the other routing protocol daemons. For instance, if it is necessary to change the kernel routing table and to redistribute the routes between different routing protocols, *ZEBRA* sends a specific message to the kernel; *VTY* – is an additional daemon which allows configuring different routing protocols through a network accessible CLI, which accepts commands similar with the ones used on Cisco devices.

III. RELATED WORK

This section presents various methods and techniques for analyzing and improving OSPF convergence, which are described by researchers in other articles.

When dealing with protocol design, the main goal is to limit the processing power or bandwidth requirements of the protocol, while the time necessary to recover from a failure in the network topology is of secondary importance. The trade-off between efficiency and overhead can be adjusted using protocol timers. For example, Hello packet is sent periodically between neighboring routers with the frequency established by *HelloInterval* (this limits the number of hello packets).

Considering real-time applications, researchers have focused to achieve fast convergence to ensure uninterrupted traffic delivery. For instance, Francois et al. [11] tried to obtain sub-second Interior Gateway Protocol (IGP) convergence in large IP networks. Their implementation is highly dependent on the existing network resources and the frequency of failures. To achieve sub-second convergence they decreased the OSPF timers, which have an important impact on network stability.

Basu and Riecke [12] have been studying stability issues. Stability is necessary if there is a change in the network topology, all the nodes are guaranteed to converge to the new network topology in finite time, in the absence of other events. Hence, the controversy between fast convergence and protocol stability requires continuous study and research.

Referring to the topic of improving OSPF convergence is of high interest for the network research domain. Some articles propose algorithms and schemes in order to avoid the convergence process. For instance, IETF IP Fast Reroute (IPFRR) framework [13] proposes the use of pre-computed backup paths in order to reroute around the failures in the network. In [14], the authors propose a new routing scheme, which has the purpose to eliminate the convergence process completely. They present a new technique which allows packets to autonomously discover a working path. In [15], the authors present a solution that involves using network graphs and the corresponding link weights to produce a set of backup network configurations. The disadvantage of the previously mentioned approaches is that they are similar to patches, which means that they need to be added to the protocol and that they assume complex configurations.

Most of the papers that study OSPF convergence are focused on the assumption that a single failure that affected the network topology has occurred. This is good because it is widely known that network failures are of the type single link failure [16], but we have to keep in mind the fact that sometimes multiple failures can occur. In this context, we

 ISBN: 978-1-61208-422-0

decided to analyze the behavior of OSPF when dealing with single and multiple failures and to compare the results.

The researchers show that multiple failures can occur due to electromagnetic pulse (EMP) attacks [17], to natural disasters such as floods, hurricanes and earthquakes. The contribution in this area implies the presentation of guidelines for topology design and for maintenance.

In this paper, we aim to understand and analyze the behavior of OSPF convergence in the presence of single or multiple failures. Also, we present some ideas that can improve network convergence.

Due to the fact that the test scenarios vary from other similar studies (we have taken into consideration single and multiple failure for a specific network topology, performed each experiment 10 times for measurements and moreover transmitted each time 50 ping packets) it is difficult to compare our results with the ones obtained by other researchers.

IV. OSPF CONVERGENCE AND TIMERS

Network convergence represents the process of synchronizing network forwarding tables after a topology change. A network has converged when none of the forwarding table is changing for a "reasonable" amount of time. The amount of time can be defined as an interval, based on the expected maximum time to stabilize after a single topology change is produced. A diagram which represents the process of OSPF convergence is presented in Figure 1.

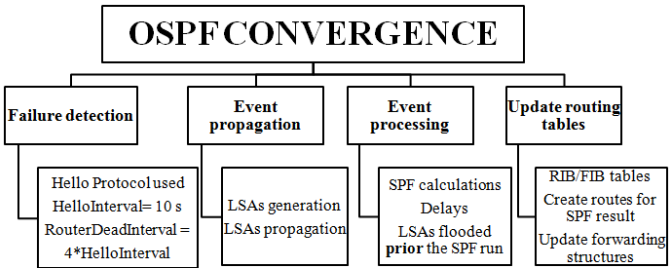

Figure 1. OSPF convergence diagram

Network restoration, which repairs the lost connections, is actually network convergence based on native IGP mechanisms. An important observation regarding IGP-based restoration techniques is the following: during the time of re-convergence, temporary micro-loops may appear in the topology (due to inconsistency of Forwarding Information Base Tables of different routers). This is very important for algorithms, because routers closer to the failure tend to update their forwarding database before the other routers.

$$Convergenc\ \ e = Failure_{\ DetectionT\ \ ime} + Event_{\ \Pr opagationT\ \ ime} + SPF_{\ RunTime} + RIB\ /\ FIB_{\ UpdateTime} \quad (1)$$

As it can be seen in formula (1), the convergence time for a link-state protocol is represented by the sum of the next components: the time necessary to detect a network failure (for instance, an interface down condition); the time it takes to propagate the event (for example, flooding the LSA, across the topology); the time necessary to perform Shortest Path First (SPF) calculations on all routers when new information is received; the time it takes to update the forwarding tables for all the routers in the area.

In this section, we describe failure detection and routing calculation related timers that are important components of convergence delay.

The top priority for fast convergence is to detect link and node failures very quickly. The primary goal is to minimize the detection/indication timers. An advantage of using point-to-point links is the fact that OSPF becomes adjacent very fast, due to the fact that Designated Routers (DRs) are no longer needed. Moreover, type 2 LSAs are not generated for point-to-point links, which reduces a little the Link-State Database (LSDB) of OSPF and also the topology complexity.

We take into consideration the fact that OSPF uses Hello protocol to detect the failure. This means that it enables routers to periodically exchange Hello packets to establish adjacency with a frequency determined by *HelloInterval*.

If Hello packets are not received by one router during *RouterDeadInterval* which is typically 4 *HelloIntervals*, the adjacency is considered down. The router that detects the failure generates new LSAs and will propagate them through the network. The default value for *HelloInterval* is considered to be 10 seconds [1]. This means that a network failure can be detected in 30 to 40 seconds after its occurrence. As it can be observed, achieving faster failure detection will significantly accelerate convergence. However, reducing the *HelloInterval* has a significant drawback: all Hello packets are processed by the router's main CPU, and if there are hundreds or more OSPF neighbors, this may have a significant impact on the router's control plane performance. The chance of false alarm increases as *HelloInterval* becomes smaller. This means that it is not recommended to reduce *HelloInterval* to the millisecond range [18].

In OSPF, topology changes are advertised using LSA/LSP (Link State Packet) flooding mechanism. To ensure that a network completely converges, a LSA/LSP must reach every router within its flooding scope.

The throttling process is controlled by three parameters: *initial* interval, *hold time*, and *max_wait time* using the command: *timers throttle lsa initial hold max_wait*.

Initial LSA generation delay has a significant impact on network convergence time, so it is important to be configured properly. The *initial delay* should be set to minimum, for instance to 5-10 milliseconds. It is not recommended to set it to zero because multiple link failure may occur synchronously.

The *hold interval* should be set so that the next LSA is sent only after the network has converged in response to the first event that occurred. In general, a single link failure results in at least two LSAs being generated, by every attached router.

Processing delay represents the time needed by the router to put the LSA on the outgoing flood lists and it is significant if the SPF process will start before flooding the LSA. Even though there are also other components that contribute to the

 ISBN: 978-1-61208-422-0

processing delay, the SPF is the most important one and we can have control over it. To ensure fast convergence, it is necessary that the LSAs are always flooded *prior* the SPF run which means that we must properly tune SPF runtime delays.

When a new LSA reaches the routers, routing calculation is scheduled. It is not recommended for the router to start executing the routing calculations immediately after receiving a LSA because more LSAs may be received and it will have to do many routing table updates in this situation. To avoid keeping the CPU busy in the case previously mentioned, OSPF uses a timer called *spfDelay* that has the purpose to delay the first routing calculation when the router receives a new LSA, so that the calculations will be performed on the entire collection of generated LSAs by the topology change. The main goal of SPF throttling is to avoid excessive calculations when the network is very unstable, but still keep the SPF reaction fast for stable networks.

V. EXPERIMENTAL EVALUATION

In this section, we perform experiments on a network topology created using an emulation system and routing software to measure and analyze OSPF convergence in the presence of single or multiple failures.

A. Test scenario

The topology is created using Mininet emulation network and in order to work with routing protocols, such as OSPF we have used Quagga routing engine. There are three applications involved.

The first application, the client, runs as a daemon in the Virtual Machine and has the role to detect changes in the Linux ARP and also in the routing tables.

The second is a standalone application, the server, which has the role to manage VMs running the client daemons. It has the role to keep the mapping between the client VMs instances and interfaces and the corresponding switches and ports. The server is responsible for deciding what to do with the packets that arrive at the controller, so it handles the protocol packets generated by Quagga and sends them out through the datapath switches.

The third application, the proxy, is responsible for interacting with the OpenFlow switches via OpenFlow protocol. The setup of the experiment is shown in Figure 2. It involves four routers each connected to a host. All of the routers will route traffic from different networks in the topology by using OSPF.

We investigate five convergence behaviors after link failure. In the first scenario, we disconnect the link between router A and router B (interface eth2) and examine the convergence time.

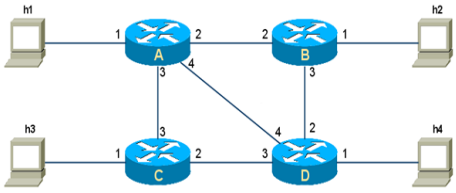

Figure 2. Evaluation Topology

In the second scenario, we remove at the same time with the link between router A and router B (interface eth2), the link between router A and router D (interface eth4) and measure the convergence time.

In the third scenario, we measure the convergence time when the link between router A and router B (interface eth2) and the link between router A and router C (interface eth3) fail at the same time.

In the fourth scenario, we determine the convergence time when the link between router B and router D (interface eth2) has failed.

The fifth experiment assumes that the link between router B and router D (interface eth2) fails at the same time with the failure of the link between router B and router A (interface eth2).

At the end of the experiment we modify OSPF timers and *HelloInterval* in order to examine any improvement in the convergence time.

B. Experimental results

In this section, we present the experimental results that we have obtained for each of the scenarios previously mentioned.

1. Single link failure (link $Router_A$–$Router_B$)

The normal traffic flows from R_A to R_B directly. In order to verify this traffic pattern, the command *traceroute* can be used. We disconnect R_B's Ethernet interface module (eth2 which connects to R_A) to simulate a broken Ethernet link.

In this case, the traffic is shifted through R_D in order to reach the host connected to R_B. We transmit fifty ping packets from R_A to R_B using ping command. We disconnect R_B's eth2 sometime during the ping command is issued.

For instance, the first 12 ping packets travel the normal path and then when R_B's eth2 is disconnected, during the transient time when the routing protocol is converging, three packets are lost. After the convergence of OSPF, the rest of the packets are sent through the backup path between R_A and R_B. In general, if there are three missing packets during the transient state of the network this indicates that OSPF needs six seconds to converge in this topology.

This ping experiment has been done ten times and the results are presented in Table I.

TABLE I. SINGLE FAILURE-RESULTS FOR OSPF CONVERGENCE

Number of experiments	Packets received	Packets lost	Convergence time (in seconds)
1	47	3	6
2	47	3	6
3	47	3	6
4	47	3	6
5	47	3	6
6	48	2	4
7	48	2	4
8	48	2	4
9	49	1	2
10	49	1	2
Average	48	2.3	4.6

 ISBN: 978-1-61208-422-0

2. Multiple link failure (link $Router_A$–$Router_B$ and link $Router_A$–$Router_D$)

The experiment setup is similar to the previous one, however this time when we transmit the fifty ping packets from R_A to R_B using ping command, we will disconnect R_B's eth2 and R_D's eth4 sometime during the ping command is issued. This ping experiment has been done ten times and the results can be seen in Table 2.

TABLE II. MULTIPLE FAILURE - RESULTS FOR OSPF CONVERGENCE

Number of experiments	Packets received	Packets lost	Convergence time (in seconds)
1	42	8	16
2	42	8	16
3	42	8	16
4	43	7	14
5	43	7	14
6	43	7	14
7	43	7	14
8	44	6	12
9	44	6	12
10	44	6	12
Average	43	7	14

As it can be observed from the results in Table I and Table II, when dealing with single link failures, the convergence time is smaller (which means faster convergence). When multiple failures occur, more delay is introduced and the convergence time rapidly increases, reaching an average value of 14 seconds for two link failures.

3. Multiple link failure (link $Router_A$–$Router_B$ and link $Router_A$–$Router_C$)

The experiment setup is similar to the previous one. The main difference is that this time when we transmit the fifty ping packets from R_A to R_B using ping command, we will disconnect R_B's eth2 and R_C's eth3 sometime during the ping command is issued. After performing the experiment 10 times, we obtained the results presented in Table III.

TABLE III. MULTIPLE FAILURE - RESULTS FOR OSPF CONVERGENCE

Number of experiments	Packets received	Packets lost	Convergence time (in seconds)
1	37	13	26
2	37	13	26
3	38	12	24
4	38	12	24
5	38	12	24
6	38	12	24
7	39	11	22
8	39	11	22
9	39	11	22
10	39	11	22
Average	38	11.8	23.6

According to Figure 3, which shows a comparison of the convergence time when we deal with single and multiple failures, it can be observed that multiple failures have a larger impact on network connectivity and protocol reaction behavior. The last values represent the average convergence time for each case.

The existence of multiple failures means multiple routing calculations which will certainly introduce more delay to convergence.

Single failure represent the situation when the link between $Router_A$ and $Router_B$ fails, the multiple failure 1 is the case when links between $Router_A$–$Router_B$ and $Router_A$–$Router_D$ fail, and the multiple failure 2 is the case when links between $Router_A$–$Router_B$ and $Router_A$–$Router_C$ fail.

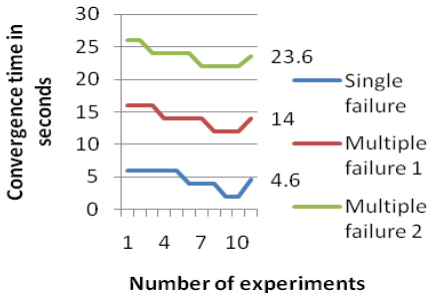

Figure 3. Comparison between convergence time when single or multiple failure occur.

4. Single link failure (link $Router_B$–$Router_D$)

The experiment setup is the same with the first, except the fact that this time when we transmit the fifty ping packets from R_B to R_D using ping command, we will disconnect R_D's eth2 sometime during the ping command is issued. This ping experiment has been done ten times and the results are presented in Table IV.

TABLE IV. SINGLE FAILURE - RESULTS FOR OSPF CONVERGENCE

Number of experiments	packets received	packets lost	Convergence time (in seconds)
1	42	8	16
2	42	8	16
3	43	7	14
4	43	7	14
5	43	7	14
6	43	7	14
7	43	7	14
8	44	6	12
9	44	6	12
10	44	6	12
Average	43	6.9	13.8

5. Multiple link failure (link $Router_B$–$Router_D$ and link $Router_B$–$Router_A$)

The experiment setup is slightly different from the previous one. This time when we transmit the fifty ping packets from R_B to R_D using ping command, we will disconnect R_D's eth2 and R_A's eth2 sometime during the ping command is issued. After performing the experiment 10 times, the results obtained can be observed in Table V.

TABLE V. MULTIPLE FAILURE - RESULTS FOR OSPF CONVERGENCE

Number of experiments	Packets received	Packets lost	Convergence time (in seconds)
1	40	10	20
2	40	10	20
3	40	10	20
4	40	10	20
5	41	9	18
6	41	9	18
7	41	9	18
8	41	9	18
9	42	8	16
10	42	8	16
Average	41	9.2	18.4

C. Options to improve OSPF convergence

Gathering all of the information above, we tried to find an optimum convergence profile based on the fact that we have different information from each router. We modify the initial *spfDelay* time, the minimum and the maximum hold time between consecutive SPF's using the command *timers throttle spf 10 100 1000* in the router's OSPF interface. After this change and taking into consideration the last case of multiple link failure (link $Router_B$–$Router_D$ and link $Router_B$–$Router_A$ fail), we obtained the values showed in Table VI for OSPF convergence time.

TABLE VI. MULTIPLE FAILURE - RESULTS FOR OSPF CONVERGENCE

Number of experiments	Packets received	Packets lost	Convergence time (in seconds)
1	43	7	14
2	43	7	14
3	44	6	12
4	44	6	12
5	44	6	12
6	44	6	12
7	44	6	12
8	45	5	10
9	45	5	10
10	45	5	10
Average	44	5.9	11.8

Another experiment that we performed is to set the *HelloInterval* to 5 seconds and to compare it to the results obtained when *HelloInterval* has the default value, which is 10 seconds. For this experiment, we have considered the case when the link between $Router_A$–$Router_B$ and the link between $Router_A$–$Router_D$ fail. The results of the convergence time are shown in Table VII and a chart with the comparison between the times can be seen in Figure 4.

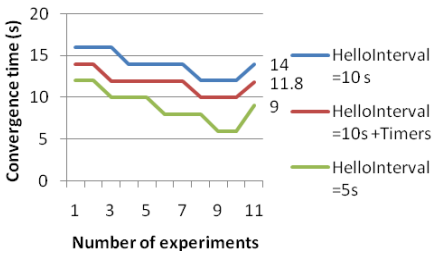

Figure 4. Comparison between convergence times with different HelloInterval values.

TABLE VII. MULTIPLE FAILURE - RESULTS FOR OSPF CONVERGENCE

Number of experiments	hellointerval=10s (no timer modifications)	hellointerval=10s (with timer modifications)	hellointerval=5s (with timer modifications)
1	16	14	12
2	16	14	12
3	16	12	10
4	14	12	10
5	14	12	10
6	14	12	8
7	14	12	8
8	12	10	8
9	12	10	6
10	12	10	6
Average	14	11.8	9

We can clearly observe from the figure that the convergence delay is increased when *HelloInterval* is larger. For instance, if the *HelloInterval* is 5 seconds, then the convergence time has an average value of 9 and if the *HelloInterval* is 10 seconds, then the convergence time reaches an average of 11.8 seconds. This test scenario involves the existence of multiple failures which introduce even more delay into the convergence process. This is because multiple failures can partition the network into many isolated parts. When *HelloInterval* has the value 5 seconds, the detection time variation will not exceed the *spfDelay*. In our experiments, the convergence time reaches an average of 8 seconds. Due to the fact that the *HelloInterval* is 10 seconds, the chance that both *spfDelay* and *spfHold* will delay successive routing calculations is higher. Also, because of the value of *HelloInterval*, in some partitions of the network, detecting the failures is much slower. Therefore, the convergence time takes approximately 14 seconds in our test scenarios. All these results demonstrate that the convergence can be delayed by timers because of protocol reaction to single and multiple failures.

VI. CONCLUSION

The aim of this paper is to measure and analyze OSPF convergence in presence of single and multiple failures and their impact on end-to-end traffic. We investigated the routing convergence under five different situations and we can conclude that OSPF converges in about 10 seconds when there is a broken Ethernet connection, and in about 19-20 seconds when there are two broken Ethernet connections. This means that the convergence is greatly delayed when multiple failures occur in the network topology.

According also to the experimental results, the convergence time is influenced by the values of OSPF timers. Larger timer values cause a slower convergence, while smaller timer values ensure a fast convergence. It is recommended to set the timers to smaller values to improve convergence time when dealing with dynamic networks. However, tuning timers require a lot of investigation on specific networks and knowledge about network management. We believe that this is still a wide research area that has just started developing.

 ISBN: 978-1-61208-422-0

ACKNOWLEDGMENT

This work has been funded by the Sectoral Operational Programme Human Resources Development 2007-2013 of Ministry of European Funds through the Financial Agreement POSDRU/159/1.5/S/134398 and by program Partnerships in priority areas – PN II carried out by MEN-UEFISCDI, project No. 47/2014.

REFERENCES

[1] Introduction to routing protocols. [Online]. Available from: http://www.cisco.com/networkers/nw00/pres/2204.pdf 2015.06.02

[2] N. Dubois, M. Capelle, S. Chou, and B. Fondeviole, "The benefits of monitoring routing protocols in live networks", in Proc. of IP Operations and Management, 2004, pp. 9-15.

[3] E. Baccelli, and R. Rajan, "Monitoring OSPF Routing", in Proc. of the IEEE/IFIP International Symposium on Integrated Network Management (IM 20001), 2001, pp. 825-838.

[4] D. Sankar and D. Lancaster, "Routing Protocol Convergence using Simulation and Real Equipment", in Advances in Communications, Networks and Security, vol. 10, 2013, pp. 186-194.

[5] G. Lichtwald, U. Walter, and M. Zitterbart, "Improving Convergence Time of Routing Protocols", in Proc. of the 3rd International Conference on Networks (IEEE-ICN2004), 2004.

[6] T. Griffin and G. Wilfong,"An analysis of BGP convergence properties", in Proc. of ACM SIGCOMM, 1999, pp. 277-288.

[7] J. Moy, "OSPF version 2", Internet Engineering Task Force, Request For Comments (Standards Track) RFC 2328, 1998.

[8] Dijkstra's algorithm. [Online]. Available from: http://en.wikipedia.org/wiki/Dijkstra's_algorithm 2015.03.22

[9] Introduction to Mininet. [Online]. Available from: https://github.com/mininet/mininet/wiki/Introduction-to-Mininet 2015.03.22

[10] Zebra software. [Online]. Available from http://www.gnu.org/software/zebra/ 2015.03.22

[11] P. Francois, C. Filsfils, J. Evans, and O. Bonaventure, "Achieving sub-second IGP convergence in large IP network", Computer Commun.Rev., vol. 35, no. 3, 2005, pp. 35-47.

[12] A. Basu and J. Reicke, "Stability issues in OSPF routing", in Proceedings of 2001 Cnference on Applications, technologies, arhitectures, and protocols for computer communications, 2001, pp. 225-236.

[13] M. Shand and S. Bryant, "IP Fast Reroute Framework", Internet Engineering Task Force, Request For Comments (Standards Track) RFC 5714, 2010.

[14] K. Lakshminarayanan, M. Caesar, M. Rangan, and T. Anderson, "Achieving Convergence-Free Routing using Failure-Carrying Packets", in Proc. of ACM SIGCOMM, 2007, pp. 241-252.

[15] A. Kvalbein, A. F. Hansen, T. Čičić, S. Gjessing, and O. Lysne, "Fast IP network recovery using multiple routing configurations", in Proc. of IEEE INFOCOM, 2006, pp. 1-11.

[16] A. Markopoulou, G. Iannaccone, S. Bhattacharaya, C. Chuah, and C. Diot, "Characterization of failures in an IP backbone", in Proc. of IEEE INFOCOM, 2004, pp. 749-762.

[17] J. S. Foster Jr., "Report of the commission to assess the threat to the United States from electromagnetic pulse (EMP) attack", vol. I, Executive report, 2004, pp. 1-208.

[18] M. Goyal, K. Ramakrishnan, and W. Feng, "Achieving faster failure detection in OSPF networks", in Proc. IEEE International Conference on Communications (ICC2003), 2003, pp. 296-300.

[19] M. Goyal, "Improving Convergence Speed and Scalability in OSPF: A Survey", IEEE Commun. Surveys & Tutorials, vol. 14, no. 2, 2012, pp. 443-463.

[20] S. Banerjee, S. Shirazipourazad, and A. Sen, "Design and Analysis of Networks with Large Components in Presence of Region-Based Faults", in Proc. of IEEE International Conference on Communications (ICC2011), 2011, pp. 1-6.

 ISBN: 978-1-61208-422-0