EMULATING GATEWAY LOAD BALANCING PROTOCOL USING CISCO MODELING LABS

Andrei-Alexandru EVTEI[1], Iustin-Alexandru IVANCIU[1], Daniel ZINCA[1], Virgil DOBROTA[1]
[1]*Communications Department, Technical University of Cluj-Napoca, Romania*
Evtei.Va.Andrei@student.utcluj.ro ; Iustin.Ivanciu@com.utcluj.ro; Daniel.Zinca@com.utcluj.ro
Corresponding author: Virgil Dobrota (e-mail: Virgil.Dobrota@com.utcluj.ro)

Abstract: This paper evaluates Gateway Load Balance Protocol (GLBP) running in Cisco Modeling Labs (CML) emulator, involving auxiliary software tools/commands (iPerf, Cisco TRex Stateless GUI, ping). Officially not performed before for Cisco IOSv images, this study demonstrates the protocol effectiveness and reliability in a controlled experimental setup. We performed the following scenarios: (1) functional testing of GLBP; and (2) traffic measurements. The first one was focused on the redundancy and load-balancing capabilities of this protocol. These tests showed that it successfully reroutes traffic to backup devices, in the event of a primary router failure, ensuring uninterrupted network services. Furthermore, the load-balancing tests confirmed that GLBP efficiently distributes traffic across multiple routers, optimizing resource utilization and preventing any single device from becoming a bottleneck. In the second scenario, traffic throughput and latency were measured under controlled conditions, GLBP-enabled router group had similar performances, as a single IOSv router, in terms of throughput. For latency, the group outperformed the single router setup, maintaining low latency levels even under heavy traffic loads generated by the TRex node.

Keywords: *Cisco IOSv, Cisco TRex, CML, iPerf, GLBP.*

I. INTRODUCTION

Both network administration and security are dependent on traffic monitoring. This means an observation in a continuous manner, the analysis of the data flows, but also the control of them. It allows security assurance, troubleshooting, resource optimization, and understanding of baseline performance. Due to effective traffic management, network administrators could spot anomalies, control transfer rates, and ensure the overall health of the network. This approach is useful only for a short period, i.e., noticing issues and disasters at the time of the incident [1]. To prevent unfavorable events and prepare the network hardware in case of failures, it is necessary to test and study possible scenarios within a controlled environment, without altering the real equipment. This methodology verifies if all devices, protocols, and network configurations function as they should, based on a range of scenarios, capable of testing the limits. According to [2], the controlled environments can be classified as: physical/ hybrid testbeds, emulators, and simulators.

The solution addressed in this paper for an isolated environment that allows testing and validating results in a range of scenarios is based on Cisco Modelling Labs (CML). This is a network emulator for testing, troubleshooting, designing, studying, and validating network topologies, different conditions [3]. It facilitates the correction of potential problems before implementation in the production environment, thus reducing the risk of errors and contributing to the optimization of operational performance. This Cisco's platform offers an easy-to-use working environment, with a wide and varied range of tools designed to make accessible the testing and study of scenarios, simple or complex, being intended for network engineers and those who want to explore the field of networking, The reference scenario for evaluating the functionality of the emulator involves the study of the Gateway Load Balancing Protocol (GLBP), one of the protocols that have not been tested by Cisco in the CML application, not knowing if it works as it should [4]. The GLBP protocol is also Cisco's property, being designed to allow the balanced distribution of network traffic between multiple access gateways, thus ensuring redundancy and recovery in the event of a disaster. Testing the GLBP protocol involves the use of specific traffic monitoring tools, essential to ensure that the network can support the required traffic load without compromising performance. The ping command allows checking connectivity and response time between nodes, helping to identify potential packet loss and high latencies. The open source iPerf application is used to measure available bandwidth and assess connection stability. In addition, the TRex Stateless GUI application is used to generate variable background traffic, equivalent to noise, simulating real-world usage conditions. This allows the evaluation of network behavior under load and identifying potential performance issues under heavy traffic conditions.

The remainder of the paper is structured as follows. Section II presents the related work, whilst Section III is focused on GLBP emulation in Cisco Modeling Labs. Next section presents the experimental results. Finally, Section V concludes the paper and outlines future work.

II. RELATED WORK

Network emulators are essential tools that replicate real-world network conditions, enabling researchers and engineers to test, analyze, and optimize network performance in a controlled and flexible environment. Paper [5] presents a comparison between Virtual Internet Routing Lab (VIRL), the first version of the CML application, and other network emulation software such as Graphical Network Simulator-3 (GNS3) and Emulated

Manuscript received April 11, 2026; revised June 23, 2026

Virtual Environment Next Generation (EVE-NG). One advantage is that it offers better integration with authentic images of Cisco devices, ensuring regular updates aligned with the licensing policies. Furthermore, its foundation is based on OpenStack, an open-source cloud computing platform that enables the creation and management of public and private clouds [6]. This foundation enables environment change and extension, making it easier to create well-scaling multi node clusters. Because of its capacity to accommodate several concurrent emulations, this feature is perfect for educational environments where numerous learners require access to complex network topologies. Related to this advantage, the article [7] presents CML, between other virtual environments, as a solution for virtual labs during the COVID pandemic when remote learning became essential. The need for reliable and realistic network simulations became increasingly important as more universities and institutions shifted to online education. CML offered a solution that allowed teachers and students to conduct in-depth network exercises remotely.

Another study [8] includes CML as a realistic simulation environment, where the topic of cybersecurity is addressed in the context of a healthcare network. The zero-trust security model, which assumes that no traffic within the network is considered trustworthy, was validated using CML. Test scenarios included a traditional network design and a zero-trust-based one, demonstrating improved security for healthcare networks, especially in the context of older and vulnerable equipment. In the article [9], the application was used to validate dynamic network slicing, supporting Internet service providers in managing high traffic. Network slicing allows for the creation of customized logical networks, tailored to the specific requirements of different services. The tests demonstrated that the dynamic network slicing application can effectively handle diverse traffic and service requirements, using CML's emulation capabilities to build realistic network environments. In [10], CML was used together with the Border Gateway Protocol Replay Tool (BRT) v0.2, designed to improve the security and performance of the protocol. This is essential for routing on the Internet, but it is vulnerable to attacks and misconfigurations. It allows the replay of previously captured BGP updates in a controlled environment, using CML to understand the behavior of the protocol and to diagnose the issues. Evaluations have shown that Cisco Modeling Labs can emulate realistic network environments and can facilitate detailed testing of the BGP protocol.

III. EMULATION OF GATEWAY LOAD BALANCE PROTOCOL IN CML

Gateway Load Balance Protocol (GLBP) is a Cisco proprietary first-hop redundancy network protocol, used in topologies that require continuous path for data traffic [11]. It is part of the family of First Hop Routing Protocols (FHRPs), network protocols designed to improve availability and reliability of the network gateway by offering automated failover system using backup devices. Two other FHRP types of protocols are the following: Hot Standby Router Protocol (HSRP), Cisco proprietary, and Virtual Router Redundancy Protocol (VRRP), an open standard protocol defined in [12]. These two offer redundancy by designating a primary device that is responsible for all the traffic within and one or more for backup. GLBP enhances this by offering a different approach, specifically to distribute the traffic load through all the devices that constitute the GLBP group. This not only ensures high availability, but also it maximizes network efficiency by using all equipment without wasting resources. GLBP gathers the group of devices belonging to it by usage of a single virtual IP (vIP) address and some virtual Media Access Control addresses (vMACs). The vIP, different from the physical IP, has the role of the default gateway in the subnet and it needs to be manually configured by the user. vMAC addresses are generated automatically and they used for identification of the equipment in the group.

In GLBP there are Active Virtual Gateway (AVG) and Active Virtual Forwarders (AVFs). AVG is the manager of the group, being responsible for assigning vMACs to group members including themselves. Address Resolution Protocol (ARP) requests are sent by subnet hosts to vIP and it realizes a bound between hosts and vMACs in accordance with the load balancing method used. Active Virtual Forwarders, with a maximum number of four devices for each group, have the responsibility to route the traffic sent to vIP. If the AVG fails, the standby router will become the AVG, and a new standby gateway is elected. If an AVF fails, the AVG reassigns the responsibility for the virtual MAC address (implying traffic through it) to another AVF. Preemption and priorities can be used in AVG and AVFs selection.

Also, there are implemented timers for monitoring the status of the group members and management of vMACs removal in case of failure. For the monitoring status there are the Hello Timer that indicates their presence and status, with 3 seconds on default, and Hold Timer, also known as Dead Timer, which specifies how long a router in the group waits for Hello messages from the others to consider a neighbor inactive, with 10 seconds on default. Regarding management of vMACs, there are the Redirect Timer and Forwarder Timer that ensure smooth transition and continuity in the case when AVG and AVFs are responsible for the traffic from a vMAC belonging to a failed AVF, GLBP achieves load balancing through three methods. Round-Robin method, set by default, distributes the ARP requests received by AVG across all AVFs in turns in a cyclical manner, weighted method that spreads traffic proportionally through AVFs, allowing more powerful router to handle more traffic and host-dependent method, where each host is assigned the same AVF, based on the MAC address of the client.

Starting the implementation, the initial resources of the local host machine were allocated according to the specifications required by the CML application and the virtual images (by the time of experiments, we involved the licensed version 2.7) [1]. Note that the updated version 2.9 (running in 2026) has new features that have no impact for our approach: better user experience, but for more complex simulations, with support for containerization, needed only in the future work. The allocated resources had to be minimal to allow other users to build the same lab without constraints. The experimental environment was built on VMware Workstation Pro, as a Virtual Machine (VM). The core topology included the following devices: (1) three IOSv routers, forming the GLBP configuration; (2) two

IOSvL2 switches, connecting the router group to the external network and to the internal network represented by an Alpine Host device (see Figure 1).

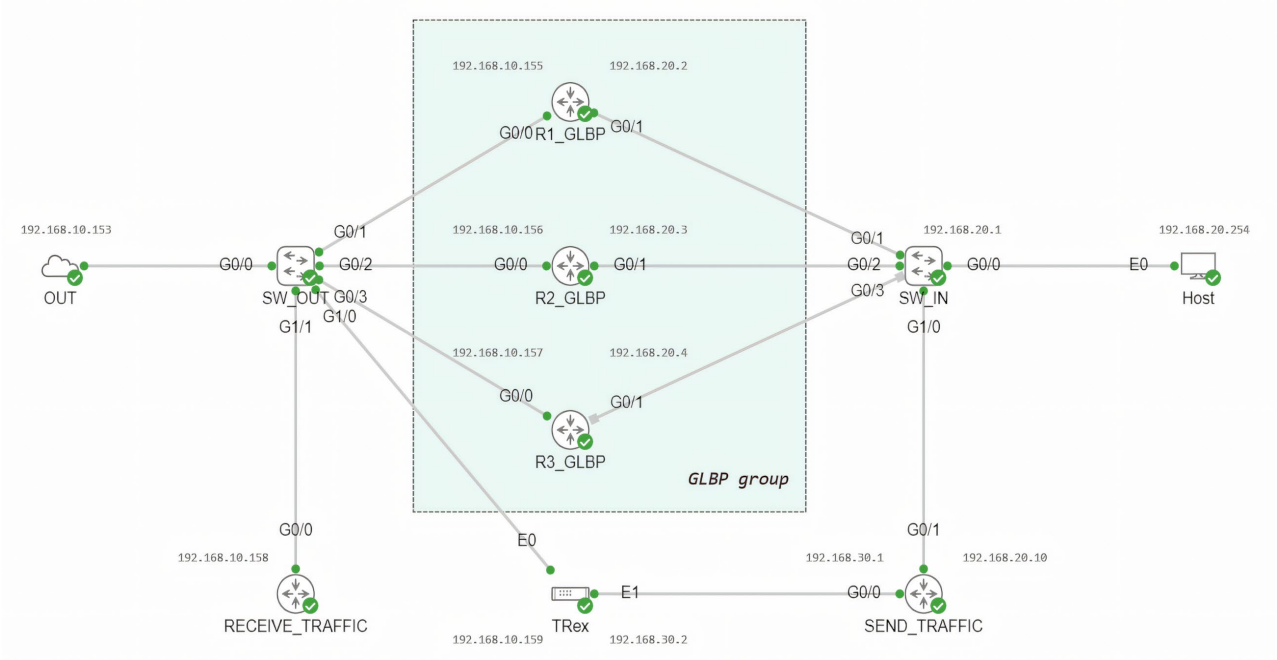

Figure 1. GLBP network topology built in CML [1]

The OUT node, based on the External Connector image, operated in L2 Bridge mode, allowing throughput tests to be performed between the Host node and the local Windows host machine through the GLBP group. Two IOSv routers were also connected to the IOSvL2 switches along with the TRex node to generate data traffic. Variable data traffic was sent through the SEND_TRAFFIC node to the RECEIVE_TRAFFIC node address. The Alpine Linux machine and the local Windows host machine had the iPerf tool installed. This was used for traffic monitoring, allowing a client-server connection to be established to observe the GLBP throughput. It is worth mentioning that, by the time of writing the paper, there was an incompatibility between the CMLv2.7 and the Cisco Trex Stateless GUI application [1]. The issue was resolved by downloading the CMLv2.6.1, which included a compatible TRex image.

A. GLBP Group

All three routers used the IOSv image, the virtual machine implementation of the Cisco IOS equipment. We discuss the configuration for the first router only (being similar for the other two). Thus, we considered 192.168.10.155/24 for Gigabit Ethernet 0/0 outside interface, and 192.168.20.2/24 for Gigabit Ethernet 0/1 inside interface (see Figure 2). The default route was 192.168.10.1/24. Regarding Network Address Translation (NAT) we used Cisco's Port Address Translation (PAT) solution. An access control list allowed all traffic from the addresses in the 192.168.20.0/24 network to go through the outside interface of the router. The configuration commands are displayed in Figure 3.

```
R1_GLBP(config)#int g0/0
R1_GLBP(config-if)#ip address 192.168.10.155
255.255.255.0
R1_GLBP(config-if)#no shutdown
R1_GLBP(config-if)#int g0/1
R1_GLBP(config-if)#ip address 192.168.20.2
255.255.255.0
R1_GLBP(config-if)#no shutdown
R1_GLBP(config-if)#exit
R1_GLBP(config)#ip route 0.0.0.0 0.0.0.0 192.168.10.1
```

Figure 2. Static IP address allocation of R1_GLBP

```
R1_GLBP(config)#int g0/0
R1_GLBP(config-if)#ip nat outside
R1_GLBP(config-if)#int g0/1
R1_GLBP(config-if)#ip nat inside
R1_GLBP(config-if)#access-list 1 permit 192.168.20.0
0.0.0.255
R1_GLBP(config-if)#exit
R1_GLBP(config)#ip nat inside source list 1 interface
g0/0 overload
```

Figure 3. PAT configuration of R1_GLBP router

The target router configuration of the project (see Figure 4) enables the router to use the GLBP protocol in the inside network. To have equivalent configurations on all three routers, the command required same group number (1 in this case) and the same virtual IP address (192.168.20.1/24).

As observed in Figure 1, there was a single Active Virtual Gateway (AVG), represented by the router R2_GLBP 192.168.20.3/24, with a Standby router R3_GLBP with 192.168.20.4/24, and three Active Virtual Forwarders (AVFs) with specific MAC addresses. Of the last four octets of the MAC, the first two octets represented

the group number and the last two represented the AVF number.

```
R1_GLBP(config)#int g0/1
R1_GLBP(config-if)#glbp 1 ip 192.168.20.1
R1_GLBP(config-if)#do sh glbp brief
Interface Grp Fwd Pri State   Address
Active router   Standby router
Gi0/1     1   -   100 Listen 192.168.20.1
192.168.20.3  192.168.20.4
Gi0/1     1   1   -   Active 0007.b400.0101
local     -
Gi0/1     1   2   -   Listen 0007.b400.0102
192.168.20.3  -
Gi0/1     1   3   -   Listen 0007.b400.0103
192.168.20.4  -
```

Figure 4. GLBP protocol configured on R1_GLBP

Note that other GLBP configurations (preemptions, priorities and timers' values) were set as default.

B. Alpine Host

This Alpine host was connected to the inside switch, as a terminal node for traffic monitoring of the network topology. It involved iPerf in client mode, at one end, and iPerf server mode on the hosting Windows machine at the other end. The transfer rates through the GLBP routers were measured too. We conducted other experiments with ping from the Alpine node to the RECEIVE_TRAFFIC node. The network configuration for the Alpine Host is presented in Figure 5. In the Figure 6, iPerf version 3 was installed on the Linux host to evaluate the throughput from the client's perspective.

```
auto lo
iface lo inet loopback
auto eth0
iface eth0 inet static
    address 192.168.20.2/24
    gateway 192.168.20.1
    hostname alpine_host
```

Figure 5. Alpine node static IP address configuration

```
inserthostname-here:~$ sudo apk add iperf3
fetch https://dl-
cdn.alpinelinux.org/alpine/v3.19/main/x86_64/APKINDEX.t
ar.gz
fetch https://dl-
cdn.alpinelinux.org/alpine/v3.19/community/x86_64/APKIN
DEX.tar.gz
(1/2) Installing iperf3 (3.16-r0)
(2/2) Installing iperf3-openrc (3.16-r0)
Executing busybox-1.36.1-r15.trigger
OK: 1152 MiB in 460 packages
```

Figure 6. iPerf3 installation on Linux host

C. TRex Traffic Generator

Cisco's TRex traffic generator was used within the TRex node, connected to the pair of SEND_TRAFFIC and RECEIVE_TRAFFIC router nodes to forward the generated traffic and to the SW_OUT node to acquire Internet connectivity. Its image was an Alpine Linux VM that managed to operate in stateless mode, and the IP address was 192.168.10.159 (as in Figure 7).

D. Local Host

The local Windows host had two roles, beside hosting CML: (1) to provide the TRex Stateless GUI software for the TRex node; (2) to run the iPerf server for throughput measurements, as shown in Figure 8.

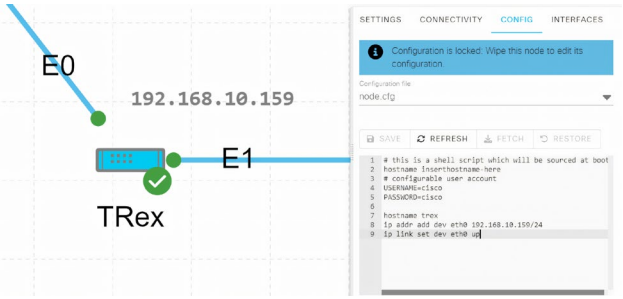

Figure 7. TRex node networking configuration

```
C:\Users\uclabs>cd Downloads
C:\Users\uclabs\Downloads>cd iperf3.1.1_32
C:\Users\uclabs\Downloads\iperf3.1.1_32>iperf3.exe -s
-----------------------------------------------
Server listening on 5201
-----------------------------------------------
```

Figure 8. iPerf3 running on local Windows host

IV. EXPERIMENTAL RESULTS

We conducted two experiments: *Experiment 1: GLBP Functions Test*, having two scenarios: (1) redundancy in case of link failure; and (2) load-balance capabilities; *Experiment 2: Traffic Monitoring Test,* having two scenarios: (1) traffic throughput with iPerf; (2) latency with triggered ping commands. In all cases a comparison was made between a single IOSv router and a GLBP group of IOSv routers, in the presence of controlled background traffic generated by the TRex node. The traffic profile for the following experiments is presented in Figure 9.

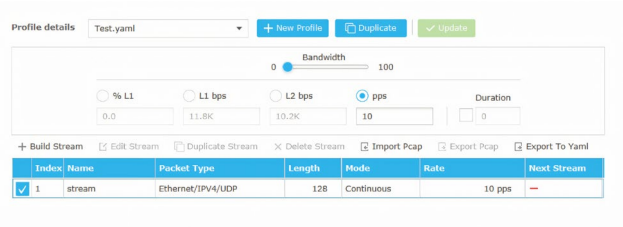

Figure 9. Traffic profile of the TRex

There were 128-byte UDP segments involved, as the background traffic did not need a connection-oriented approach. There were continuous transmissions without using patterns, the only parameter that was modified being the rate of packets per second [pps].

Experiment 1/ Scenario 1: GLBP Functions Test with Redundancy in Case of Link Failure

The purpose of this scenario was to verify the redundancy and recovering capability of the GLBP group after a link failure happens on one of the AVFs responsible for traffic forwarding at that moment. The following component elements were involved:

- Alpine Node: it sends ping requests to the RECEIVE_TRAFFIC node.

- R1_GLBP: this is the router through which Alpine Node sends ping requests.
- R2_GLBP and R3_GLBP routers: both in Listening state; R2_GLBP was also the AVG.
- RECEIVE_TRAFFIC node: it receives ping requests from the Alpine Node.

We presumed that after the link failure on the R1_GLBP link, the R2_GLBP, being the AVG, will give the responsibility for the R1_GLBP vMAC to the next router in the round-robin configuration, specifically to itself. After the link recovery, R1_GLBP reclaimed the old vMAC, behaving as previously. First, the vMAC of the AVFs was checked in the console, as in Figure 10.

```
R1_GLBP#show glbp brief
Interface Grp Fwd Pri State   Address
Active router  Standby router
Gi0/1     1   -   100 Listen  192.168.20.1
192.168.20.3  192.168.20.4
Gi0/1     1   1   -   Active  0007.b400.0101
local         -
Gi0/1     1   2   -   Listen  0007.b400.0102
192.168.20.3  -
Gi0/1     1   3   -   Listen  0007.b400.0103
192.168.20.4  -
```

Figure 10. vMAC check in R1_GLBP console

After that, we verified the ARP table of the Alpine Linux machine to discover what vMAC was used by the GLBP group (see Figure 11).

```
inserthostname-here:~$ arp -a
? (192.168.20.3) at 52:54:00:0c:42:3b [ether] on eth0
? (192.168.20.2) at 52:54:00:18:f5:c8 [ether] on eth0
? (192.168.20.1) at 00:07:b4:00:01:01 [ether] on eth0
? (192.168.20.10)at 52:54:00:1f:23:80 [ether] on eth0
```

Figure 11. ARP table of the Alpine node

Next, the ping command was triggered towards the RECEIVE_TRAFFIC node, and the link failure (see Figure 12) was produced by disconnecting the link.

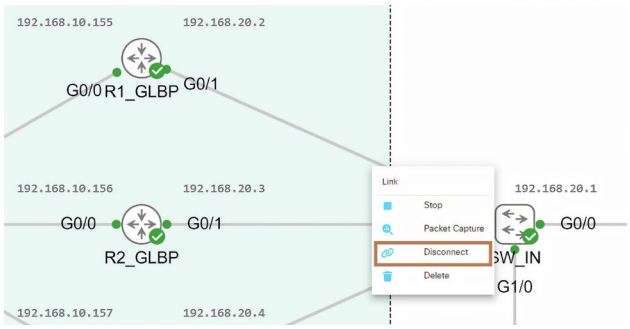

Figure 12. Link failure

Later, the behavior of the Internet Control Message Protocol (ICMP) packets transmitted through ping were observed in the traffic captured through link between SW_OUT and RECEIVE_TRAFFIC nodes, as in Figure 13. After 7.69 seconds, ICMP messages lasted until the second 16.69, when the ping transfer continued as previously. It was observed that the interval between the two packets was approximately the same as the default Hold Timer value of 10 seconds. One remark was that the switching between the failed AVF and the new AVF was smooth, being even less than the Hold Timer value.

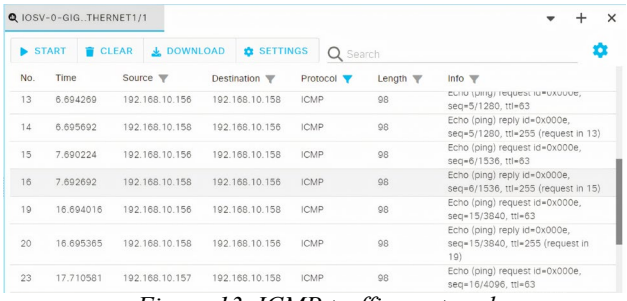

Figure 13. ICMP traffic captured

From the R1_GLBP point of view, the status immediate after the failure was observed in Figure 14. It showed that status details were unknown, as the router was not able anymore to track the GLBP group management packets.

```
Interface Grp Fwd Pri State   Address
Active router Standby router
Gi0/1     1   -   100 Init    192.168.20.1
unknown       unknown
Gi0/1     1   1   -   Init    0007.b400.0101
unknown       -
Gi0/1     1   2   -   Init    0007.b400.0102
unknown       -
Gi0/1     1   3   -   Init    0007.b400.0103
unknown       -
```

Figure 14. R1_GLBP perspective after the link failure

After the link recovery, R1_GLBP could receive again GLBP packets from the other members but for the moment, it operated in Listen State, unable to forward packets as shown in Figure 15. One remark: the responsibility for its old vMAC was given to the R3_GLBP router, the one with the IP address of 192.168.20.4/24.

```
R1_GLBP#sh glbp brief
Interface Grp Fwd Pri State   Address
Active router Standby router
Gi0/1     1   -   100 Listen 192.168.20.1
192.168.20.3  192.168.20.4
Gi0/1     1   1   -   Listen 0007.b400.0101
192.168.20.4  -
Gi0/1     1   2   -   Listen 0007.b400.0102
192.168.20.3  -
Gi0/1     1   3   -   Listen 0007.b400.0103
192.168.20.4  -
```

Figure 15. R1_GLBP state after recovery

After a while, the old vMAC address was given back to the R1_GLBP, reclaiming the Active state and becoming AVF (see Figure 16).

```
R1_GLBP#sh glbp brief
Interface Grp Fwd Pri State   Address
Active router  Standby router
Gi0/1     1   -   100 Listen 192.168.20.1
192.168.20.3   192.168.20.4
Gi0/1     1   1   -   Active 0007.b400.0101
local         -
Gi0/1     1   2   -   Listen 0007.b400.0102
192.168.20.3  -
Gi0/1     1   3   -   Listen 0007.b400.0103
192.168.20.4   -
```

Figure 16. R1_GLBP state after a while

We concluded that the redundancy capability of the GLBP has been proven, the assumptions being just partially true. This is because vMAC was given to the R3_GLBP instead of R2_GLBP. We explain it that, in case of a failed AVF, the vMAC belonging to it was given to the device with the highest IP address in case of a weighting value tie, even if the load balancing algorithm was enabled.

Experiment 1/ Scenario 2: GLBP Functions Test with Load Balancing

This second scenario involved the Load Balancing (LB) capability of the GLBP, an extra functionality of the First Hop Redundancy Protocols (FHRPs). We involved the same entities as the previous scenario, with the same role, except that all three routers were treated equally this time. The LB algorithm used was the well-known round robin, where all ARP requests received by the AVG were distributed across all AVFs in turns in a cyclic manner. Alpine's first connection with the GLBP gateway involved the use of ARP to obtain the vMAC of the gateway by knowing its virtual IP. After the response given by the AVG, the vMAC was stored in the Alpine host's ARP table and it was used every time the host wanted to communicate through gateway. This implied that every time the same AVF with the related vMAC forwarded the traffic sent by the Alpine machine. This impediment could be solved by clearing the host ARP table to obtain the new vMAC, in a sequential manner provided by the round robin algorithm.

During the experiment, this method to receive different MAC addresses was approached due to the Linux command `sudo ip neighbor flush all`. This removed all the discovered neighbors, i.e., the list of the MAC addresses with their corresponding IP. The command `arp -a` displayed the ARP table. As in Figures 17a, 17b and 17c, every time the list had been removed and the host had sent ping messages to the IP address of the RECEIVE_TRAFFIC node, a new MAC address appeared on the table, replacing the previous one in a sequential order.

```
inserthostname-here:~$ arp -a
? (192.168.20.1) at 00:07:b4:00:01:01 [ether] on eth0
inserthostname-here:~$ sudo ip neigh flush all
192.168.20.1 dev eth0 lladdr 00:07:b4:00:01:01 used 0/0/0
probes 4 STALE

*** Round 1, deleting 1 entries ***
*** Flush is complete after 1 round(s) ***
inserthostname-here:~$ ping 192.168.10.158
PING 192.168.10.158 (192.168.10.158): 56 data bytes
64 bytes from 192.168.10.158: seq=0 ttl=42 time=17.245
ms
^C
--- 192.168.10.158 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 17.245/17.245/17.245 ms
```

Figure 17a. Load balancing capability of GLBP (MAC: 00:07:b4:00:01:01)

The order of the addresses observed were R1_BGLP, R2_BGLP and R3_BGLP, with the periodical continuity of the sequence.

```
inserthostname-here:~$ arp -a
? (192.168.20.1) at 00:07:b4:00:01:02 [ether] on eth0
inserthostname-here:~$ sudo ip neigh flush all
192.168.20.1 dev eth0 lladdr 00:07:b4:00:01:02 ref 1 used
0/0/0 probes 4 REACHABLE

*** Round 1, deleting 1 entries ***
*** Flush is complete after 1 round(s) ***
inserthostname-here:~$ ping 192.168.10.158
PING 192.168.10.158 (192.168.10.158): 56 data bytes
64 bytes from 192.168.10.158: seq=0 ttl=42 time=15.633
ms
^C
--- 192.168.10.158 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 15.633/15.633/15.633 ms
```

Figure 17b. Load balancing capability of GLBP (MAC: 00:07:b4:00:01:02)

```
inserthostname-here:~$ arp -a
? (192.168.20.1) at 00:07:b4:00:01:03 [ether] on eth0
[…]
```

Figure 17c. Load balancing capability of GLBP (MAC: 00:07:b4:00:01:03)

We concluded that the load balancing of the GLBP was fully functional, further experiments envisaged would be related to other load balancing mechanisms, such as the weighted and host-dependent methods.

Experiment 2/ Scenario 1: Traffic Monitoring Test with Throughput Measurement

This scenario regarding the traffic monitoring was based on the throughput measurement through the GLBP router group. First, the transfer rate was measured in a baseline setup, where only one router from the GLBP was active, having the same behavior as if no GLBP has been configured. The components of this baseline setup were the following:

- Alpine host and Windows host: establish the client-server connection using iPerf3
- R1_GLBP: the router through which the throughput is measured
- TRex: generates background traffic through R1_GLBP
- SEND_TRAFFIC: sends the traffic through virtual IP address of the GLBP group
- RECEIVE_TRAFFIC: receive the traffic generated by TRex

In the second step, there were two main differences. (1) R2_GLBP was active and it formed together with R1_GLBP a two-router GLBP group; (2) A TCP connection made by the two hosts had been moved on the second router, while the background traffic generated by TRex remained on the first router. Thus, we supposed that the defined background traffic in the beginning of this section was measured five times, in a period of 60 seconds with a variable traffic intensity. At the end of the measurement, iPerf computed the average throughput. According to Cisco, the performance of the IOSv was limited in CMLv2.7 to 2.8Mbps for traffic passing through one router and 2.4Mbps for two bounded routers [3]. The experimental results are presented in Table 1 (a single router approach) and in Table 2 (two-router approach).

	10pps [Mbps]	100pps [Mbps]	500pps [kbps]	1000pps [kbps]	2000pps [kbps]
Test 1	1.61	1.45	612	95	140.0
Test 2	1.58	1.44	629	72	90.0
Test 3	1.60	1.45	664	116	72.5
Test 4	1.60	1.44	612	86	105.0
Test 5	1.59	1.43	647	78	88.6
Average	**1.59**	**1.44**	**632.8**	**89.4**	**99.2**

Table 1: Throughput measurement for a single router

	10pps [Mbps]	100pps [Mbps]	500pps [kbps]	1000pps [kbps]	2000pps [kbps]
Test 1	1.59	1.45	664	157	157
Test 2	1.58	1.42	682	150	147
Test 3	1.60	1.46	629	151	144
Test 4	1.61	1.45	647	105	157
Test 5	1.59	1.44	647	158	175
Average	**1.59**	**1.44**	**653.8**	**144.2**	**156**

Table 2: Throughput measurements for a GLBP group

Also, a visual comparison between the two setups is presented in Figure 18.

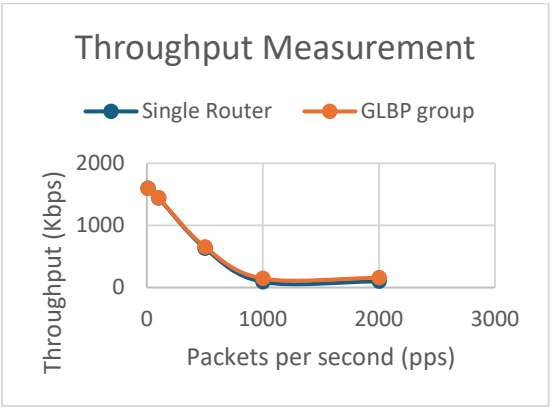

Figure 18. Throughput measurement comparison between the two setups

The results showed that the throughput between the two setups did not differ significantly, except at the end of the graph. Thus, here the GLBP group seemed to be slightly better than single router only. However, the difference is insignificant, approximately 50 kbps. Further experiments can be carried out with other devices, with higher limitations than the ones of the devices included in Cisco CMLv2.7.

Experiment 2/ Scenario 2: Traffic Monitoring Test with Latency Measurement

The second scenario about traffic monitoring was related to the amount of delay the ICMP packets experienced in a high-traffic environment. This was built the same way as the previous, except the Alpine device which sent ping requests toward the same node as TRex, to the RECEIVE_TRAFFIC router. Figure 19 shows a small-scale approach of ping usage for monitoring the RTT. The flag `-w` from the Linux command line represented how many seconds the ping command lasts. In the main scenario, the ping was set for 60 seconds. Same as previous, for both setups, five tests were done by taking the average RTT value, compared in five different cases where the traffic intensity produced by UDP segments increased gradually. The experimental results for the single router setup are presented in Table 3 and the results for the GLBP group router setup are presented in Table 4. The comparison between the two setups is represented graphically in Figure 20.

```
inserthostname-here:~$ ping 192.168.10.158 -w 3
PING 192.168.10.158 (192.168.10.158): 56 data bytes
64 bytes from 192.168.10.158: seq=0 ttl=42 time=7.640 ms
64 bytes from 192.168.10.158: seq=1 ttl=42 time=7.317 ms
64 bytes from 192.168.10.158: seq=2 ttl=42 time=6.732 ms

--- 192.168.10.158 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 6.732/7.229/7.640 ms
```

Figure 19. Ping command

	10pps [ms]	100pps [ms]	500pps [ms]	800pps [ms]	1000pps [ms]
Test 1	7.974	8.220	10.006	11.901	2,579
Test 2	7.725	9.306	10.400	12.199	2,573
Test 3	8.444	9.010	9.424	14.778	2,530
Test 4	8.255	8.896	9.581	13.136	2,462
Test 5	7.706	7.922	9.619	11.973	2,532
Average	**8.020**	**8.670**	**9.806**	**12.797**	**2,535**

Table 3: Latency measurements for a single router

	10pps (ms)	100pps (ms)	500pps (ms)	800pps (ms)	1000pps (ms)
Test 1	8.380	10.803	10.439	12.636	11.250
Test 2	8.690	9.862	9.401	11.333	13.207
Test 3	7.456	9.573	10.846	11.348	13.535
Test 4	8.675	9.786	12.190	12.246	12.411
Test 5	9.116	8.556	10.676	11.764	12.853
Average	**8.463**	**9.716**	**10.710**	**11.865**	**12.651**

Table 4: Latency measurement for a GLBP group

Compared to the previous scenario, in this one the differences were significant, as is observed in Figure 20.

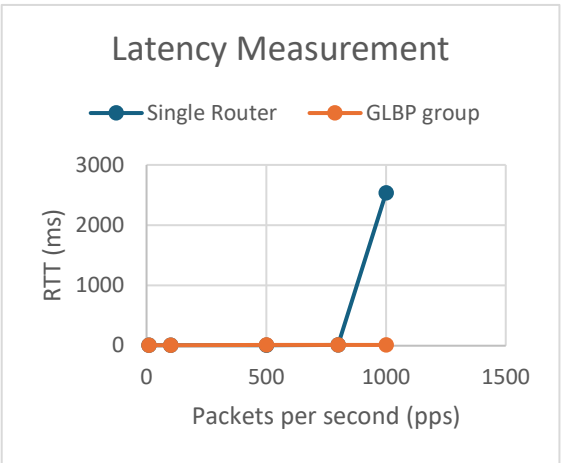

Figure 20. Latency measurement for a GLBP group

After a certain threshold, the single router setup failed, and it did not cope with the traffic. On the other hand, in the GLBP group, for the same amount of background traffic, the ICMP packets traveled uninterrupted toward the destination, same as for the TRex UDP datagrams. However, at the beginning of the graph, better observed in the two reference tables, the single router setup performed better than the GLBP setup, the difference being between 0.4 and 1.1 milliseconds.

CONCLUSIONS AND FUTURE WORK

The study demonstrated protocol effectiveness and reliability in a controlled experimental setup. Despite the limitations of the node throughput within the CML environment, the experiments confirmed that GLBP is functioning as it was intended (i.e., providing redundancy and load-balancing capabilities). Future work will be focused on involving the latest licensed version of CML (v2.9 or above) and exploring new protocols running mainly in IPv6, with devices that have higher throughput capabilities, in more complex scenarios.

ACKNOWLEDGMENT

An initial expanded version of this work was presented by A.A Evtei as B.Sc. Thesis in Telecommunications Technologies and Systems, at Technical University of Cluj-Napoca in July 2024.

REFERENCES

[1] A.A. Evtei, “Emulating Gateway Load Balancing Protocol Using Cisco Modelling Labs “, B.Sc. Thesis, Technical University of Cluj-Napoca, July 2024.

[2] S. V. Tagliacane, P. W. C. Prasad, G. Zajko, A. Elchouemi, and A. K. Singh, “Network simulations and future technologies in teaching networking courses: Development of a laboratory model with Cisco Virtual Internet Routing Lab (Virl),” in 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), Chennai, India: IEEE, Mar. 2016, pp. 644–649. doi: 10.1109/WiSPNET.2016.7566212.

[3] “Cisco Modeling Labs,” Cisco Systems, 2026. [Online]. Available: ttps://www.cisco.com/c/en/us/products/cloud-systems-management/modeling-labs/index.html.

[4] “IOSv-Cisco Modeling Labs v2.7,” Cisco Systems, 2026, [Online]. Available: https://developer.cisco.com/docs/modeling-labs/iosv/.

[5] S. Reissmann, S. Rieger, C. Seifert, and C. Pape, “Using Cisco VIRL and GNS3 to Improve the Scale-out of Large Virtual Network Testbeds in Higher Education,” 2018.

[6] “Open-Source Cloud Computing Infrastructure,” OpenStack, 2026. [Online]. Available: https://www.openstack.org/

[7] G. Tsochev, “Research on Web Applications for Remote Laboratory Exercises on Computer Networks,” in 2021 International Conference on Information Technologies (InfoTech), Varna, Bulgaria: IEEE, Sep. 2021, pp. 1–4. doi: 10.1109/InfoTech52438.2021.9548329.

[8] D. Tyler and T. Viana, “Trust No One? A Framework for Assisting Healthcare Organisations in Transitioning to a Zero-Trust Network Architecture,” Applied Sciences, vol. 11, no. 16, p. 7499, Aug. 2021, doi: 10.3390/app11167499.

[9] L.M. Moreira Zorello, S. Troia, S. Giannotti, R. Alvizu, S. Bregni, and G. Maier, “On the Network Slicing for Enterprise Services with Hybrid SDN,” in 2020 IEEE Latin-American Conference on Communications (LATINCOM), Santo Domingo, Dominican Republic: IEEE, Nov. 2020, pp. 1–6. doi: 10.1109/LATINCOM50620.2020.9282318

[10] B. Al-Musawi, R. Al-Saadi, P. Branch, and G. Armitage, “BGP Replay Tool (BRT) v0.2,” Swinburne University of Australia, 2017.

[11] “GLBP-Gateway Load Balancing Protocol,” Cisco Systems, 2026, [Online]. Available: https://www.cisco.com/en/US/docs/ios/12_2t/12_2t15/feature/guide/ft_glbp.html

[12] A. Lindem, A. Dogra, “Virtual Router Redundancy Protocol (VRRP) Version 3 for IPv4 and IPv6”, RFC 9568, IETF, 2024, Available: https://datatracker.ietf.org/doc/html/rfc9568.