Network Management
Simulation of Vehicular Ad-hoc Network using Mininet-Wifi

Angeliki Kalamari

September 2018

1 Introduction

This simulation took place at Ubuntu VirtualBox using python 2.7. The goal of this project is to recreate the given experiments as presented at the papers below:

· From Theory to Experimental Evaluation: Resource Management in Software-Defined Vehicular Networks, Ramon Fontes, Christian Esteve Rothenberg et. al.

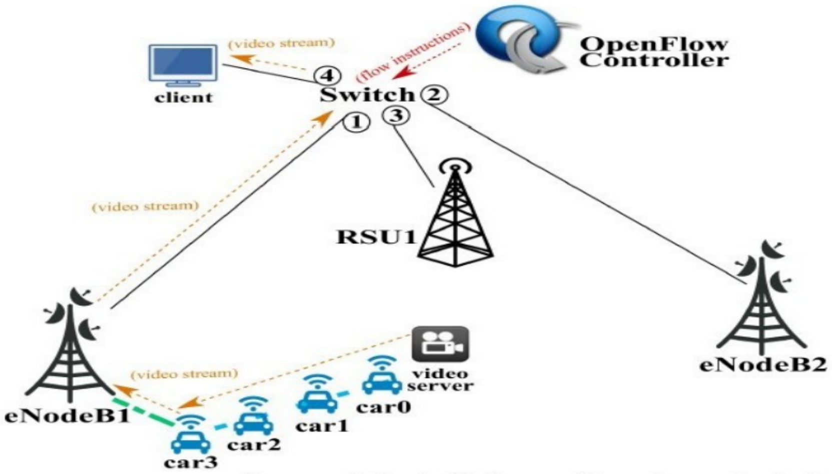
· Mininet-WiFi: A Platform for Hybrid Physical-Virtual Software-Defined Wireless Networking Research, Ramon Fontes, Christian Esteve Rothenberg.

The execution for the first stage of this project can done by using the command **sudo python experiment1.py** and for the second stage by the command **sudo python experiment2.py**

2 First Experiment

It's an experiment on resilient resource aggregating mechanisms in a SDN-based VANET scenario, where the SDN/OpenFlow controller is responsible for managing all emulated nodes.

2.1 Phase One

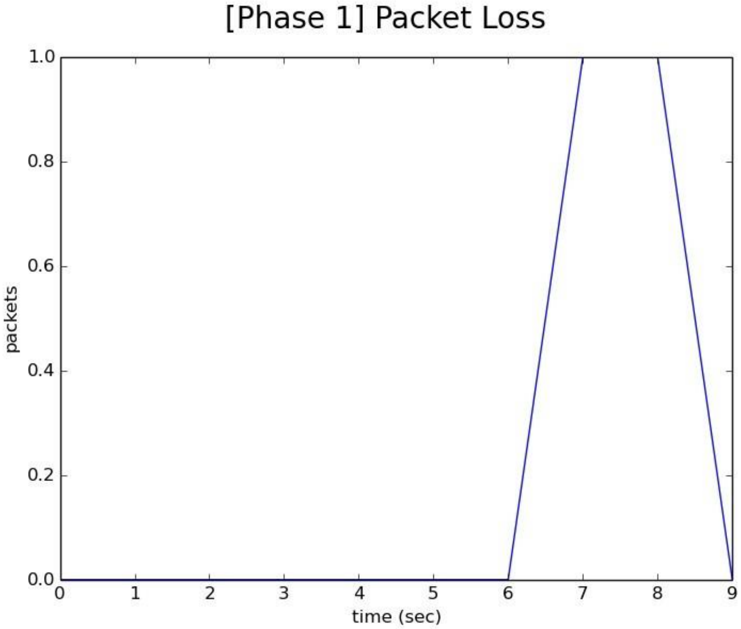


rule: *match (switch)*: in_port 1 *action*: output: 4

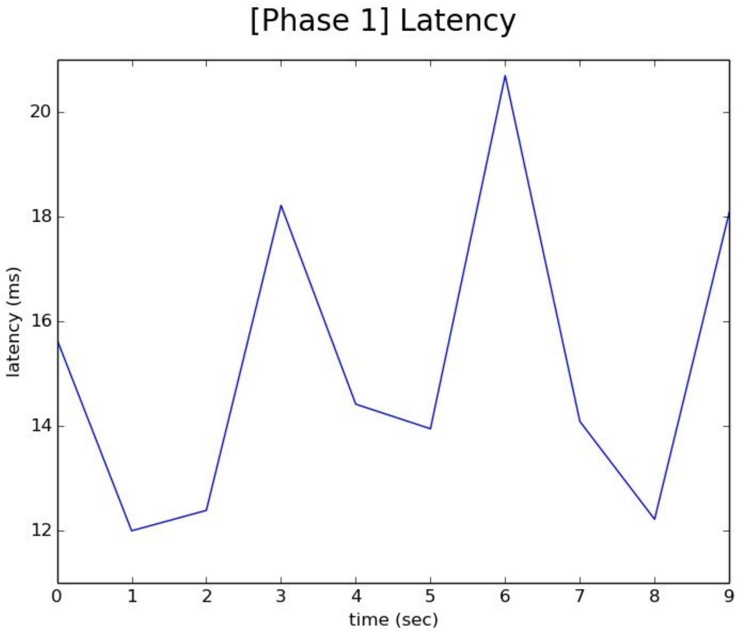
(a) Phase 1

The results after conducting the experiment were:

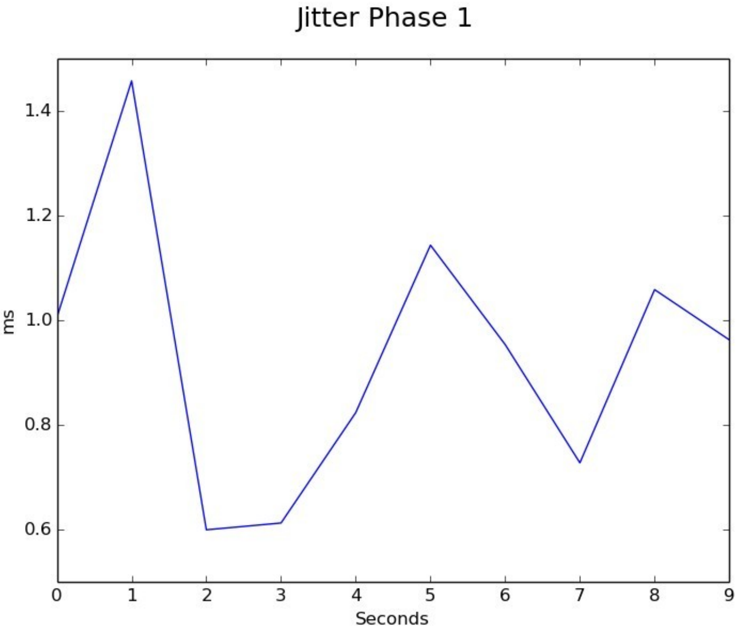
· **PacketLoss:**



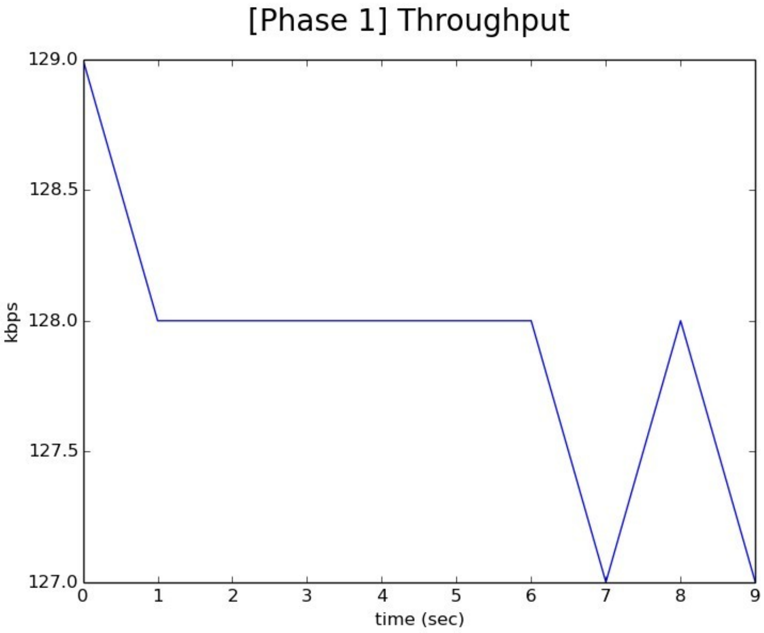
· **Latency:**



· **Jitter:**

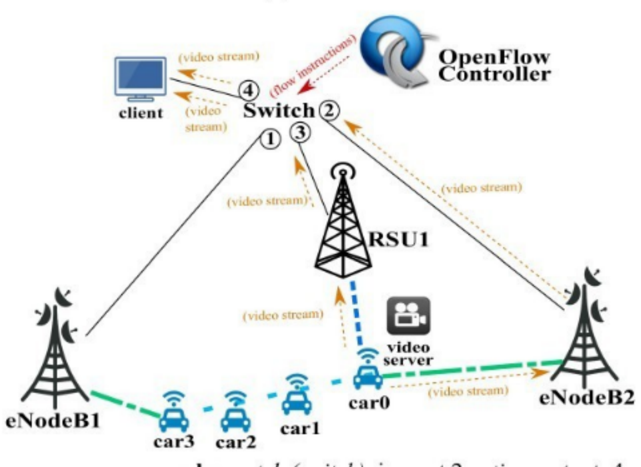


· **Throughput:**



Concluding for Phase One, it was observed that Latency is high and not stable. What causes that is the fact that the signal bounces from one vehicle (car) to the other. Furthermore, there is certain reduction in Throughput at the time when car3 connects with eNodeB1, which happens because their connection has lesser Throughput that the connection between the vehicles.

2.2 Phase Two

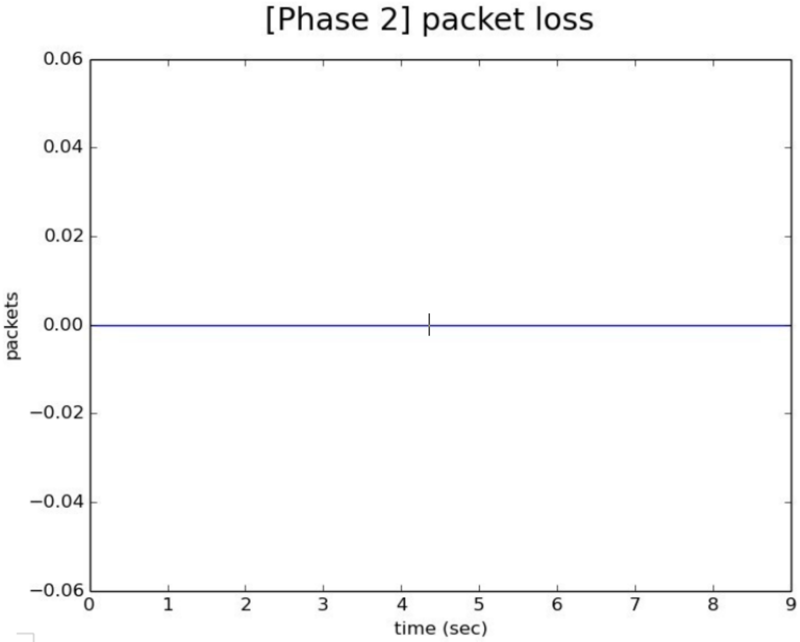


rule: *match (switch)*: in_port 2 *action*: output: 4
rule: *match (switch)*: in_port 3 *action*: output: 4

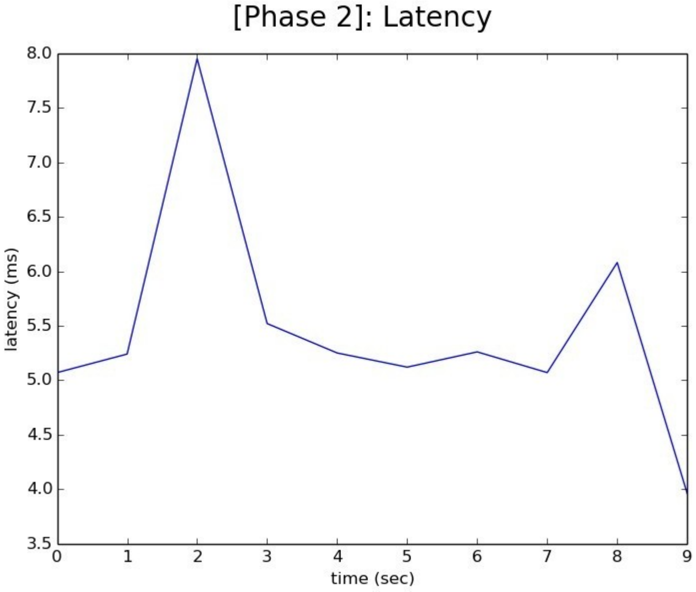
(b) Phase 2

The results after conducting the experiment were:

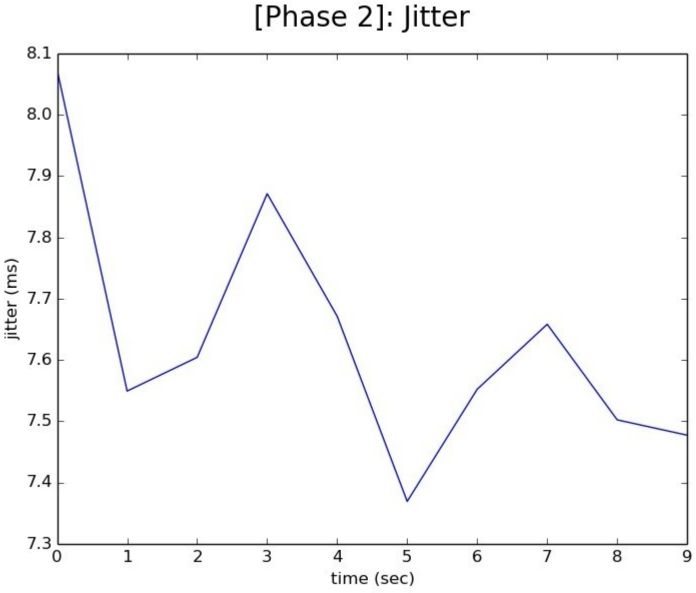
· **PacketLoss:**



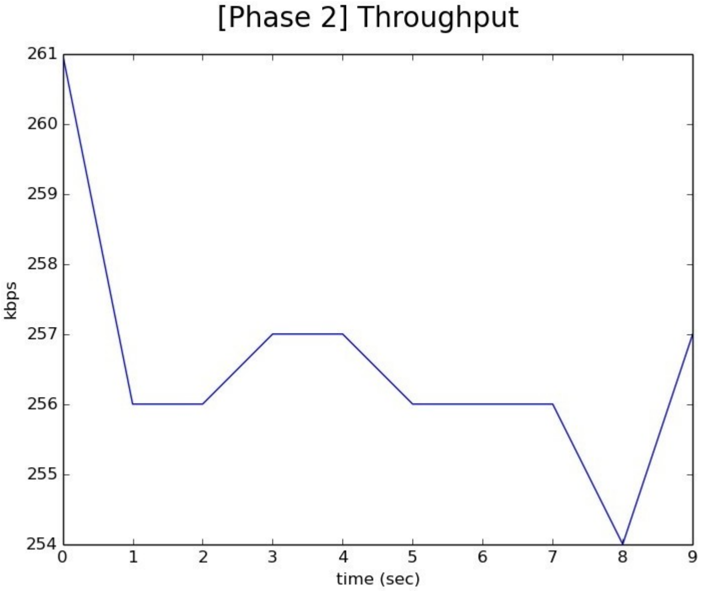
· **Latency:**



· **Jitter:**

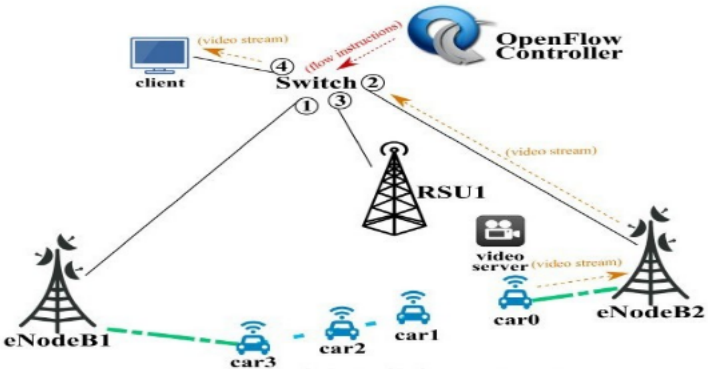


·Throughput:



Concluding, Throughput is now double, because of the fact that vehicle car0 broadcasts the video to RSU1 and to eNodeB1. As a consequence, sending the same packet two times, it's much less likely to not reach its' destination, which is the client. Therefore, the graph of PacketLoss makes absolute sense.

2.3 Phase Three

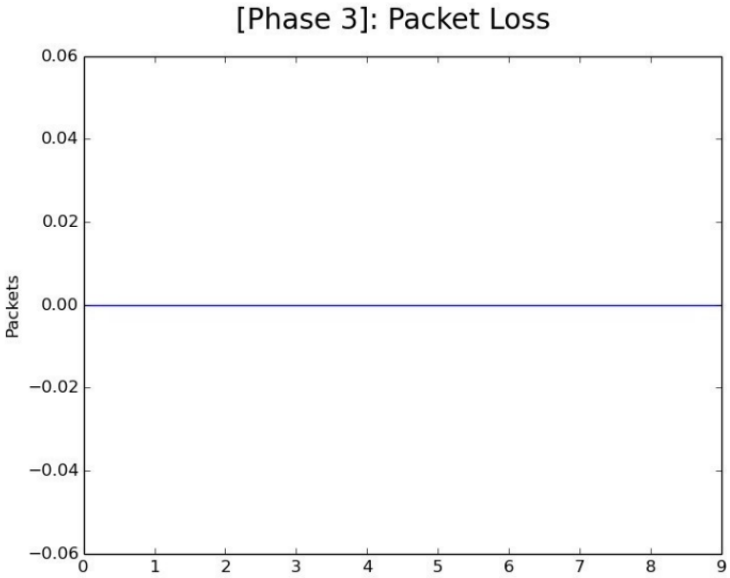


rule: *match (switch)*: in_port 2 *action*: output: 4

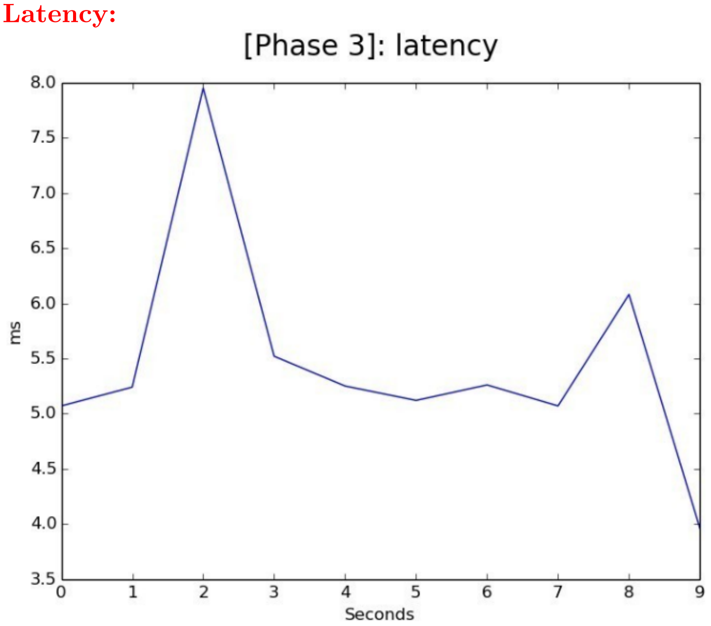
(c) Phase 3

The results after conducting the experiment were:

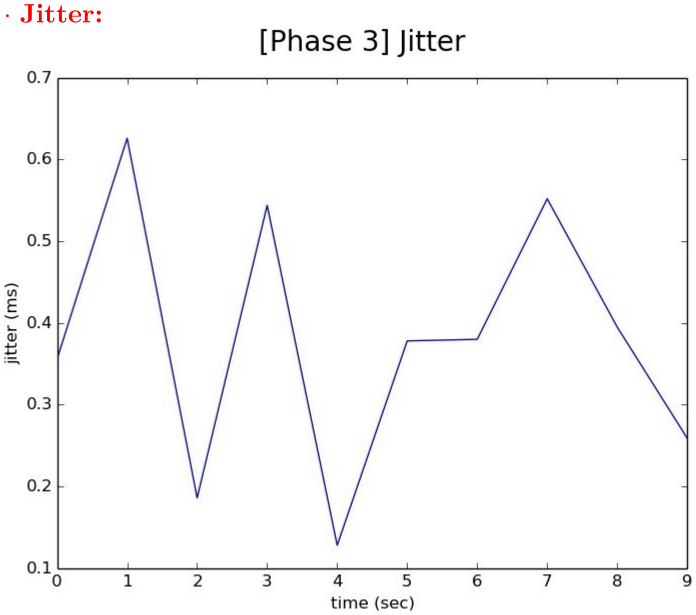
· **PacketLoss:**



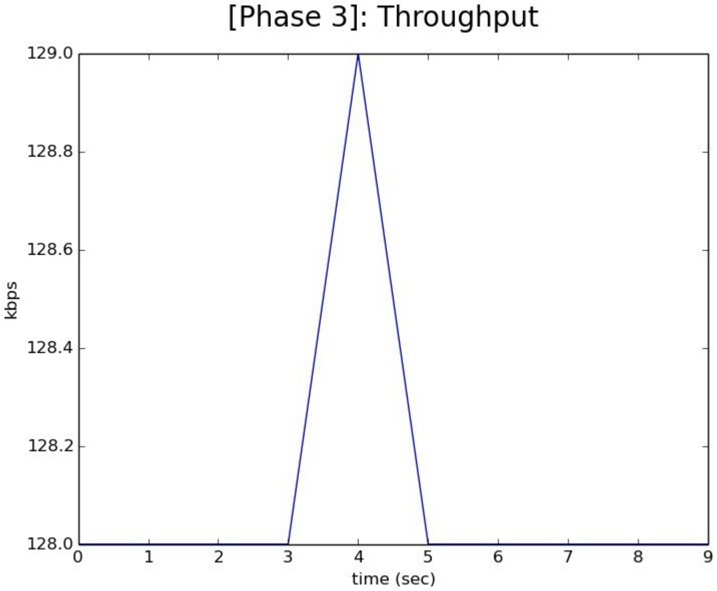
· **Latency:**



· **Jitter:**



·Throughput:

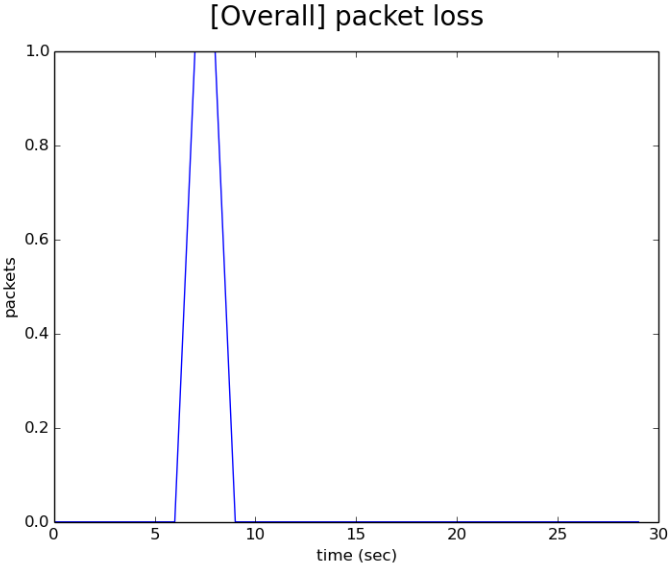


Concluding, *Third Phase* has the same Latency with the *Second Phase*, but much lesser Throughput, because vehicle car0 is broadcasting only to eNodeB1. Regarding the Jitter, it's almost zero, because of the fact that car0 doesn't broadcast duplicate packets.

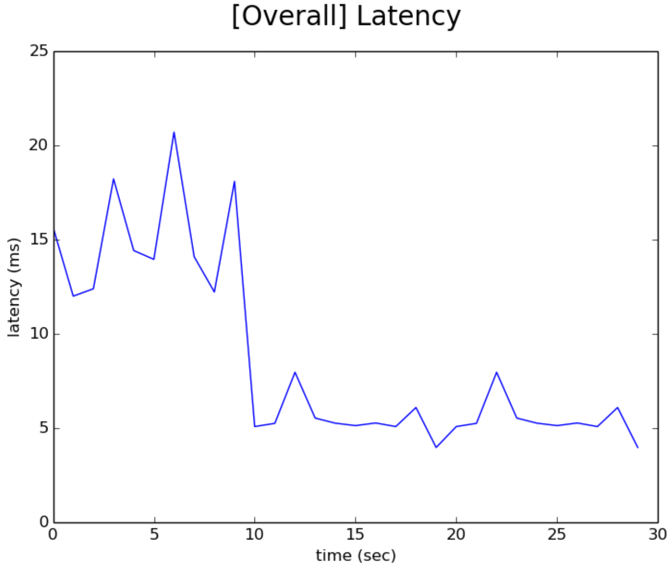
2.4 Total representation

Bellow we can observe the overall differences between the metrics of all three phases of the experiment. It's evident that *Thrid Phase* yielded the best results.

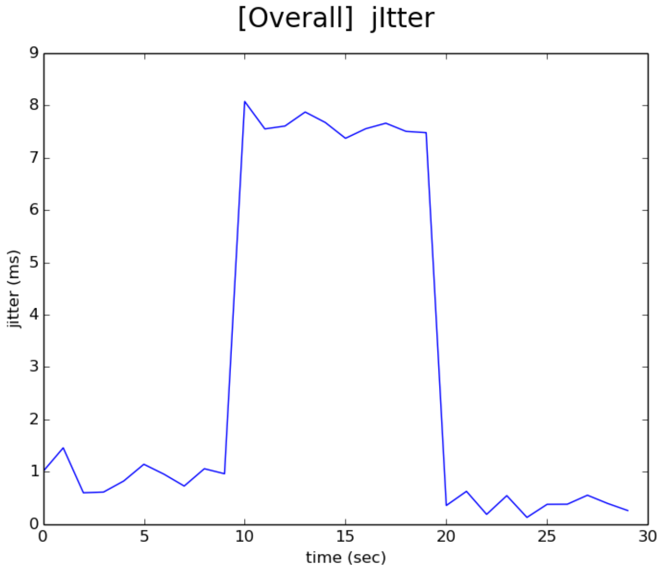
· **PacketLoss:**



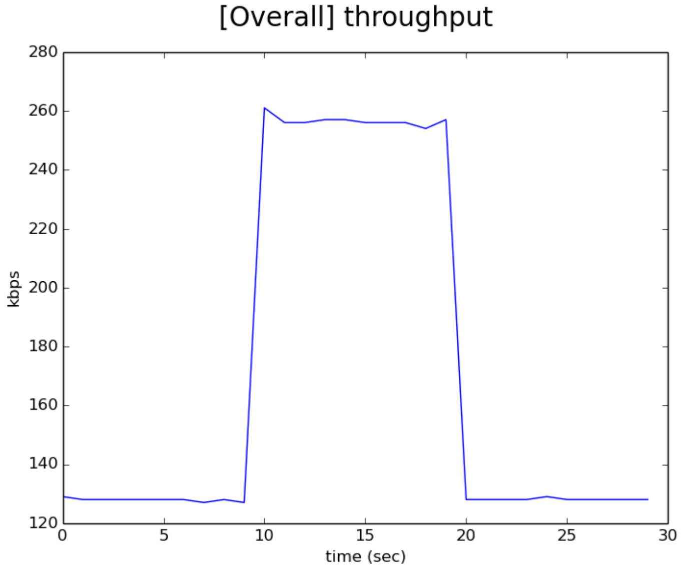
· **Latency:**



· **Jitter:**



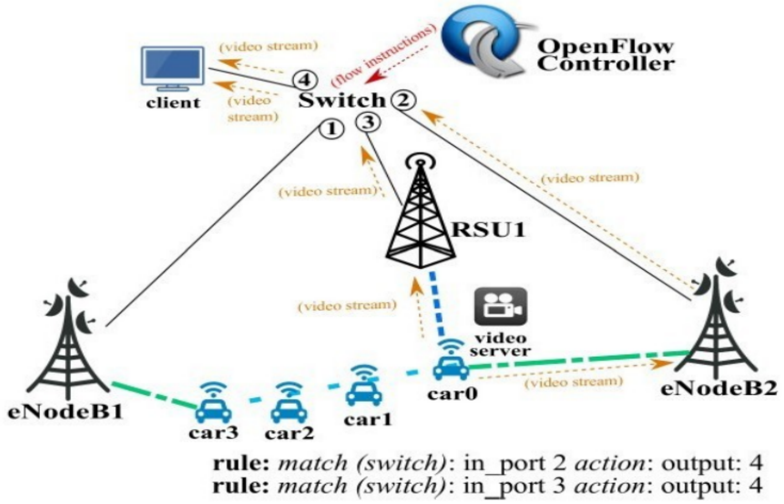
·**Throughput:**



3 Second Experiment

3.1 Phase One

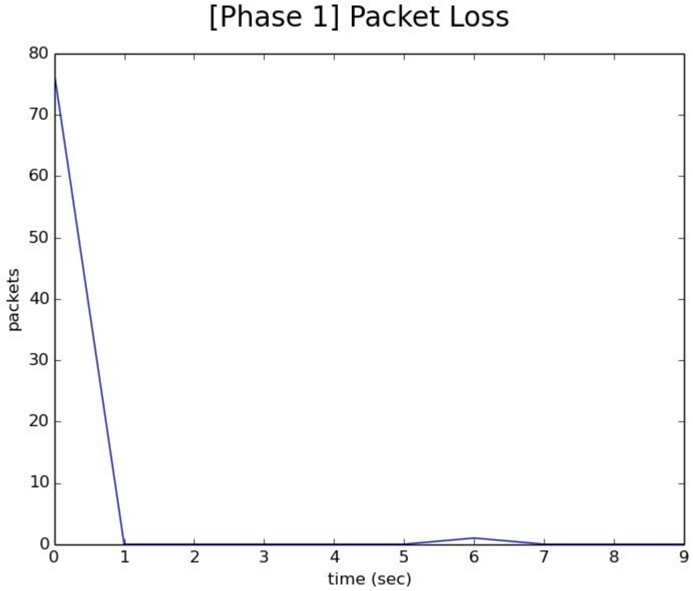
The vehicle car0 performs bicasting simultaneously to both RSU1 and eNodeB2.



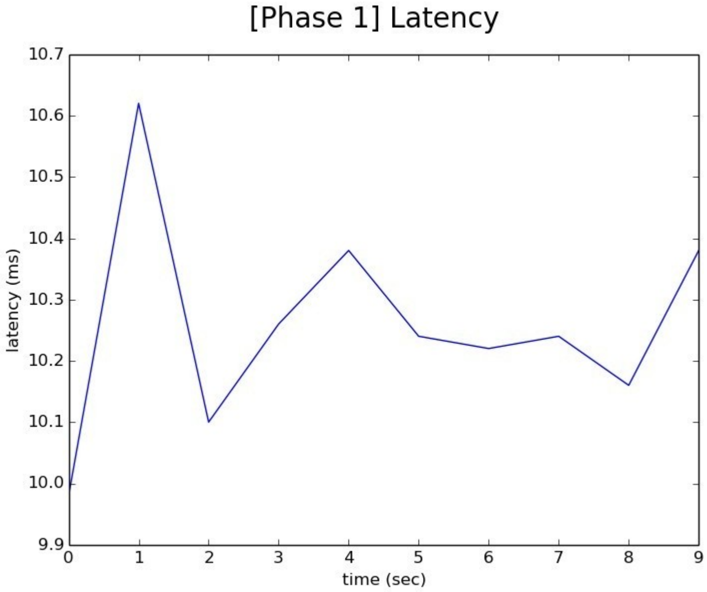
rule: *match (switch)*: in_port 2 *action*: output: 4
rule: *match (switch)*: in_port 3 *action*: output: 4

The results after conducting the experiment were:

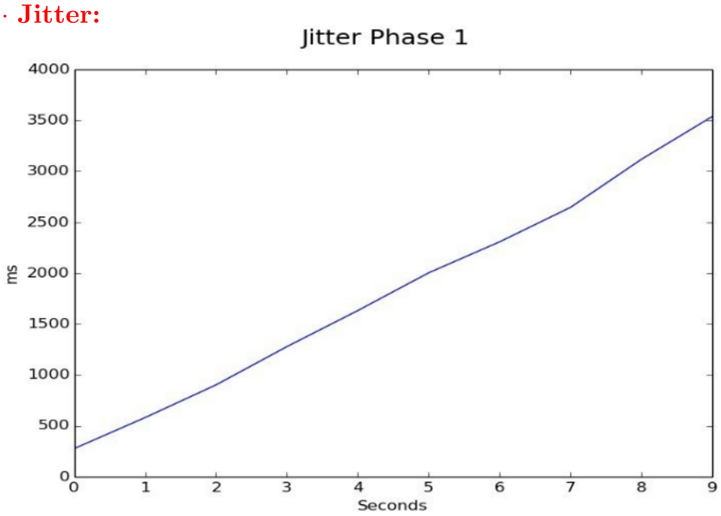
· **PacketLoss:**



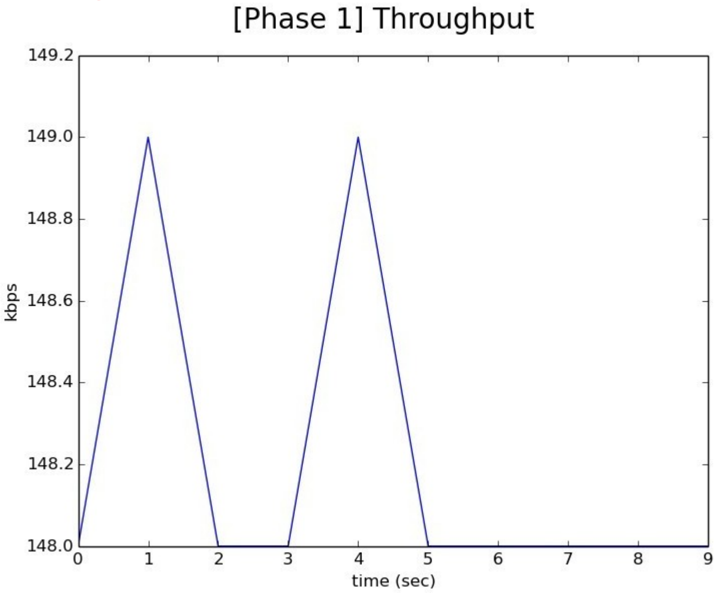
· **Latency:**



· **Jitter:**

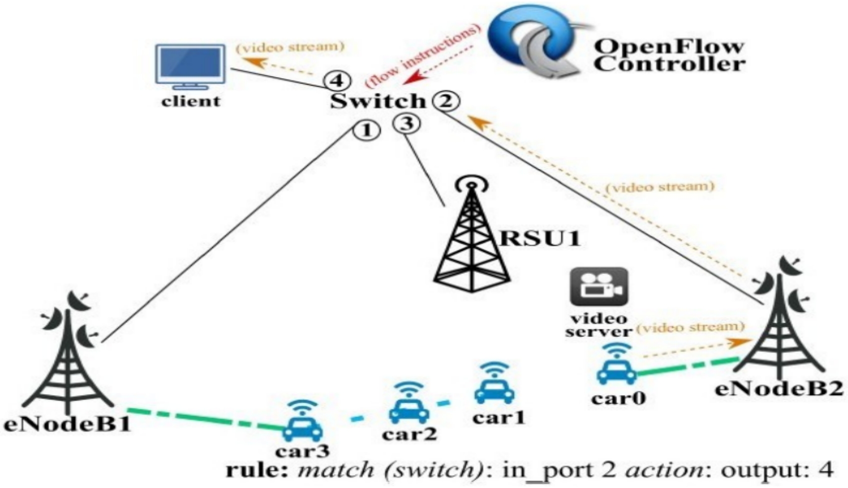


·Throughput:



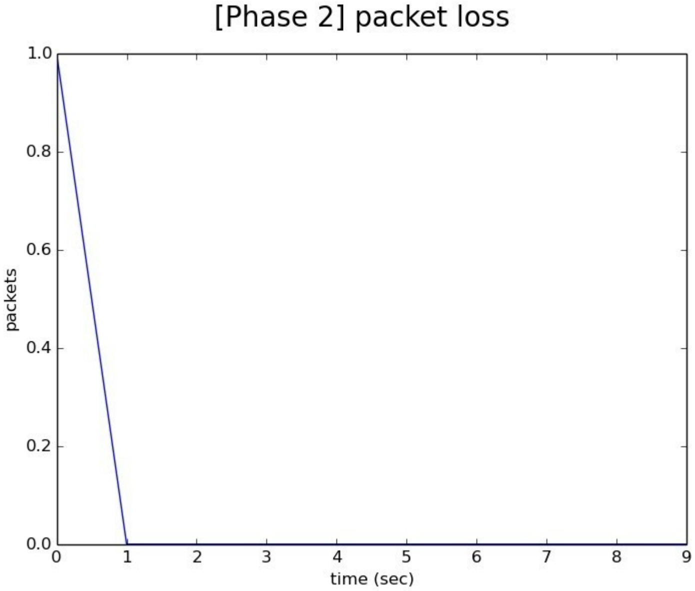
Concluding, provided that vehicle car0 is broadcasting a video stream, the propability for a packet not reaching the final destination, is significant. Therefore, PacketLoss and Jitter are extensive.
Since there are not duplicate packets, Throughput is quite low and stable.

3.2 Phase Two

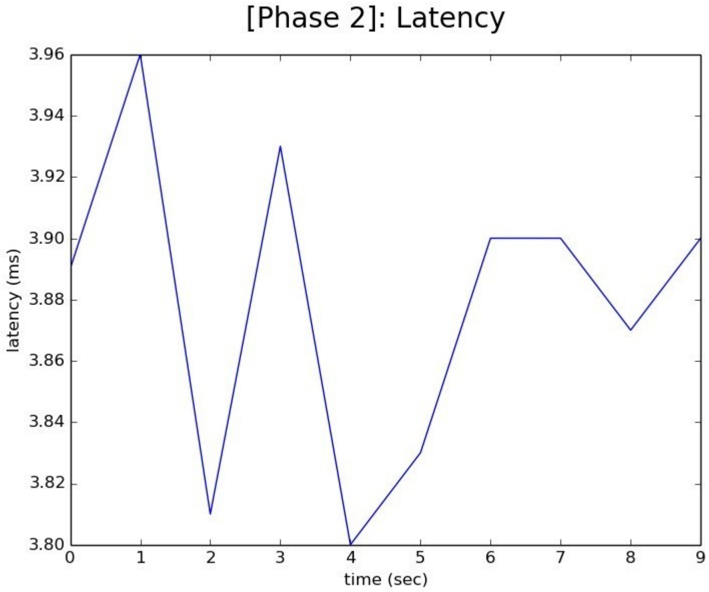


The results after conducting the experiment were:

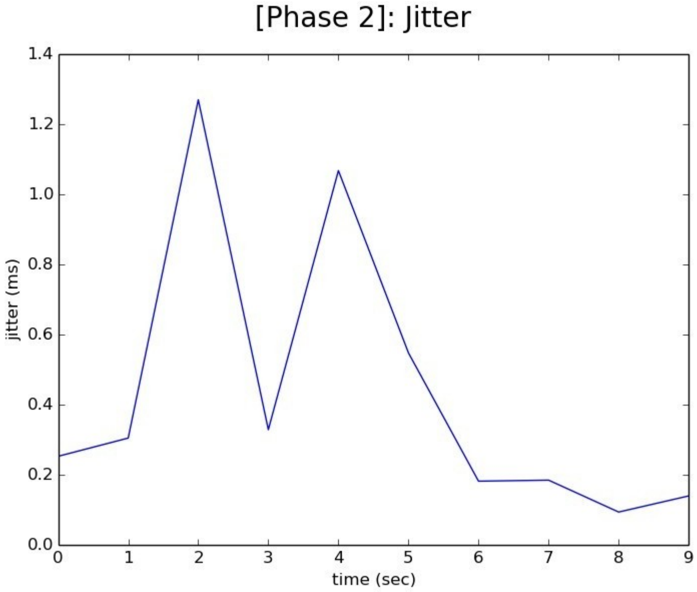
· **PacketLoss:**



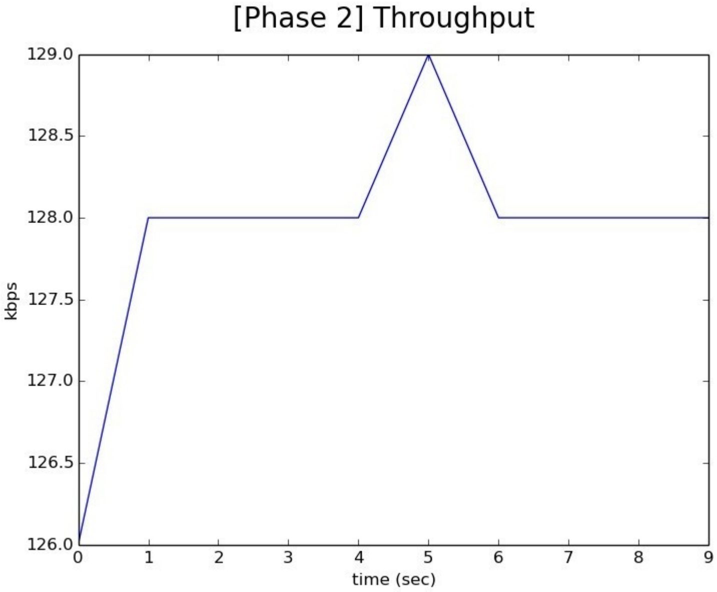
· **Latency:**



· **Jitter:**



·**Throughput:**

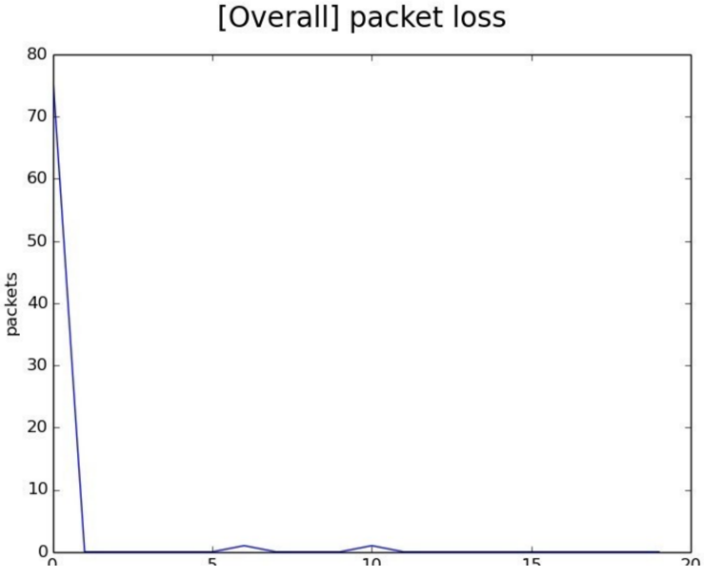


Similarly with the *Third Phase of the first experiment*, Jitter is quite close to zero, Latency is quite low, as well as Throughput, since the connection is only through eNodeB1.

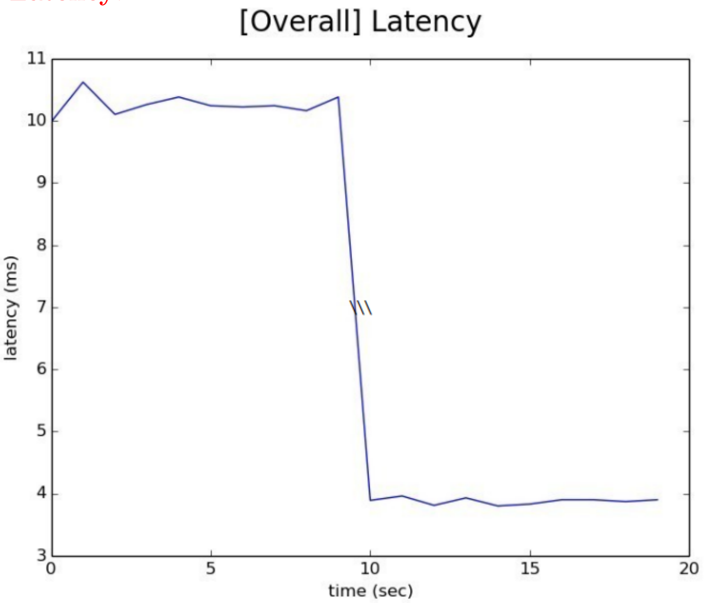
3.3 Total representation

Bellow we can observe the overall differences between the metrics of all three phases of the experiment.

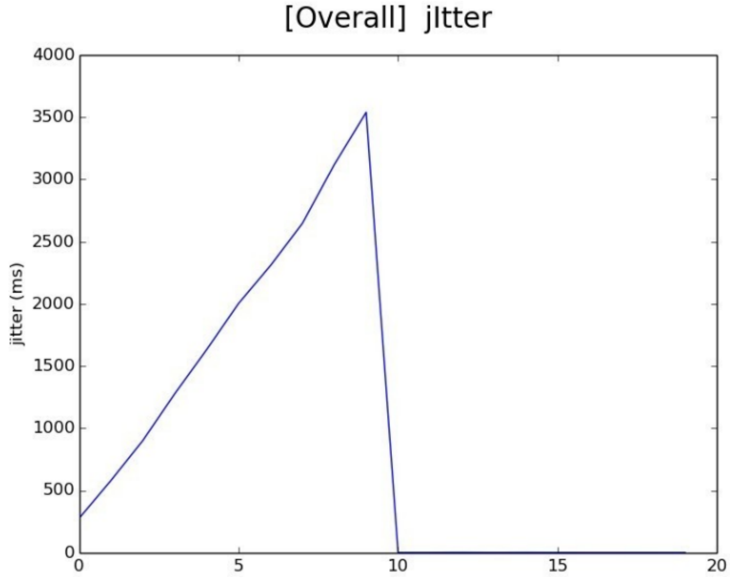
· **PacketLoss:**



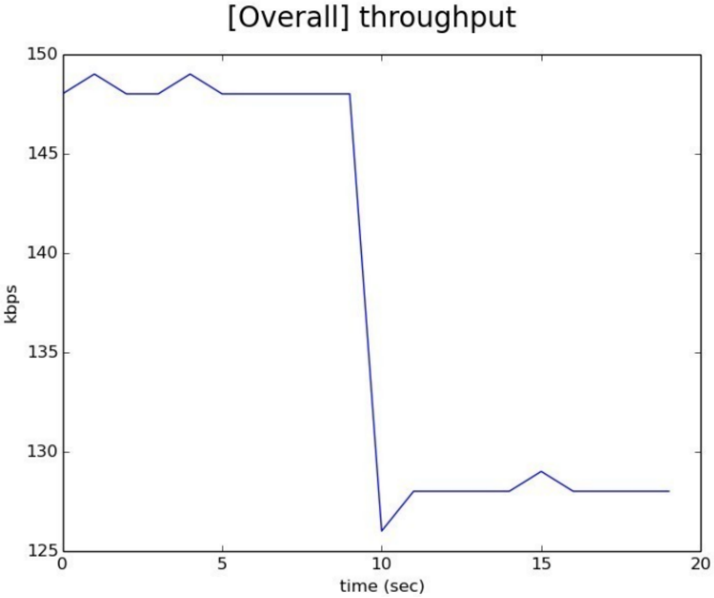
· **Latency:**



· **Jitter:**



·**Throughput:**



4 Comparison between the two experiments

At the *Second Experiment*, with bicasting, there is an extensive rise of PacketLoss. This happens due to the fact that the packets are getting broadcasted to two nodes.

△ *Noted* that the amount of PacketLoss changes at from one execution of the experiment to another, due to random factors that affect the results.

On the other hand, Latency is clearly lower at the *Second experiment with bicasting*, since there are not many hops, just like the *First Phase* of the experiment.

Results regarding Throughput in the case of bicasting, are quite lower compared to those of *Phase One of first experiment*, since in the first case packets are getting broadcasted once, while in the second case we have duplicated packets.

Finally, it's obvious that Jitter at the *second experiment* is much greater tha the *first*, because while broadcasting occur many variations of Latency which leads to rising of Jitter.