Testbed for Multi-access Edge Computing V2X applications prototyping and evaluation

Bilel CHERIF, Nicolas RIVIERE, Pascal BERTHOU, Yann LABIT

SARA/TSF Team, LAAS-CNRS

ERTS, January, 29th 2020

Agenda

- Introduction
- Context and motivations
- MEC architecture
- Existing evaluation and prototyping platforms
- Proposed testbed Architecture
- Use Cases
- Conclusion

Introduction

3 488 persons dead by road accidents in France

90 % of all accidents depend on human error

Germans spend on average 36 hours p.a. in traffic jams

The manner of driving has an impact on the fuel consumption up to 20%

[source] : ONISR, Observatoire national Interministériel de la sécurité routière, “Bilan de l'accidentalité de l’année 2018”
Frank Försterling, “Electronic Horizon How the Cloud improves the connected vehicle”, Wien, 2015

Context and Motivations(1/3)

- Context : Intelligent Transport System (ITS)
  - Exploits networking and cloud technologies.
  - Offers a whole new set of services to improve the automotive system's safety, comfort, and efficiency.

- ITS services with various quality of service requirements,
  - Safety : latency < 100 ms, high reliability

    ex : Cooperative Collision Avoidance (latency : 100 ms, high reliability requirements :10^{-5}) [1] *(V2V, V2P)*
    Intersection management service
  - Non Safety :

    ex : Traffic information and recommended itinerary (latency: 500 ms, low reliability requirements) [2] *(V2I)*

[1] 5G-PPP, 5G Automotive Vision, white paper, October 20, 2015
[2] Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Definitions, ETSI TR 102 638 V1.1.1 (2009-06)

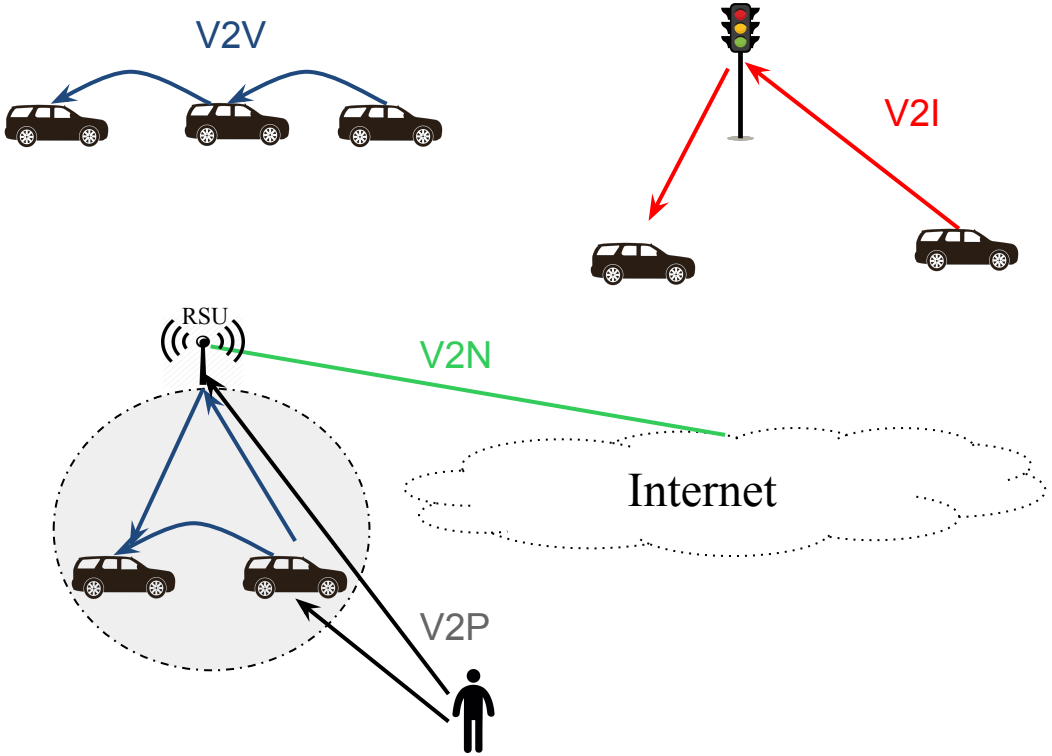
Context and Motivations (2/3)

Vehicle to vehicle (V2V)

Vehicle to infrastructure (V2I)

Vehicle to network (V2N)

Vehicle to Pedestrian (V2P)

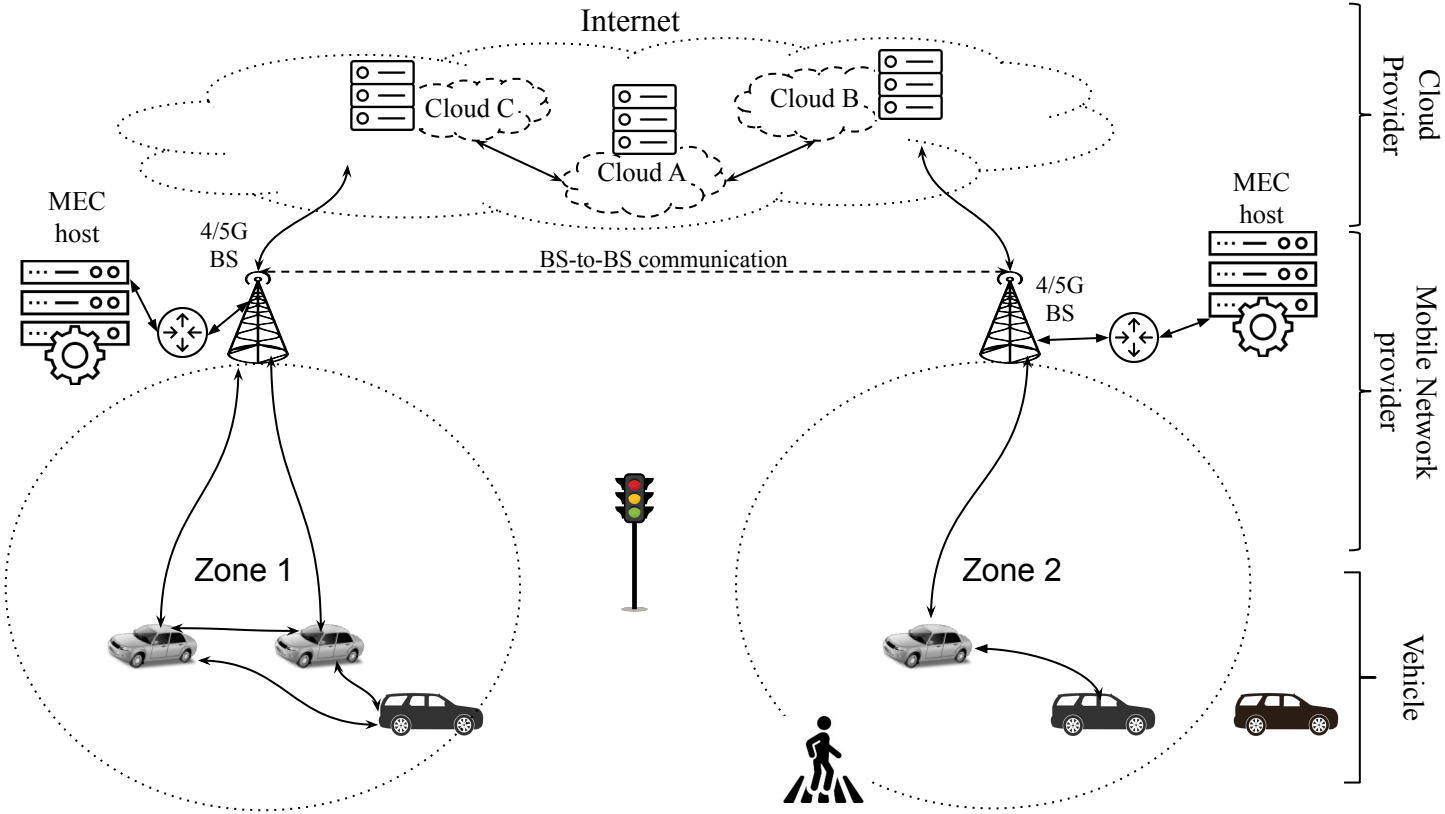


Context and Motivations (3/3)

Very low latency

High bandwidth

Network aware services



MEC architecture

MEC orchestrator:

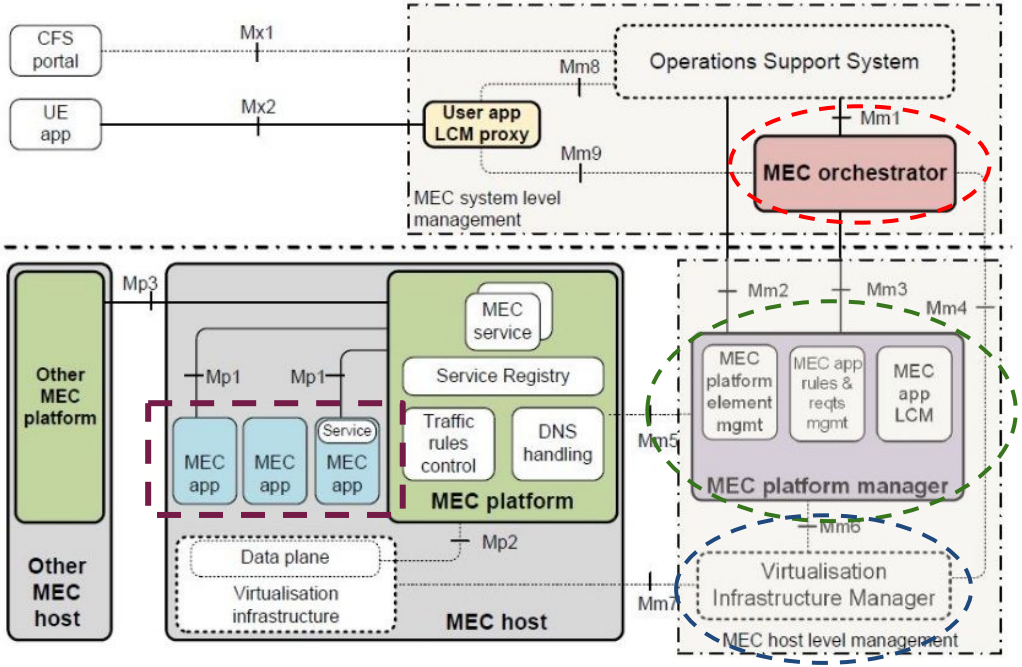
- maintains an overall view on available computing, storage, and networking resources and services
- Host selection for requested services deployment.
- Services scaling.

MEC platform manager:

- Mobile Edge platform management
- Application lifecycle management (instantiation and termination).
- Application requirement management.

Virtualization infrastructure manager:

- Virtualized resources management.
- Fault and performance monitoring.



Existing network evaluation platforms (1/2)

- No specific tool that supports the MEC platform architecture.
- Can not run application code directly without any adaptations (application code and its dependencies).
- Time cost (protocol modeling + application source code adaptation).
- Does not support complex nodes mobility (Vehicles mobility models).
- Complex integration of features like timers and threads used in realworld applications.

Existing platforms (2/2)

Some existing solutions are efficient in term of network related evaluations.

Veins :
Based on omnet++ and sumo simulators

ITetris :
Based on NS2 and sumo simulators

4 methods found in the state of the art

Application modeling:
- Oversimplified.
- Doesn't model the actual application execution.

Socket connection:
- CPU scheduling issues
- Synchronization issues

Source code integration
- Time costy.
- Could lead to application code unstability.
- Time based functions should be adapted to the simulation time domain.

Shared library integration:
- Huge amount of code that should be modified and rebuilded.
- Time based functions should be adapted to the simulation time domain.

Proposed architecture

- Network emulator
- Mobility simulator
- Host emulator

Coupled

- Hosts orchestration and resources management capabilities
- Application deployment and management capabilities

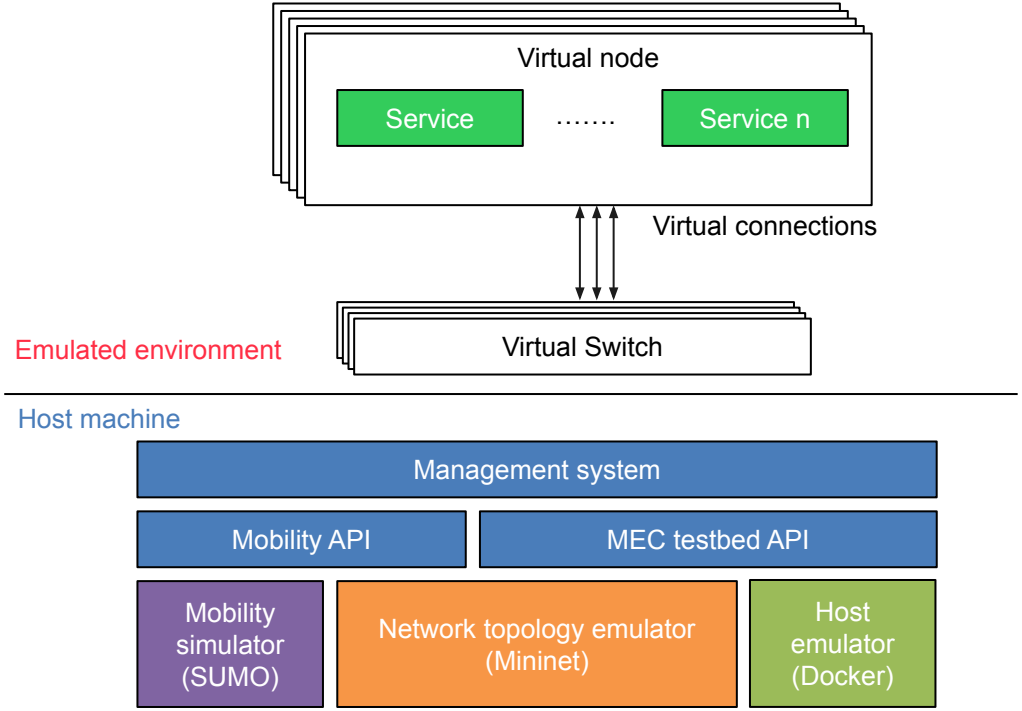
Extended

Proposed architecture

Mininet:
- Network topology emulation
- Network link configuration through queuing discipline (TcLink)

Docker:
- Hosts emulation
- Hosts isolation

Management system:
- Hosts orchestration
- Mobility management
- Association control
- Ressources management



MEC testbed - Workflow

Build nodes images

Docker image for each type of nodes

Define network topologies

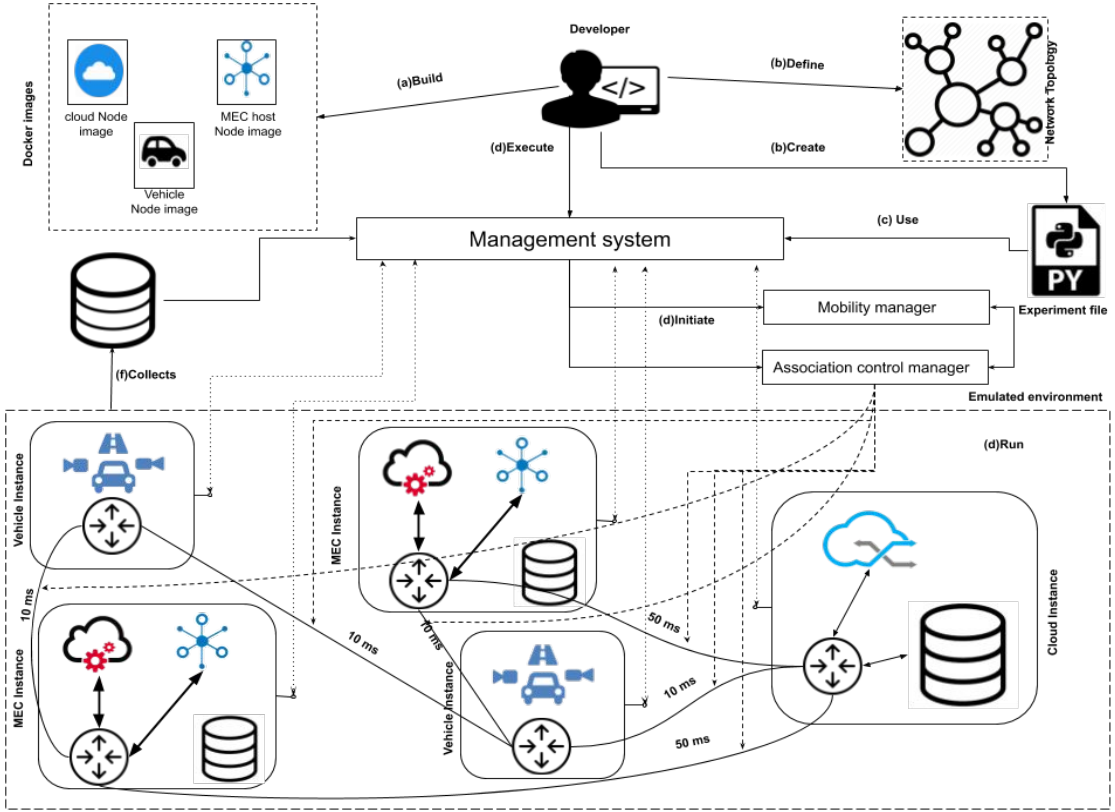
Fixed part of the network (Cloud, Mec hosts)

Define mobility model

Random, Gauss-Markov, External mobility simulator (sumo) …..

Define association policies

Distance, host load, delay ……...



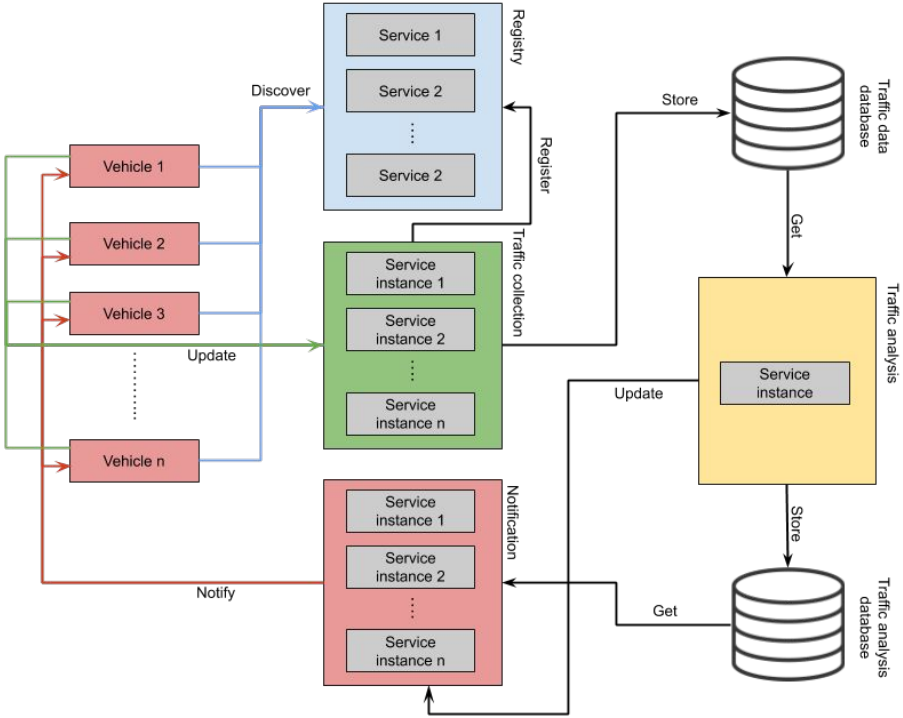
Proposed architecture - Opportunities

- Realworld protocols support.
- Running actual application code.
- Topology flexibility.
- Nodes mobility support.
- Network performance reconfigurability (Delay, Packet Loss ratio ..)

Use Case

Real-time traffic monitoring service:

- Microservices oriented architecture.
- Vehicles communicate traffic collection microservices to post/update their locations and speed.
- The traffic analysis microservice analyzes the collected vehicles' data to determine the vehicles' traffic flow.
- The traffic analysis database store the traffic flow information and expose to the other hosts.
- The notification microservices updates the vehicles regarding the traffic flow changes.

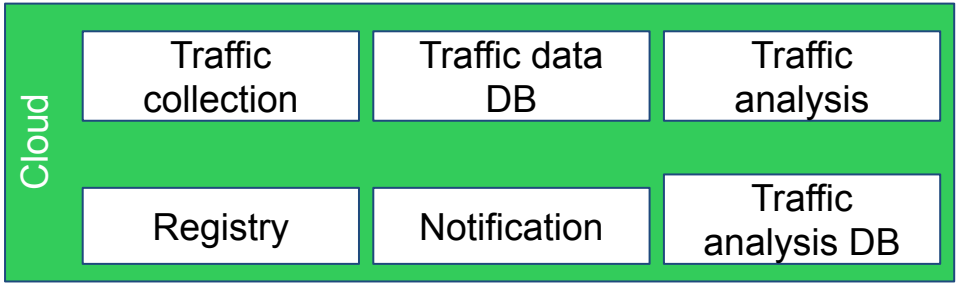


Simulation - Scenario (1/2)

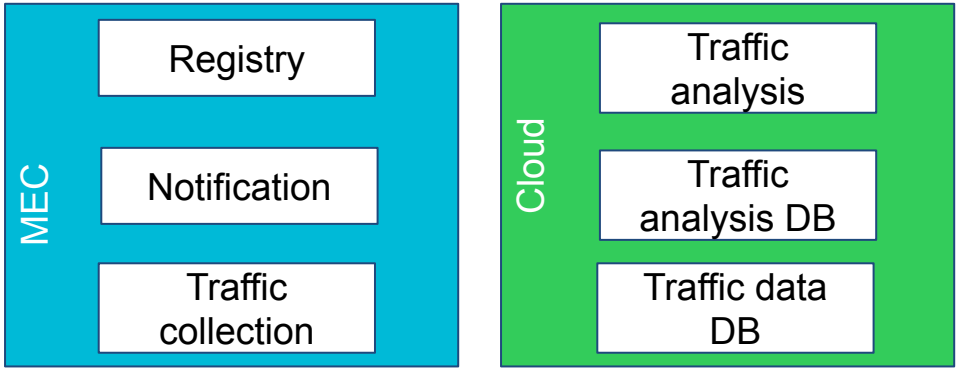
- **Goal :**
- Evaluate the testbed platform.
- Use the actual application code without any modifications.
- Evaluate the service under different topologies and configurations.

Traffic collection	Flask (RESTfull)
Traffic analysis	Flask (RESTfull)
Registry	Consul
Notification	Flask (RESTfull)
Traffic analysis DB Traffic data DB	Mongodb (RESTfull)

Scenario 1



Scenario 2



Simulation - Scenario (2/2)

	Scenario 1	Scenario 2
CPU	Intel(R) Core(TM) i7-4750HQ CPU CPU(s): 8 Thread(s) per core: 2 Core(s) per socket: 4 Frequency: 2.00GHz Max frequency: 3.2 GHz	Intel(R) Core(TM) i7-4750HQ CPU CPU(s): 8 Thread(s) per core: 2 Core(s) per socket: 4 Frequency: 2.00GHz Max frequency: 3.2 GHz
RAM	16 GB RAM Speed: 1600 MT/s	16 GB RAM Speed: 1600 MT/s
Allocated resources per host (CPU in cpu numbers) (Memory in Megabyte)	Vehicles: {"cpu": 0.25, "memory": 64} Cloud: {"cpu": 2, "memory": 2048}	Vehicles: {"cpu": 0.25, "memory": 64} Cloud: {"cpu": 2, "memory": 2048} MEC: {"cpu": 1, "memory": 512}
Link delay	Vehicle-Cloud: 100 ms Vehicle-Vehicle: 10 ms	Vehicle-MEC: 10 ms Vehicle-Vehicle: 10 ms Cloud-MEC: 50 ms
Simulation parameters	**Mobility:** Gauss-Markov model velocity_mean =3 0 alpha = 0.9 variance=0.5 Dimension = (300, 10) Number of nodes = 10 Number of Cloud hosts = 1 Cloud host position = (100, 5) **Association control model:** Algebraic distance communication range = 100	**Mobility:** Gauss-Markov model velocity_mean = 30 alpha = 0.9 variance = 0.5 Dimension = (300, 10) Number of nodes = 10 Number of MEC hosts = 2 MEC host 1 position = (0, 5) MEC host 1 position = (0, 100) **Association control model:** Algebraic distance communication range = 50
Microservices placement	Everything on the Cloud host	**MEC host:** Registry. Traffic collection microservice. Notification microservice

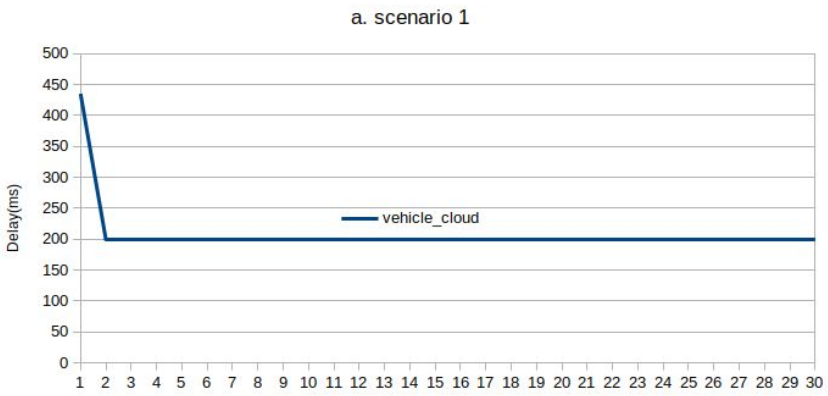
Simulation - Results

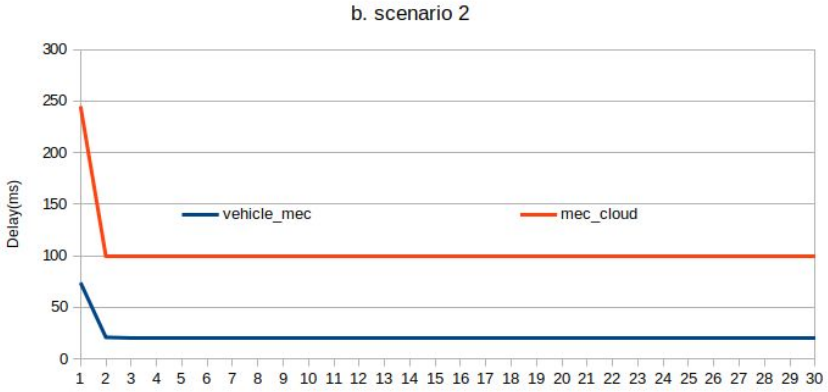
Round Trip Time:

Round Trip Time, is the time required for a packet to travel from a specific source to a specific destination and back again.

The delay is higher than the configured delay parameter, then it remains stable at the theoretical value.

The high delay value at the launch time of the emulation is caused by the CPU load at the initialization phase.





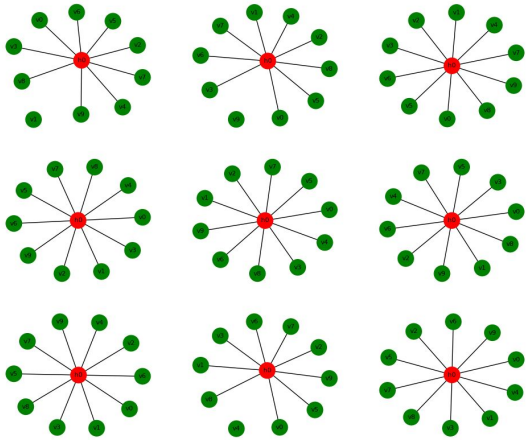
Simulation - Results

Association control:

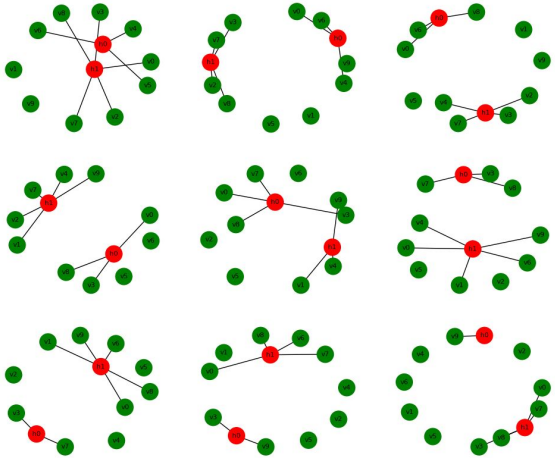
The vehicles nodes are associated to the hosts that satisfies the association policy.

In both scenarios the associations are only based on the distance of the vehicle regarding the Host (Edge/cloud).

Scenario 1



Scenario 2



Conclusion

The proposed tool models V2X applications deployment environment through network emulation. Such a tool opens the opportunities to:

- Running services under different scenarios.
- Validation of the application's behavior and its interactions.

Toward a testing tool for V2X applications prototyping and validation:

- Accurate wireless network model.
- Propagation model implementation.
- automated log analysis tool.

Thank you for your attention !

Questions ?