Installation NS3 pour communiquer avec ROS et CARLA Simulator

Afin de réaliser cette installation, on se basera sur un système d'exploitation Ubuntu 20.04 natif pour des raisons de compatibilité principalement.

Nous allons nous référer à la documentation officielle, et nous allons utiliser la version de NS3 3.30.1 pour des raisons de comptabilité.

Pour installer CARLA Simulator et ROS, puis faire le lien entre eux, suivez les documentations suivantes.

➔INSTALLATION DE NS3

NS3 est un logiciel de simulation qui permet de modéliser et d'analyser le comportement de réseaux informatiques et de communications sans avoir besoin de matériel réel. Dans notre cas nous allons installer NS3 pour simuler les échanges réseaux entre des véhicules dans CARLA Simulator.

➔INSTALLATION ROSNS3-SERVER

Les modules dans NS3, situés dans le répertoire src, représentent des bibliothèques de composants réutilisables qui implémentent différentes fonctionnalités de simulation réseau, telles que les protocoles de routage, les modèles de mobilité, ou les dispositifs de communication sans fil.

Pour assurer la liaison entre ROS et NS3, nous allons alors installer le module ROSNS3-Server. Ce module NS3 va servir de serveur UDP pour récupérer les messages que l'on souhaite envoyer depuis ROS. Il existe d'autres alternatives comme par exemple créer un serveur UDP asynchrone directement, mais ce module permet la simplification des interactions.

➔INSTALLATION DE NETANIM

Netanim est un outil de visualisation graphique qui permet de voir et d'analyser simulations de réseaux réalisées avec NS3, rendant les données complexes facilement compréhensibles et analysables. Nous allons utiliser cet outil pour visualiser nos simulations.

I) Installation NS3

- Installez les dépendances nécessaires

```
sudo apt install g++ python3 python3-dev pkg-config sqlite3 cmake
sudo apt install python3-setuptools git
sudo apt install qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools
```

- Récupérer la version de NS3 pour l'installation puis la décompresser

...ou en ligne de commande

```
mkdir ~/ns3
cd ~/ns3
wget https://www.nsnam.org/releases/ns-allinone-3.30.1.tar.bz2
tar -xvf ns-allinone-3.30.1.tar.bz2
```

- Configuration et build avec waf

```
cd ns-allione-3.30.1/ns-3.30.1/
./waf configure --enable-examples --enables-tests
./waf build
```

- Testez le bon fonctionnement avec un script exemple

```
./waf –run scratch-simulator
```

Vous devez obtenir ce résultat

```
> ./waf --run scratch-simulator
Waf: Entering directory `/home/gab/tarballs/f/ns-allinone-3.30.1/ns-3.30.1/build
'
Waf: Leaving directory `/home/gab/tarballs/f/ns-allinone-3.30.1/ns-3.30.1/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (1.412s)
Scratch Simulator
```

II) Installation du module ROSNS3-Server

- Récupérer le module

```
cd src/
git clone https://github.com/malintha/rosns3_server.git rosns3
```

- Modifier une ligne dans le fichier « wscript »

```
cd rosns3
sudo nano wscript
```

→ Supprimer dans la variable « module » (ligne 11) l'élément 'visualizer'

```
def build(bld):
    module = bld.create_ns3_module('rosns3', ['aodv','mobility','core','network', 'visualizer', 'olsr','applications', 'netanim'])
```

```
def build(bld):
    module = bld.create_ns3_module('rosns3', ['aodv','mobility','core','network','olsr','applications', 'netanim'])
```

- Configuration et build avec waf

```
cd ../../
./waf configure --enable-examples --enables-tests
./waf build
```

- Testez le bon fonctionnement avec un script exemple

```
./waf –run rosns3-example
```

Vous devez obtenir un résultat similaire

```
Initializing rosns3-server on port: 28500
Using Friss propagation loss model with:
        Path loss exponent: 2.4
        Reference power loss: 46dBmW
        Tranmission power: 16.0206dBmW
        Fading mean: 0
        Fading variance: 32
```

III) Installation de Netanim

- Build de Netanim

```
cd ../netanim-3.108
make clean
qmake NetAnim.pro
make
```

- Lancement de Netanim

```
./Neatim
```

→ Vous obtiendrez une fenêtre comme celle-ci (ici une simulation est chargée)