ROS 2

- Extension
- ROS2 Container to container communication

Extension

Following the documentation here

Build The Extension

This directory contains a script create_extension.sh that can be used to create an l4t-based Spot Extension for this example. This will create a file spot_detect_and_follow.spx, which can be uploaded to the CORE I/O. The extension requires that the payload be authorized on the robot admin console to run.

If building on a host system architecture that is not `ARM64` based run the following before continuing.

```
sudo apt-get install qemu binfmt-support qemu-user-static
```

Installing and running qemu will allow us to build ARM binaries on an x86 machine without needing a cross compiler, see Build Docker Images Documentation for more details.

This directory contains a script `create_extension.sh` that can be used to build a `ARM64` docker image and package all the files into an Extension. From the `ros2_driver` directory run the script

```
./create_extension.sh
```

This will create the `spot_ros2_driver.spx` extension file that you can upload to the CORE I/O or Scout platform. The script was tested on Ubuntu 22.04 with x86 architecture.

Before you install and run the `spot_ros2_driver.spx` you need to configure the port range used by a connection in the CORE I/O to be within the allowable port range.

Limit the ports used by a connection in the CORE I/O

SSH into the CORE I/O from the robot's WiFi

```
ssh -p 20022 spot@192.168.80.3
```

Make a copy of the default port range (not currently working with our CORE io)

```
cat /proc/sys/net/ipv4/ip_local_port_range > /proc/sys/net/ipv4/ip_local_port_range.bak
```

The default configuration of the ip_local_port_range

```
32768  60999
```

Limit the ports a networking connection can use to those reachable through the CORE I/O's firewall

```
echo "21000 22000" | sudo tee /proc/sys/net/ipv4/ip_local_port_range
```

This will force all connections from the CORE I/O to be on the port range 21000 - 22000.

ROS2 Container to container communication

To-Do

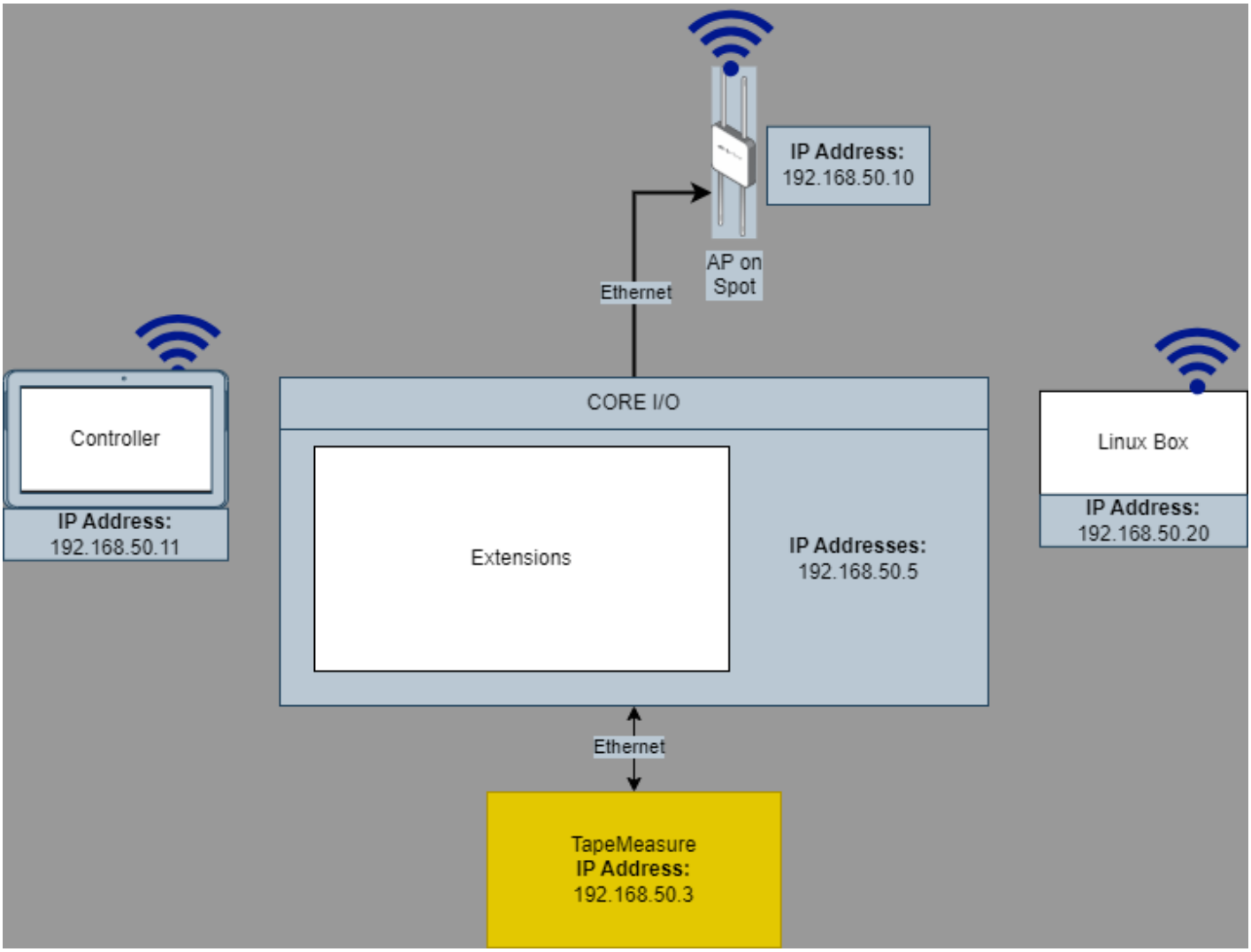
- Determine why multicast send and receive are not working on our network.
- More testing with other ROS2 packages.

Setup

In this setup, we have two containers:

1. The Boston Dynamics ROS2 Spot ROS2 container in the an extension file build using these instructions.
2. Our container built for Spot.

Network



Container Configuration

Extension Container

1. You need to ssh into the CORE I/O via the default IP of 192.168.80.3 or 192.168.50.3 if you are connected directly to the CORE I/O.
   - `ssh 192.168.80.3`
2. You will need to lock down the CORE I/O to a specific port range. This is so that the containers choose ports that are open.
   - `echo "21000 22000" | sudo tee /proc/sys/net/ipv4/ip_local_port_range`
3. **You must restart the containers/extension after changing the ports on the CORE I/O**
4. You need to find the container id of your ROS2 installation.
   - `sudo docker container ls`

1. Once you have your container id you will need to open up into the container.
    - `sudo docker exec -it 8966bcde886b /bin/bash # replace "8966bcde886b" with your container id`
2. Once inside the container.
    - The Extension container was setup with the following commands to test:

```
ping 192.168.50.20 # Ping test the dev computer.

export ROS_DISCOVERY_SERVER=192.168.50.5:21000 # This was changed from the BD default of
192.168.80.3 because we are using our own WiFi setup that bypasses Spot's internal network.

source ./install/setup.bash # Source workspace
source /opt/ros/humble/setup.bash # Source ROS2

ros2 run demo_nodes_cpp listener --ros-args --remap __node:=listener_discovery_server
```

Dev Container

The Dev container was setup with the following commands to test:

```
ping 192.168.50.5 # Ping test the CORE I/O.

export ROS_DISCOVERY_SERVER=192.168.50.5:21000 # This was changed from the BD default of
192.168.80.3 because we are using our own WiFi setup that bypasses Spot's internal network.

source ./install/setup.bash # Source workspace
source /opt/ros/humble/setup.bash # Source ROS2

ros2 run demo_nodes_cpp talker --ros-args --remap __node:=talker_discovery_server
```