Project Report: M2M Motor rig project

**Document information**

|  |  |
| --- | --- |
| Subject | EEE8089 M2M Technology IoT |
| Authors | Elena Quesada |
| Keywords | M2M, Raspberry, Routing, Motor, LAN, PID, SSH |

**Table of contents**

[1. DESCRIPTION OF THE PROJECT 3](#_Toc39176503)

[1.1. CONFIGURATION OF THE NETWORK 4](#_Toc39176504)

[1.2. SECURESHELL LAYER 6](#_Toc39176505)

[1.3. MOTOR RIG 6](#_Toc39176506)

[2. CONCLUSIONS 6](#_Toc39176507)

[ANNEX 7](#_Toc39176508)

1. DESCRIPTION OF THE PROJECT

The aim of the project is to build a safety Machine to machine (M2M) network to control remotely the spinning speed and the direction of a motor rig.

The M2M network will be defined following OSI model to implement each of the layer related in the network as it is seen on Figure 1. TCP/IP over Ethernet will be configured and will support the secure shell, allowing the application to transmit in a secure way. The application layer will be the design of the motor rig controller. The different layers can be seen in the figure.



Figure 1. OSI model analogy

Additionally, WIFI protocol will be used to develop the extend the functionality of the network. 4 different raspberry PI will be used as nodes to build, configure and implement the network together with a switch to connect 3 different private LANs over ethernet connection.

The application layer will design the motor rig controller. The direction and the spin speed of the motor will be commanded and send to the motor using the UART. Reading from a built-in encoder the application will get some feedback related to the real speed and direction of the motor and acts consequently. The implementation of a PID will allow correcting the error got from the motor due to external factors. The motor rig con used is a 24V DC motor provided by the university together with an already programmed protoboard to control it. The specification of this protoboard has to be consider when the motor is going to be used.

Although no functional application has been built over the motor controller, different functionalities can be implemented. For example, once it is possible to control the speed or frequency of the motor, associating each frequency with a note, different music songs can be played. This functionality was not an intended part of the project, but is good to look forward and see the usability of the project.

This report presents the SW implementation of the application layer and the network configuration that has been carried out during the lab sessions. Thus, if it was not possible to finish some of the stages of the project due to the current situation, a theorical approach is given instead.

* 1. CONFIGURATION OF THE NETWORK

Figure 2 shows the layout of the network. As it can be seen 4 nodes are connected between them using 3 different private LANs. Even if all of them are connected to the same switch as a matter of resources, it is not going to be able to reroute data for example from node 1 to node 4. This way, a message from a node will have to go through all the different LANs in the middle until its destination. This configuration is intended to reflect a more complex network where different LANs are needed because of the coverage distance.

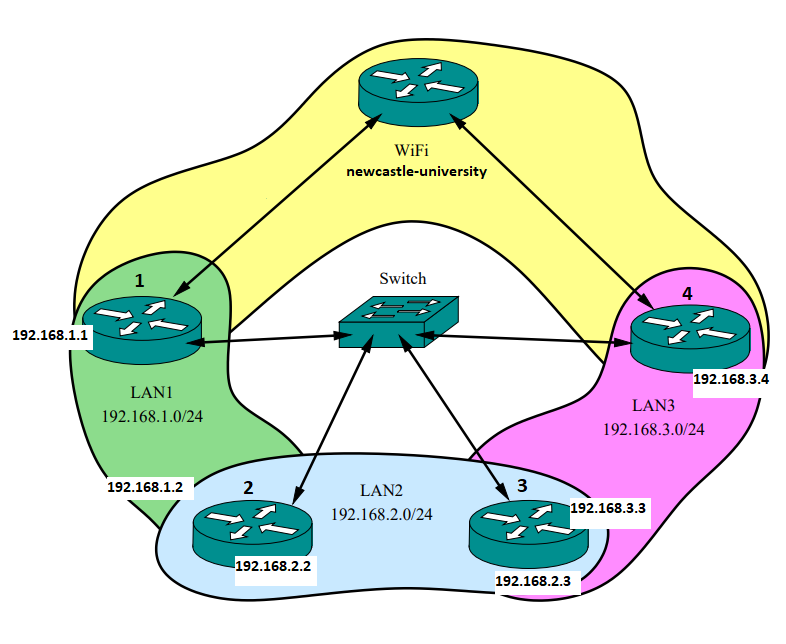


Figure 2. Network configuration

Same reason can explain the configuration of WIFI connection, although it is not explicitly used in this project. Newcastle university WIFI is configured allowing to send data to the network. This data could have been displayed in the cloud and used later on for further analysis.

The configuration of the network is presented in detail in the following sections.

* + 1. WIFI CONFIGURATION

Edge nodes will configurate WIFI connection and they will work as gateways in case any middle node needs to send data to the cloud. Since Newcastle WIFI connection is used, which is configure as a secure network, some specific configurations have to be made as follow.

Admin privileges are needed and ***wpa\_supplicant*** file has to be modified as shown in Figure 3. This configuration has to be done just once.

>>cd **/etc/wpa\_supplicant**

**>>sudo leafpad wpa\_supplicant.conf**

|  |
| --- |
| **network={**  **ssid="newcastle-university"**  **key\_mgmt=WPA-EAP**  **eap=TTLS**  **identity="your\_login\_name"**  **password="your\_password"**  **phase2="auth=MSCHAPv2"** **}** |

Figure 3.wpa\_supplicant file configuration

Node 1 and node 4 will configurate WIFI and work as a gateway for node 2 and 3. Middle nodes could have got WIFI connection too, but the system version of the raspberry Pi were not compatible with the configuration of several IPs id in the nodes without making some other changes, and since it was not part of the aim of the project, it was not made.

Network file has to be also configured in order to set wlan0 as it is shown in Figure 4:

>>cd **/etc/network**

**>>sudo leafpad interfaces**

|  |
| --- |
| **# Configure WIFI**  **allow-hotplug wlan0 iface wlan0 inet manual wpa-conf /etc/wpa\_supplicant/wpa\_supplicant.conf** |

Figure 4. Interface file configuration for WIFI

* + 1. LAN CONFIGURATION

Three different private LANs are configured, each of them with its on IP. Each node will work as a gateway for other LAN so each node will have two different IP configuration. Figure 2 shows the distributions of the LAN and the IP given to each of them.

In order to configure each of the LANs, the interface file has to be modified. An example of the configuration needed is shown in Figure 5 where LAN 1 and LAN2 are configured with their Ips for node 2. Each node will configure two interfaces: either two LAN or one LAN and the WIFI interface. Both interfaces exist together in the same node allowing the data routing from one node to other:

>>cd **/etc/network**

**>>sudo leafpad interfaces**

|  |
| --- |
| **# Configure LAN 1**  auto eth0  allow-hotplug eth0  iface eth0  inet static address 192.168.1.2  netmask 255.255.255.0  **# Configure LAN 2**  auto eth0:1  allow-hotplug eth0:1  iface eth0:1  inet static address 192.168.2.2  netmask 255.255.255.0 |

Figure 5. Interface file configuration configuring LAN 1 and LAN 2 for node 2

The configuration of the interface file has to be made each time the node is rebooted either manually or using a script.

* + 1. ROUTING

As it was said, middle nodes are going to work as routers. Thus, IPv4 routing must be enable in the raspberry PI. Furthermore, since all of the nodes are connected to the same switch but we do not want Linux to update its ARP table, it is needed to disable redirecting accepts and sends. In order to do that, sysctl.conf file has to be modified as it is shown on Figure 6. This way the messages will be routed from the node 1 to the node 4, going through all the nodes in the middle:

**>>**cd **/etc**

**>>sudo leafpad** sysctl.conf

|  |
| --- |
| #enable routing  net.ipv4.ip\_forward=1  # inhibit accept redirects  net.ipv4.conf.all.accept\_redirects = 0  #inhibits send redirects  net.ipv4.conf.all.send\_redirects = 0 |

Figure 6. sysctl.conf file

When a message has as destination a node that is not in the LAN, it has to be routed using the gateways. The gateways will be configured on each node in order to create the ARP tables according to the configuration. For example, node 4 will configures its gateway of LAN3 as follows:

>>**route add -net 192.168.30 netmask 255.255.255.0 gw 192.168.2.3**

**Thus, the gateway of the LAN3 for node 4 is the IP related with the LAN 2 for the node 3.** ARP tables needs to be fulfilled each time the node is rebooted either manually or using a script.

* 1. NAT CONFIGURATION

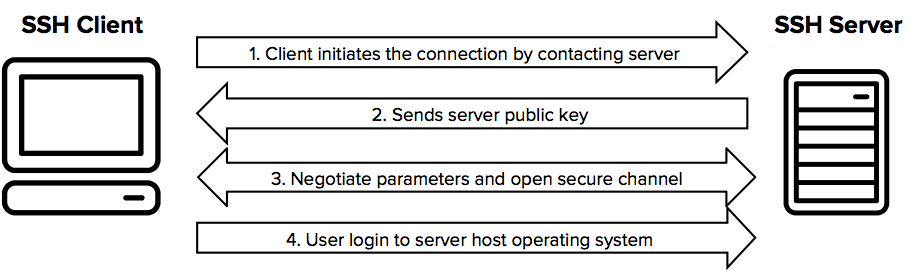
We are creating

* 1. SECURE SHELL LAYER

The secure shell (SSH) protocol is a method for secure remote login from M2M. It works as the safety and presentation layer of the network and guaranteeing to establish a secure session and transmitted in a controlled and reliable way protecting the communications security and integrity with strong encryption.

SSH works in the client-server model, where the client is responsible for estasblishing the connection with the sever which means that the connection is established by the SSH client connecting to the SSH server. The SSH client drives the connection setup process and uses public key cryptography to verify the identity of the SSH server. After the setup phase the SSH protocol uses strong symmetric encryption and hashing algorithms to ensure the privacy and integrity of the data that is exchanged between the client and server.

The figure below presents a simplified setup flow of a secure shell connection.



* 1. MOTOR RIG

CONCLUSIONS

ANNEX