Instituto Tecnológico Autónomo de México

Departamento Académico de Sistemas Digitales

*Nuevas Tecnologías Inalámbricas*

***Monitoreo y Análisis de una Red Ad-hoc***

Integrantes:

Alfonso Venancio- 149211

Efraín Aguilar- 149643

Leandro Pantoja- 150883

Ulises Alejandre - 159235

28 de febrero de 2019

**Introducción**

El ITAM tiene una red de radios que operan en la banda de 2.4 GHz y en esta tarea se realizó un análisis de la red, que es de tipo ad-hoc, para observar su desempeño. Los valores que se analizaron fueron el RSSI (potencia de la señal recibida) y LQI (calidad del enlace). La implementación se realizará en Python y se hará un análisis gráfico mediante R.

**Desarrollo**

Esta tarea consistió en realizar un program de Python que desplegará diversos parámetros de desempeño en un intervalo arbitrario.

El programa debe ser capaz de

1. Almacenar valores RSSI y LQI de cada enlace y graficarlos. Además de reportar el RSSI y LQI promedios, varianza, máximo y mínimo para cada enlace.
2. Reportar desconexiones y porcentaje de desconexión del total de consultas en el tiempo de análisis.
3. Medir latencia,(tiempo de respuesta gateway-tabla vecinos otros) y reportar promedio,varianza, maximo y minimo de cada nodo.
4. Analizar desempeño de la red con muestras tomadas durante **24 hrs**.

Algunas medidas a tomar son que el periodo de recolección de valores es de cada 10 min. Se usa el protocolo TCP para el acceso al gateway.

**Resultados**

Los resultados mostrados posteriormente son el resultado de haber dejado corriendo el programa de python durante 32 hrs aproximadamente.

Para la primera parte de esta tarea de encontrar todos los valores de RSSI y LQI primero se uso el codigo de practicas anteriores para encontrar todas las mac de los radios cercanos.

wsnQuery ='{"id": "routes", "mac": \"'+macLocal+'\"}'

Posteriormente se creo una funcion **val\_rssi\_lqi\_latencia(maclocal)**, la cual recibe como parámetro una mac y manda una consulta de todos los vecinos , para posteriormente recorrer una lista de todos los radios e ir extrayendo los parámetros **rssi, lqi, latencia, desconexión**  y los guarda en un diccionario.

wsnQuery ='{"id": "all\_neighbours", "mac": \"'+macLocal+'\"}'

for radio in arregloTodosRadios:  
 print(radio['mac'])  
 #diccionario con las mediciones  
 dicEstado = {  
 "ini\_monit" : str(time.ctime(start\_time)),  
 "mac\_local" : macLocal,  
 "mac\_destino": radio['mac'],  
 "rssi" : radio['rssi'],  
 "lqi" : radio['lqi'],  
 "latencia" : str(elapsed\_time),  
 }  
 arrStats.append(dicEstado)  
 return arrStats

Después de recopilar los parámetros deseados y exportarlos a un archivo csv **monitoreo.csv** , en R se hacen los ajustes correspondientes para poder leer los datos y se agrupa según el nodo y se ordena la latencia de forma ascendente. Con las función de **mean** se calcula el valor promedio del RSSI y LQI, con **var** la varianza, y con **min** y **max** el mínimo y el máximo respectivamente. Con el siguiente código se creó la tabla con la que se muestran las estadísticas por enlace:

data\_monitor\_nodo <- data %>%

group\_by(mac\_local,mac\_destino) %>%

summarise(latencia\_prom= mean(latencia), latencia\_var = var(latencia),

min\_lat = min(latencia), max\_lat = max(latencia),

rssi\_prom = mean(rssi),rssi\_var = var(rssi),

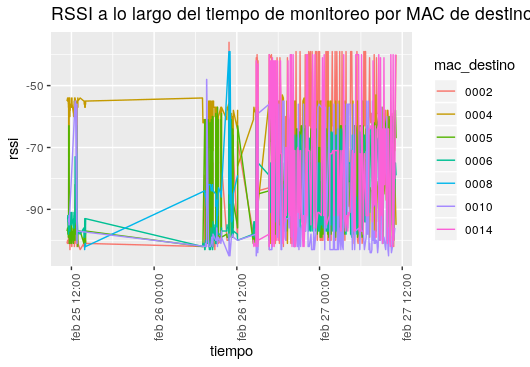
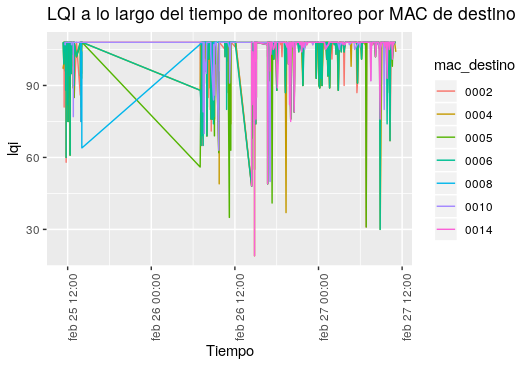
rssi\_min = min(rssi), rssi\_max = max(rssi),

lqi\_prom = mean(lqi), lqi\_var = var(lqi),

lqi\_min = min(lqi), lqi\_max = max(lqi)) %>%

arrange(mac\_destino, mac\_local)

Finalmente, con **ggplot** se grafican estos parámetros a lo largo del tiempo.



Para la segunda parte del trabajo, la cual consistió en reportar el número de desconexiones y su porcentaje respecto al total de consultas realizadas. Se modifico la funcion **val\_rssi\_lqi\_latencia(maclocal)** para que implementara un timer el cual nos indicaba de una desconexión en un nodo si el tiempo que transcurría era mayor a un segundo y de la misma manera este parámetro se agrega al diccionario y al archivo csv

desc = 0 #estado de conexion  
 if elapsed\_time > 1:  
 desc = 1

dicEstado = {  
 "ini\_monit" : str(time.ctime(start\_time)),  
 "mac\_local" : macLocal,  
 "mac\_destino": radio['mac'],  
 "rssi" : None,  
 "lqi" : None,  
 "desconexion": 1  
 }  
 arrStats = []  
 arrStats.append(dicEstado)

Después en R se realizaba la suma total de todas las desconexiones y para cada mac de radio se calculaba su total de desconexiones y su porcentaje con la ayuda de una agrupación a la base de datos con los parámetros de mac local y mac destino

#obtener mac y número de desconexiones de los nodos

desconectados <-data\_monitor\_desc %>%

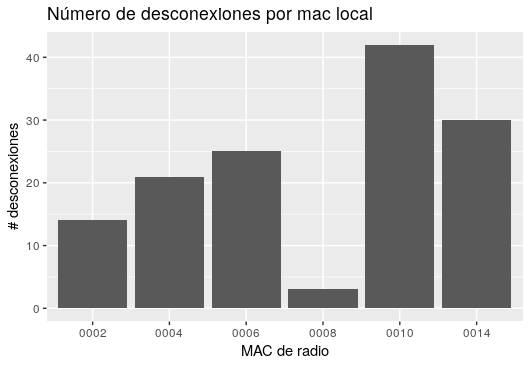
group\_by(mac\_local,mac\_destino) %>%

summarise(num\_desc = sum(desconexion))

desconectados <- desconectados %>%

mutate(proporcion\_desc = num\_desc/n)

Finalmente se usaba de nuevo la función ggplot para graficar el total de desconexiones según la mac.

****

Para la parte tres de medir la latencia de la red, se agregó un tiempo inicial después de enviar la consulta desde el gateway a los nodos vecinos y se paró el tiempo una vez que se obtenía la respuesta a la solicitud. Por medio de la resta del tiempo inicial y el final se encontraba la latencia la cual la guardamos en una variable **elapsed\_time,** después se agrega a la base de datos con el nombre de latencia. Finalmente en R se calcula la latencia promedio.

wsnQuery ='{"id": "all\_neighbours", "mac": \"'+macLocal+'\"}'  
 start\_time = time.time() #tiempo desde 1/1/1970 en segundos  
 respuesta = envia\_solicitud(wsnQuery)  
 end\_time = time.time()  
 elapsed\_time = end\_time - start\_time #en segundos

**Conclusión**

Con esta tarea fue posible aprender cómo analizar una red en general y cuál es el procedimiento aproximado que se debe realizar para conocer el desempeño de la red. Fue importante saber que debemos tomar varias muestras durante intervalos de tiempo significativos para poder observar variaciones en el desempeño de acuerdo a las diferentes situaciones que se pueden presentar a lo largo del día. Finalmente, pudimos comprobar nuevamente el potencial que tiene Python para realizar tareas específicas y la utilidad que representa para hacer de una manera más sencilla.

**Referencias**

* https://wisense.wordpress.com/2014/12/23/understanding-rssi-and-lqi-values-reported-by-the-cc1101/