ns-3 IoT

En este documento hablaremos sobre las modificaciones creadas en el simulador ns-3 con el fin de realizar experimentos con mdns y coAP.

Estructura y localización

ns-3 se encuentra situdado en /home/semaesoj/ns3/source/ns-3.26. A partir de ahora cuando hablemos de ns-3, asumiremos que nos encontramos en ese directorio.

Aquí podemos encontrar diferentes scripts que automatizan el lanzamiento de distintas simulaciones. El más importante y principal es **run**. Este script se ocupa de lanzar ns-3 con los argumentos proporcionados y una vez finalizada la simulación, comprime todos los resultados en un zip cuyo nombre son los distintos argumentos utilizados (más el tiempo unix de cuando se generó ese archivo comprimido) y limpia el directorio de las trazas generadas (una vez comprimidas).

El resto de scripts tan sólo se limitan a llamar a run con parametros determinados para generar experimentos en lotes. Todos ellos comprimen los zips generados por **run** en otro zip con el nombre de la *familia* de experimentos a la que pertenecen. Dichos scripts son:

- **mcast_cache.sh:** Envía las respuestas en multicast e incluye servicios de terceros.
- **mcast_cache_stime.sh:** Envía las respuestas en multicast, incluye servicios de terceros e intenta no contestar si otro nodo lo ha hecho primero.
- **mcast_nocache.sh:** Envía las respuestas en multicast.E
- **ucast_cache.sh:** Envía las respuestas en unicast.
- **ucast_nocache.sh:** Envía las respuestas en unicast, con el cache de servicios de anteriores consultas.
- **alltest.sh:** es un script que llama al resto de scripts de "lotes".

Estos scripts ejecutan lotes de simulaciones con los siguientes parametros:

- **Pruebas de intervalo**: Con una velocidad de 0m/s, se ejecutan simulaciones con descubrimientos cada 15,30,60,90 y 120.
- **Pruebas de velocidad**: Con un intervalo fijo de 30 segundos, ejecutamos simulaciones con 2m/s, 5m/s, 10m/s y 20 m/s.

Estos experimentos a su vez se ejecutan variando los protocolos de routing (nada, SMF, OLSR, OLSR+SMF) y el protocolo de descubrimiento (coAP o mDNS)

El código de la aplicación que se instala en los nodos de la simulación se encuentra en src/applications/model/coap e incluye los siguientes archivos:

- **coap_cache.cc:**
- **coap_mdns.cc:**
- **coap_rx.cc:**
- **coap_tools.cc:**
- **coap_tx.cc:**
- **coap_node.cc:**
- **coap_node.h:**
- **mdns.h:**

El código de la simulación se encuentra en la carpeta scratch bajo el nombre `coap.cc` este es el script que configura toda la simulación, el movimiento de los nodos, las direcciones Ip, la monitorización de la energía, etc.

Compilación

Cualquier modificación necesita ser recompilada para funcionar. Esto se hace automaticamente si al lanzar una simulación, la herramienta de gestión de dependencias (waf) detecta que el código fuente ha sido modificado. Sin embargo, es aconsejable que en vez de lanzar una simulación, se compile antes el código con el fin de evitar que un experimento falle por un simple error sintáctico. Esto se hace invocando al comando `make`.

Simulación

Como ya dijimos antes, nosotros recomendamos que las simulaciones sean ejecutadas a traves del script `run`. Sin embargo, y con el fin de dar la posibilidad al usuario de controlar de forma directa el simulador, explicaremos antes el comando "directo" que se ha de usar para ejecutar una simulación. Para invocar al script de la simualacion situado en la carpeta `scratch` deberemos usar la siguiente linea:

```
./waf --run coap
```

esta linea ejecutará la simulación con los parámetros estándar predefinidos en `scratch/coap.cc`. Si queremos sin embargo ver que parametros podemos modificar, debemos usar:

```
./waf --run "coap --PrintHelp"
```

La salida de este programa será algo similar a esto:

```
coap [Program Arguments] [General Arguments]

Program Arguments:
    --distance:       Distance between nodes (m) [500]
    --speed:          Speed of the nodes (m/s) [0]
    --gain:           Rx Gain (dB) [-10]
    --txpower:        Transmission Power (dB) [13]
    --energy:         Initial Energy (J)) [5000]
    --voltage:        Battery voltage (v) [5]
    --idlecurrent:    Current consumed when iddle (A) [0.015]
    --txcurrent:      Current consumed when transmitting (A) [0.12]
    --routing:        Current routing protocol 0=NONE,1=SMF,2=OLSR_SMF,3=OLSR,4=AODV_S
    --mdns:           1=mDNS 0=coAP discovery [0]
    --ping:           1=ping on 0=ping off [0]
    --cache:          1=cache on 0=cache off [0]
    --stime:          1=smart answer time on [0]
    --mcast:          1=mcast answers 0=ucast answers [0]
    --cacheinterval:  Determines the cache show up time [30]
    --verbose:        Verbosity level 0=script+SMF, 1=coap [0]
    --interval:       The time to wait between requests. If it is 0-> Server Mode. [60
    --maxAge:         Delete items after max-Age? If >0 it is the defaultage given to
    --runtime:        number of seconds the simulation will last [3000]

General Arguments:
    --PrintGlobals:              Print the list of globals.
    --PrintGroups:               Print the list of groups.
    --PrintGroup=[group]:        Print all TypeIds of group.
    --PrintTypeIds:              Print all TypeIds.
    --PrintAttributes=[typeid]:  Print all attributes of typeid.
    --PrintHelp:                 Print this help message.
```

Una simulación usal podrá ser, por ejemplo:

```
./waf --run "coap --routing=0 --ping=1 --verbose=1 --cache=0 --mcast=1 --stime=0 --mdn
```

Como hemos dicho antes, nosotros consideramos más cómodo usar **run** para ejecutar las simulaciones. El comando run es muy similar al comando que hemos visto anteriormente, puesto que es algo así como un wrapper que añade funciones adicionales.La simulación ejecutada anteriormente, por ejemplo, se escribiría así:

```
./run --routing=0 --ping=1 --verbose=1 --cache=0 --mcast=1 --stime=0 --mdns=1 --speed=
```

run también permite ver las opciones de las que dispone, tanto él como el script que ejecuta:

```
./run help
```

Y es util para eliminar trazas residuales de una simulación fallida, por ejemplo:

```
./run clean
```

Trazas generadas

Cada nodo es identificable por dos cosas. Su dirección IP y su ID de nodo. Una deficiente planificación del formato de las trazas ha hecho que se usen las dos notaciones en las trazas generadas durante las simulaciones. La correlación es sencilla: las direcciones IP comienzan en 192.168.1.1 y el ID en 0. por lo tanto `IP = 192.168.1.[ID+1]`

- **[IP_ADDR]_cache:** Muestra las entradas de la caché. La cabecera indica el tiempo en el que se imprime la caché, la IP del nodo al que pertenece y el número de nodos almacenados. Cada entrada muestra el instante de tiempo en el que caducará, la IP y la url correspondiente con el servicio.

```
CACHE_DUMP      5       10.1.1.17       24
90      10.1.1.16/temp
90      10.1.1.22/temp
90      10.1.1.21/temp
90      10.1.1.11/temp
90      10.1.1.23/temp
90      10.1.1.12/temp
90      10.1.1.6/temp
90      10.1.1.24/temp
90      10.1.1.1/temp
90      10.1.1.25/temp
90      10.1.1.2/temp
90      10.1.1.18/temp
```

- **coap_pcap-[ID]-0.pcap:** Capturas de tráfico.
- **energy_[ID].log**: Logs de energía. Guardan el tiempo en el que se ha capturado la lectura y el valor de la batería en Julios.

```
0.00533669    remaining energy=5000
0.00593208    remaining energy=5000
0.00600268    remaining energy=5000
0.00655269    remaining energy=5000
0.00746961    remaining energy=5000
0.00788075    remaining energy=5000
0.00799105    remaining energy=5000
0.00803439    remaining energy=5000
```

```
0.0088024    remaining energy=5000
0.00920705    remaining energy=5000
0.00944009    remaining energy=5000
0.00957839    remaining energy=5000
0.00972839    remaining energy=5000
0.0101292    remaining energy=4999.99
0.0106004    remaining energy=4999.99
0.0107304    remaining energy=4999.99
0.0109697    remaining energy=4999.99
0.0119464    remaining energy=4999.99
0.0133857    remaining energy=4999.99
```

- **routetables:** Tabla de rutas del algoritmo de routing.

```
Node: 21, Time: +5.0s, Local time: +5.0s, Ipv4ListRouting table
  Priority: 10 Protocol: ns3::smf::RoutingProtocol
  Priority: 5 Protocol: ns3::olsr::RoutingProtocol
Node: 21, Time: +5.0s, Local time: +5.0s, OLSR Routing table
Destination             NextHop         Interface       Distance
10.1.1.12               10.1.1.17               1               2
10.1.1.16               10.1.1.17               1               2
10.1.1.17               10.1.1.17               1               1
10.1.1.18               10.1.1.17               1               2
10.1.1.21               10.1.1.21               1               1
10.1.1.23               10.1.1.23               1               1
10.1.1.24               10.1.1.23               1               2
 HNA Routing Table: empty
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Node: 21, Time: +5.0s, Local time: +5.0s, Ipv4StaticRouting table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -      -   0
10.1.1.0        0.0.0.0         255.255.255.0   U     0      -      -   1
224.0.0.0       0.0.0.0         240.0.0.0       U     0      -      -   1
```

- **output.log:** Traca general que incluye registros de las transmisiones y recepciones de los nodos.

```
0.004099s 10.1.1.6 receive 64 bytes from 10.1.1.16:5683
        |-> TYPE:COAP_ACK CODE:UNKNOWN MID:17767
        |-> MAX-AGE: 90
        |-> URL: Localhost Service:temp
0.004099s 10.1.1.12 receive 64 bytes from 10.1.1.16:5683
        |-> TYPE:COAP_ACK CODE:UNKNOWN MID:17767
        |-> MAX-AGE: 90
        |-> URL: Localhost Service:temp
0.00522267s 10.1.1.11 receive 21 bytes from 10.1.1.21:5683
        |-> TYPE:COAP_CON CODE:COAP_GET MID:17767
        |-> DISCOVERY REQUEST
0.00522267s 10.1.1.11 send 105 bytes to 224.0.1.187:5683 id:17767
0.00522267s 10.1.1.17 receive 21 bytes from 10.1.1.21:5683
        |-> TYPE:COAP_CON CODE:COAP_GET MID:17767
        |-> DISCOVERY REQUEST
0.00522267s 10.1.1.17 send 105 bytes to 224.0.1.187:5683 id:17767
0.00522267s 10.1.1.21 receive 21 bytes from 10.1.1.21:5683
        |-> TYPE:COAP_CON CODE:COAP_GET MID:17767
        |-> DISCOVERY REQUEST
0.00522267s 10.1.1.21 send 105 bytes to 224.0.1.187:5683 id:17767
0.00592667s 10.1.1.1 receive 64 bytes from 10.1.1.16:5683
        |-> TYPE:COAP_ACK CODE:UNKNOWN MID:17767
        |-> MAX-AGE: 90
        |-> URL: Localhost Service:temp
0.00592667s 10.1.1.7 receive 64 bytes from 10.1.1.16:5683
        |-> TYPE:COAP_ACK CODE:UNKNOWN MID:17767
        |-> MAX-AGE: 90
        |-> URL: Localhost Service:temp
```

```
0.00655103s 10.1.1.17 receive 64 bytes from 10.1.1.22:5683
        |-> TYPE:COAP_ACK CODE:UNKNOWN MID:17767
```

- **state_[ID].log**: Almacena logs sobre los estados de wifi.

```
36.2452    state=CCA_BUSY start=+36243142836.0ns duration=+1864000.0ns
36.2452    state=IDLE start=+36245006836.0ns duration=+190000.0ns
36.2452    state=TX start=+36245196836.0ns duration=+1864000.0ns
36.2472    state=IDLE start=+36247060836.0ns duration=+114716.0ns
36.249    state=CCA_BUSY start=+36247175552.0ns duration=+1784000.0ns
36.249    state=IDLE start=+36248959552.0ns duration=+10976.0ns
36.2493    state=RX start=+36248970528.0ns duration=+304000.0ns
36.2494    state=IDLE start=+36249274528.0ns duration=+175397.0ns
36.2512    state=CCA_BUSY start=+36249449925.0ns duration=+1776000.0ns
36.2512    state=IDLE start=+36251225925.0ns duration=+12061.0ns
36.2514    state=CCA_BUSY start=+36251237986.0ns duration=+57213.0ns
36.2514    state=IDLE start=+36251295199.0ns duration=+100007.0ns
36.2517    state=CCA_BUSY start=+36251395206.0ns duration=+146780.0ns
36.2517    state=IDLE start=+36251541986.0ns duration=+110296.0ns
36.2534    state=RX start=+36251652282.0ns duration=+1760000.0ns
36.2534    state=IDLE start=+36253412282.0ns duration=+10000.0ns
36.2534    state=TX start=+36253422282.0ns duration=+304000.0ns
36.2539    state=IDLE start=+36253726282.0ns duration=+166066.0ns
36.2553    state=CCA_BUSY start=+36253892348.0ns duration=+202046.0ns
36.2553    state=IDLE start=+36254094394.0ns duration=+1192558.0ns
36.2558    state=CCA_BUSY start=+36255286952.0ns duration=+304000.0ns
36.2558    state=IDLE start=+36255590952.0ns duration=+255395.0ns
```