Tutorial 6 – Contiki 6LoWPAN Networking

Aim

Contiki is an event based operating system that designed to operate on resource scarce and low power embedded system platforms. This tutorial shows you how to use 6loWPAN Networking communications stack of Contiki. Using Contiki, you will be able to form multi-hop IP networks consisting of different motes communicating over shared protocols. However, it does provide significant interoperability with other IP networks. You will learn how to use IP-protocols for link communication (mote-to-mote, or broadcast) and also how to install and configure a IPv6 subnet enabling point-to-point multihop communication.

NOTE: this tute will not cover the basics of UDP or TCP.

6lowpan/IPv6 Basics

A complete introduction to IPv6 network topology and addressing are beyond the scope of this tutorial; unfamiliar readers may wish to familiarize themselves with some of the concepts at play here.

hosts vs. routers: in an IP network, devices may participate as *hosts* or *routers.* Generally, *hosts* do not forward packets or participate in routing protocols, while *routers* do. Every mote running the network stack functions as a router and is capable of forwarding packets and making routing decisions. Motes thus form a multihop IP subnetwork.

subneting: The smallest unit of network management in IPv6 is the subnet. A subnet consists of a number of motes (*node routers*) and one or more higher function devices (*edge routers*) which perform a number of other routing functions for the network. The edge routers also may provide routing to other networks. In terms of network topology, these subnets are generally configured as stub networks.

MTU: the Maximum Transfer Unit refers to the largest payload which may be sent in a link-layer data frame. Most node routers use the IEEE 802.15.4 physical layer, which limits the MTU to around 100 octets. To provide the larger payloads to upper-layer protocols, 6lowpan "layer 2.5" fragmentation is implemented; as a result, the maximum size IP-layer datagram is 1280 octets.

Acknowledgement: Dr. Matt D' Souza (University of Queensland), Dr. Brano Kusy (CSIRO)

IPv6 Addressing

An IPv6 address is 64 bits long and is represented as (in hex):

```
xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
```

Zeros omitted: Any blocks of zeros are omitted (0x0000) and replaced with ::

Eg. `abcd:2601:3456:ecf1:0000:0000:0000:2016`

Becomes:

```
abcd:2601:3456:ecf1::2016
```

6lowpan/IPv6 Routing

Commissioning a simple IPv6 sub-network requires some extra work involving programming motes and running drivers on a PC.

[Make sure you erase your SensorTag everytime before you download a new code.]

Step 1: Install a RPL Border Router

The RPL Border Router acts an router and SLIP modem device, when connected to a PC. Program the sensortag with rpl-border-router example from contiki-examples/networking/rpl-border-router

Step 2: Compile the tunslip6 program

The tunslip6 program will allow the RPL Border Router to act as a tun device. You will need to compile the tunslip6 program. Go to the contiki-git/tools folder and run make.

```
cd contiki-git/tools
make tunslip6
```

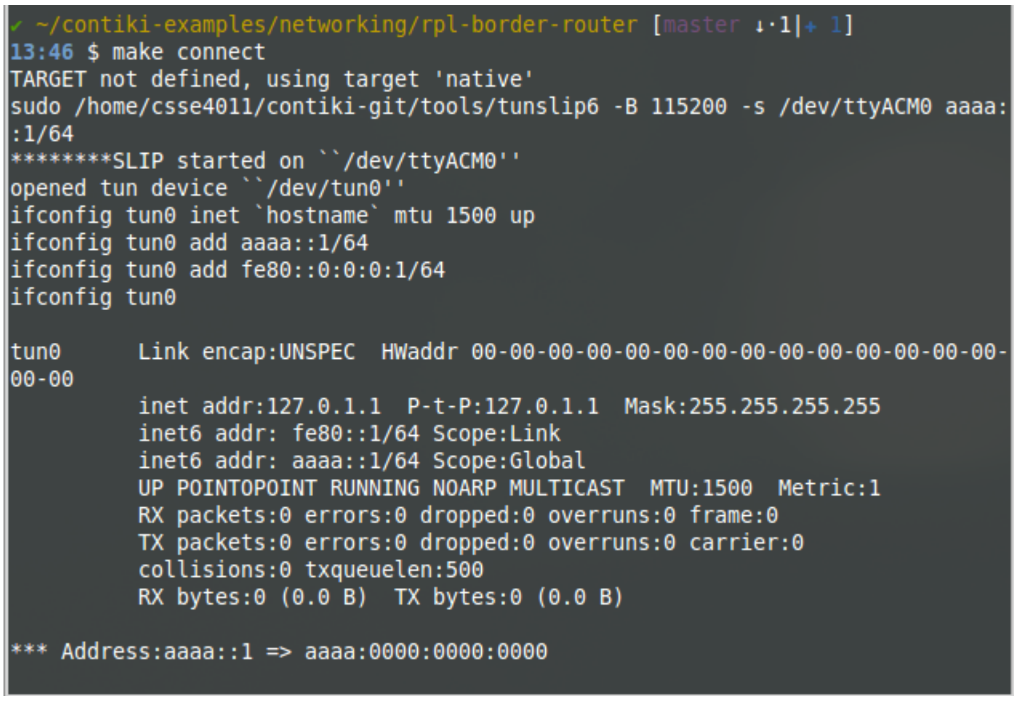
Step 3: Run the tunslip6 program

In the contiki-examples/networking/rpl-border-router, Run the tunslip6 program by typing:

```
cd ~/contiki-examples/networking/rpl-border-router
make TARGET=srf06-cc26xx BOARD=sensortag/cc2650
sudo make prog
make connect
```

This will run the tunslip6 program in the terminal window (see below). Leave this window open and use another window to continue.

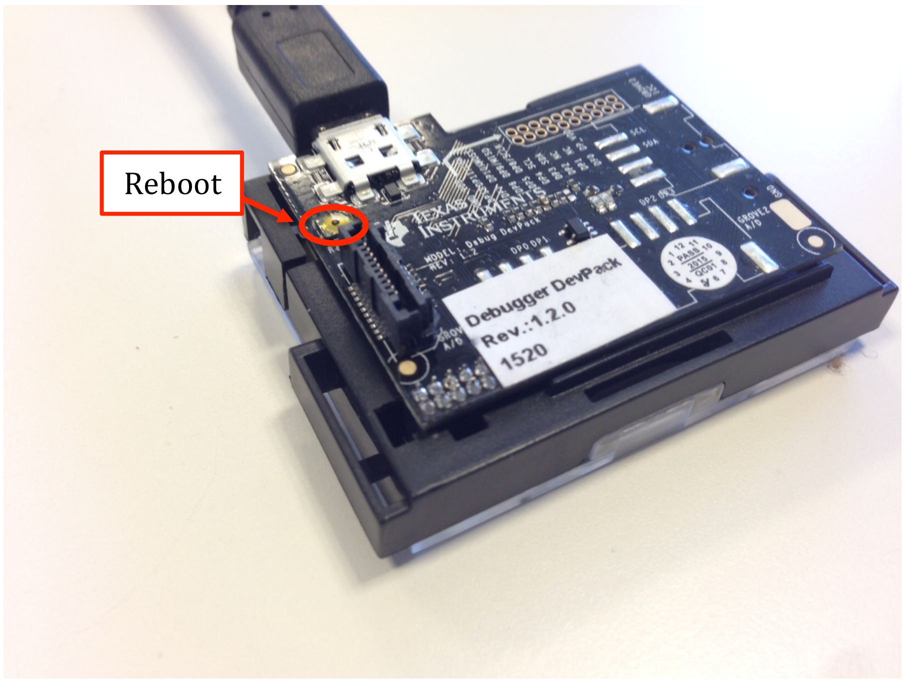
Acknowledgement: Dr. Matt D' Souza (University of Queensland), Dr. Brano Kusy (CSIRO)



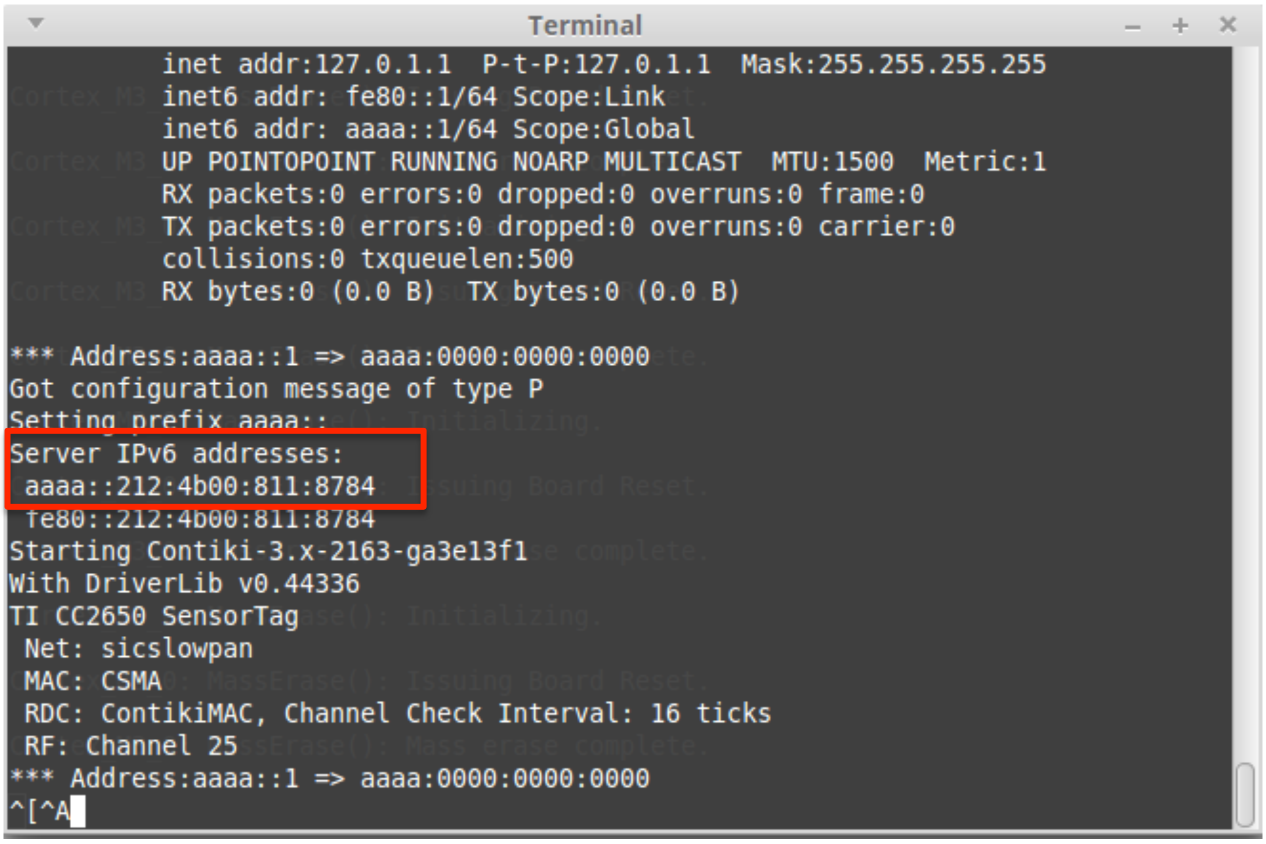
```
✓ ~/contiki-examples/networking/rpl-border-router [master ↓·1|+ 1]
13:46 $ make connect
TARGET not defined, using target 'native'
sudo /home/csse4011/contiki-git/tools/tunslip6 -B 115200 -s /dev/ttyACM0 aaaa:
:1/64
********SLIP started on ``/dev/ttyACM0''
opened tun device ``/dev/tun0''
ifconfig tun0 inet `hostname` mtu 1500 up
ifconfig tun0 add aaaa::1/64
ifconfig tun0 add fe80::0:0:0:1/64
ifconfig tun0

tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-
00-00
          inet addr:127.0.1.1  P-t-P:127.0.1.1  Mask:255.255.255.255
          inet6 addr: fe80::1/64 Scope:Link
          inet6 addr: aaaa::1/64 Scope:Global
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

*** Address:aaaa::1 => aaaa:0000:0000:0000
```

tunslip6 Runing

Press the small yellow button near the miniUSB on the DevPack board to reboot the SensorTag, then you can see the following information which contains the IPv6 address of your SensorTag, **write down your IPv6 address**, you will need it in the following experiments.



Acknowledgement: Dr. Matt D' Souza (University of Queensland), Dr. Brano Kusy (CSIRO)



```
          inet addr:127.0.1.1  P-t-P:127.0.1.1  Mask:255.255.255.255
          inet6 addr: fe80::1/64 Scope:Link
          inet6 addr: aaaa::1/64 Scope:Global
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

*** Address:aaaa::1 => aaaa:0000:0000:0000
Got configuration message of type P
Setting prefix aaaa::
Server IPv6 addresses:
 aaaa::212:4b00:811:8784
 fe80::212:4b00:811:8784
Starting Contiki-3.x-2163-ga3e13f1
With DriverLib v0.44336
TI CC2650 SensorTag
 Net: sicslowpan
 MAC: CSMA
 RDC: ContikiMAC, Channel Check Interval: 16 ticks
 RF: Channel 25
*** Address:aaaa::1 => aaaa:0000:0000:0000
^[^A
```

NOTE: If you need to run the actual command for the sensortag, then try:

```
Sudo ~/contiki-git/tools/tunslip6 -B 115200 -s /dev/ttyACM0 aaaa::1/64
```

Networking Tools – nc6, ping6, ifconfig, route and WireShark

Commonly used networking tools are netcat (nc6), ping6, ifconfig, route and wireshark. You will be using these tools.
[Note that you don't need to run the following code during the lab, just be familiar with these commands]

- **ping6**

Ping6 is a command-line tool that sends an ICMP (Internet Control Message Protocol) echo request packet and is used to determine if a connection to an IP address exists.

```
ping6 aaaa::212:4b00:afe:a684
```

Change a684 to the address of your node.

Hint: You may find out your node's address by resetting the sensortag (leave tunslip6 running). Some sensortags might need to be reconnected to function normally after resetting (please remember to rerun tunslip6 program because it will be terminated after the usb cable is disconnected from the sensortag).

- **ifconfig**

ifconfig is a command-line tool that is used to control your networking interfaces. Your router node will appear as a TUN (Network **TUN**nel) modem device in ifconfig.

```
ifconfig tun0
```

- **netcat (nc6)**

netcat is a command-line tool that allows you to open TCP and UDP connections. To open a TCP connection on address [aaaa::212:4b00:afe:a684], port 80, type

```
$ nc6 aaaa::212:4b00:afe:a684 80 < req.txt
```

```
$cat req.txt
GET / HTTP/1.0
```

To install netcat (nc6)

```
$ sudo apt-get install netcat6
```

To open a UDP connection on address [aaaa::212:4b00:afe:a684], port 3000, type (NOTHING WILL HAPPEN BECAUSE rpl-border-router node does not respond to UDP port 3000).

```
$ nc6 –u aaaa::212:4b00:afe:a684 3000
```

- **Route**

Route is a command-line tool that is used to view and control the IP networking routing table. Use the following command to display all routes for IPv6:

```
route –-inet6
```

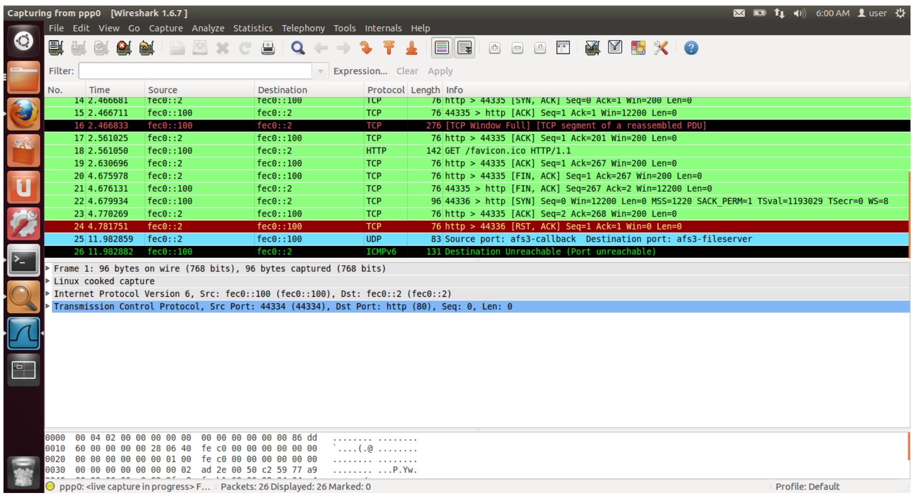
- **Wireshark**

Wireshark is graphical tool that can be used to view network traffic. To install wireshark, type:

```
$ sudo apt-get install wireshark
```

You must run wireshark with root privalages. Select “List captured interfaces” icon (far left of screen) and select tun0, to view the network traffic from the rpl-border-router nodes.

Acknowledgement: Dr. Matt D' Souza (University of Queensland), Dr. Brano Kusy (CSIRO)


Capturing from ppp0 [Wireshark 1.6.7]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply
6:00 AM user

```
No.  Time       Source     Destination Protocol Length Info
  14 2.466681   fec0::2    fec0::100   TCP        76 http > 44335 [SYN, ACK] Seq=0 Ack=1 Win=200 Len=0
  15 2.466711   fec0::100  fec0::2     TCP        76 44335 > http [ACK] Seq=1 Ack=1 Win=12200 Len=0
  16 2.466833   fec0::100  fec0::2     TCP       276 [TCP Window Full] [TCP segment of a reassembled PDU]
  17 2.561025   fec0::2    fec0::100   TCP        76 http > 44335 [ACK] Seq=1 Ack=201 Win=200 Len=0
  18 2.561050   fec0::100  fec0::2     HTTP      142 GET /favicon.ico HTTP/1.1
  19 2.630696   fec0::2    fec0::100   TCP        76 http > 44335 [ACK] Seq=1 Ack=267 Win=200 Len=0
  20 4.675978   fec0::2    fec0::100   TCP        76 http > 44335 [FIN, ACK] Seq=1 Ack=267 Win=200 Len=0
  21 4.676131   fec0::100  fec0::2     TCP        76 44335 > http [FIN, ACK] Seq=267 Ack=2 Win=12200 Len=0
  22 4.679934   fec0::100  fec0::2     TCP        96 44336 > http [SYN] Seq=0 Win=12200 Len=0 MSS=1220 SACK_PERM=1 TSval=1193029 TSecr=0 WS=8
  23 4.770269   fec0::2    fec0::100   TCP        76 http > 44335 [ACK] Seq=2 Ack=268 Win=200 Len=0
  24 4.781751   fec0::2    fec0::100   TCP        76 http > 44336 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
  25 11.982859  fec0::2    fec0::100   UDP        83 Source port: afs3-callback  Destination port: afs3-fileserver
  26 11.982882  fec0::100  fec0::2     ICMPv6    131 Destination Unreachable (Port unreachable)

▸ Frame 1: 96 bytes on wire (768 bits), 96 bytes captured (768 bits)
▸ Linux cooked capture
▸ Internet Protocol Version 6, Src: fec0::100 (fec0::100), Dst: fec0::2 (fec0::2)
▸ Transmission Control Protocol, Src Port: 44334 (44334), Dst Port: http (80), Seq: 0, Len: 0

0000  00 04 02 00 00 00 00 00  00 00 00 00 00 00 86 dd   ........ ........
0010  60 00 00 00 00 28 06 40  fe c0 00 00 00 00 00 00   `....(.@ ........
0020  00 00 00 00 00 00 01 00  fe c0 00 00 00 00 00 00   ........ ........
0030  00 00 00 00 00 00 00 02  ad 2e 00 50 c2 59 77 a9   ........ ...P.Yw.

ppp0: <live capture in progress> F...   Packets: 26 Displayed: 26 Marked: 0   Profile: Default
```

TCP

TCP is the standard Internet protocol for reliable, in-order delivery of data across the network. Although inefficient, its ubiquity makes it impossible to ignore; thus, UIP provides a very simple TCP stack. TCP is considerably more complicated then UDP, and requires careful use in the embedded setting to prevent resource exhaustion.

The TCP examples can be found in contiki-examples/networking/tcp-server folder.

TCP Server

The TCP Server is an echo server. It will receive a string input on port 8080 and respond with the same string. The TCP server must be connected to the RPL Border Router. The TCP server can be accessed using netcat6, to send and receive strings.

RPL Border Router

The TCP Socket will connect to the RPL border router. You can ping the TCP client and server from the RPL border router.

```
ping6 <address of sensortag>
```

You can connect to the TCP socket using netcat6 on port 8080.

```
nc6 <address of sensortag> 8080
```

Below is for Lab 3.

[Next, you will need to team up with one fellow student. One SensorTag functions as the server, and another one is the client.]

UDP

Acknowledgement: Dr. Matt D' Souza (University of Queensland), Dr. Brano Kusy (CSIRO)

The UDP examples can be found in contiki-examples/networking/udp-ipv6 folder.

UDP Server
The UDP Server will connect to the UDP server and periodically send a string to the UDP Server on port 3001. The UDP Server will listen on port 3000. In the makefile change:
CONTIKI_PROJECT=udp-server

UDP Client
The UDP Client will connect to the UDP server and periodically send a string to the UDP Server on port 3000. The UDP Client will listen on port 3001. In the makefile change:
CONTIKI_PROJECT=udp-client

RPL Border Router
The UDP Clients and Server will connect to the RPL border router. You can ping the UDP client and server from the RPL border router.
[Note that there is only one RPL border router set up by tutor, the tutor will ping your server and client.Tell the tutor your Ipv6 address, he will confirm if you have set up the server/client successfully or not.]

```
ping6 <address of sensortag>
```

Conclusion

This lesson has introduced the basics of networking with the Contiki 6LoWPAN library.

Related Documentation

- Contiki Wiki: https://github.com/contiki-os/contiki/wiki

Acknowledgement: Dr. Matt D' Souza (University of Queensland), Dr. Brano Kusy (CSIRO)