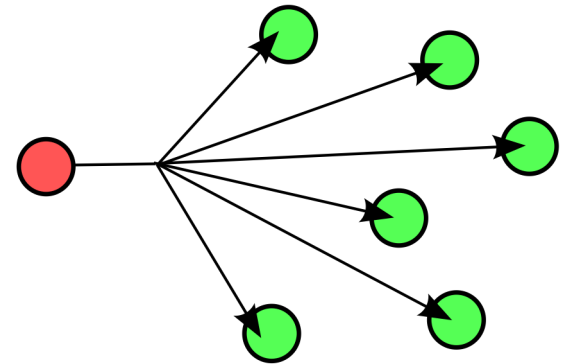


Broadcast Communication Tutorial

What is Broadcast?

- Broadcast is used to describe a type of communication in which a piece of information is sent from one node to all the other nodes in a network
- One-to-one communication, called unicast, is typical for wired networks, but broadcast is more common in the case of wireless networks



Rime Network Stack

- Rime is a light-weight layered communication stack for sensor networks that provides a set of communication primitives ranging from best-effort anonymous local area broadcast to reliable network flooding
 - The purpose of Rime is to simplify implementation of sensor network protocols and facilitate code reuse
 - For details, refer to: <http://contiki.sourceforge.net/docs/2.6/a01798.html>
- Contiki includes the Rime stack, in addition to the IPv4 and IPv6 stacks, as an alternative for low-power wireless networks, which don't require all the functionality (and complexity) of the traditional stacks

Rime Implementation in Contiki

- The source files for the Rime stack are available in “contiki/core/net/rime”
- Use examples for Rime can be found in the directory “contiki/examples/rime”
 - broadcast
 - unicast
 - collect
 - mesh
 - multihop

Broadcast Simulation Example

- The simplest way to open the simulation is to select “Broadcast Simulation” in the IoTrain-Sim interface
- Alternatively, you can open it manually as follows
 - Open Cooja
 - Click File > Open simulation > Browse...
 - Go to the folder “iotrain-sim/database/fundamental_training/networking/broadcast/simulation”
 - Select the file “broadcast.csc”
- The simulation control window will appear showing 10 motes in the Network panel (see next page)
 - Click the “Start” button to begin the simulation
 - Communication will be visualized in the Network panel, and traffic details will be shown in the Mote output panel

Broadcast Simulation in Cooja

Broadcast example - Cooja: The Contiki Network Simulator

File Simulation Motes Tools Settings Help

Network

View Zoom

Simulation control

Run Speed limit

Start Pause Step Reload

Time: 29:05.617
Speed: ---

Mote output

Time	Mote	Message
23:58.373	ID:1	broadcast message sent
23:58.593	ID:3	broadcast message sent
23:59.031	ID:6	broadcast message sent
24:00.356	ID:9	broadcast message sent
24:00.861	ID:7	broadcast message sent
24:00.864	ID:5	broadcast message received from 7.0: 'Wang'
24:01.274	ID:10	broadcast message sent
24:03.706	ID:4	broadcast message sent
24:04.136	ID:5	broadcast message sent
24:04.540	ID:8	broadcast message sent
24:04.742	ID:3	broadcast message sent
24:05.313	ID:2	broadcast message sent
24:05.998	ID:1	broadcast message sent
24:06.106	ID:9	broadcast message sent
24:06.796	ID:6	broadcast message sent
24:07.322	ID:7	broadcast message sent
24:08.206	ID:4	broadcast message sent

Filter:

Source Code Commentary

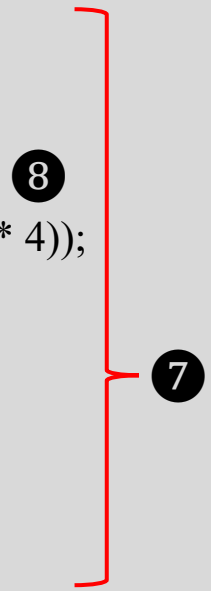
- Each node sends a string via a broadcast packet
 - Source code: iotrain-sim/database/fundamental_training/networking/broadcast/simulation/broadcast-ex.c *

```
#include "contiki.h" ❶
#include "net/rime/rime.h" ❷
#include "random.h"
#include "dev/button-sensor.h"
#include "dev/leds.h"
#include <stdio.h> ❸
/*-----*/
PROCESS(example_broadcast_process, "Broadcast example"); ❹
AUTOSTART_PROCESSES(&example_broadcast_process); ❺
/*-----*/
static void
broadcast_recv(struct broadcast_conn *c, const linkaddr_t *from) ❻
{
    printf("broadcast message received from %d.%d: '%s'\n", from->u8[0], from->u8[1], (char *)
packetbuf_dataptr());
}
static const struct broadcast_callbacks broadcast_call = {broadcast_recv};
static struct broadcast_conn broadcast;
```

* The file name is not "broadcast.c" to avoid a conflict with the Rime broadcast implementation

Source Code Commentary (cont.)

```
/*-----*/  
PROCESS_THREAD(example_broadcast_process, ev, data)  
{  
    static struct etimer et;  
    PROCESS_EXITHANDLER(broadcast_close(&broadcast);)  
    PROCESS_BEGIN();  
    broadcast_open(&broadcast, 129, &broadcast_call);  
    while(1) {  
        /* Delay 2-4 seconds */  
        etimer_set(&et, CLOCK_SECOND * 4 + random_rand() % (CLOCK_SECOND * 4));  
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));  
        packetbuf_copyfrom("Hello", 6);  
        broadcast_send(&broadcast);  
        printf("broadcast message sent\n");  
    }  
    PROCESS_END();  
}  
/*-----*/
```



8

7

Source Code Commentary (cont.)

- ① Header file needed by Contiki applications
- ② Header file needed for the Rime stack
- ③ Header file needed for the `printf()` function
- ④ Name for the application process
- ⑤ Automatically start the application process
- ⑥ Callback function that is invoked when a broadcast packet is received; the function prints information about the packet sender and received data
 - The first argument is of type `broadcast_conn *`, and contains information about the connection and receive and send functions
 - The second argument is of type `linkaddr_t *`, and contains information about the sender
 - For details, refer to “`contiki/core/net/rime/broadcast.c`” and “`core/net/rime/broadcast.h`”

Source Code Commentary (cont.)

- 7 Operations for broadcast communication
 - Open a best-effort broadcasting connection on a given UDP port, with the callback function given as argument to be called when a packet is received on this connection
`broadcast_open(struct broadcast_conn *, uint16_t, const struct broadcast_callbacks *)`
 - Send a broadcast packet on an already open connection (data must be prepared in advance via a call to the `packetbuf_copyfrom()` function)
`broadcast_send(struct broadcast_conn *)`
 - Close an open best-effort broadcast connection
`broadcast_close(struct broadcast_conn *)`
- 8 Use a timer to introduce a random delay of 2 to 4 seconds between the broadcast packets

Exercise I

- Modify the source code to alter the content of the packet that is broadcasted in the provided example
- Hints
 - Locate the function `packetbuf_copyfrom()` in the source code file “broadcast-ex.c”, and replace the string “Hello” with another word
 - You must change the string length argument to a value equal to the length of the new word + 1, in order to account for the null character at the end of the string

Example: `packetbuf_copyfrom("Bye", 4)`

Exercise II

- Try other source code samples located in the Rime directory “contiki/examples/rime” to understand their behavior