The Test of Contiki-based Protocol Stack of 6LoWPAN

Liu GuangDi *
Library of Chengdu University
Chengdu, China
E-mail: liuguangdi1103@126.com
* Corresponding Author

Tie Ling
School of Information Science and Technology
Chengdu, China
E-mail: 2396565376@qq.com

Abstract—Propose a 6LoWPAN protocol stack test scheme based on the Contiki operating system [1]. Use the simulation tool cooja to test the performance of 6LoWPAN protocol [2]. Analyze the network connectivity, energy consumption, timeline when the header compression algorithm, packet fragmentation and rest ruction were introduced into the scheme.

Keywords-Component; 6LoWPAN Protocol Stack; Contiki Operate Systerm; Cooja; Wireless Sensor Networks; IPv6 Stack Protocol;

I. WIRELESS SENSOR NETWORK BASED ON 6LOWPAN

This article performs the test of 6LoWPAN protocol stack based on contiki system. The simulation results show that the protocol stack can achieve normal transmission of the packet in wireless sensor networks.

II. THE SIMULATION OF THE 6LOWPAN STACK

6LoWPAN technology is actually an IPv6 protocol stack of wireless sensor network. It constructs an adaptation layer to compress header. IPv6Dispatch field is added to IPv6 header. The compress_hdr_ipv6 () function is main processing function [4].

1280 bytes are minimum MTU in IPv6 link layer; however, the link layer provides only 127 bytes in IEEE802.15.4 standard, so adaptation layer must be added into 6LoWPAN stack. It uses runtime packet fragmentation and reassembles to solve the problem of large packet transmitted.

III. THE SIMULATION PLATFORM OF 6LOWPAN

A. *Cooja Simulation Platform*

Cooja is the simulator in the Contiki operation system. It supports cross-layer simulation based on instruction level, system level and application level. In this paper, the test environment topology is shown in Fig. 1. The completed 6LoWPAN protocol stack is installed in each node. The node 1 and 2 communicate with node 3. The communication mode is based on Client/server. udp-client.c program is running in the client; udp-server.c program is running in the server [5].

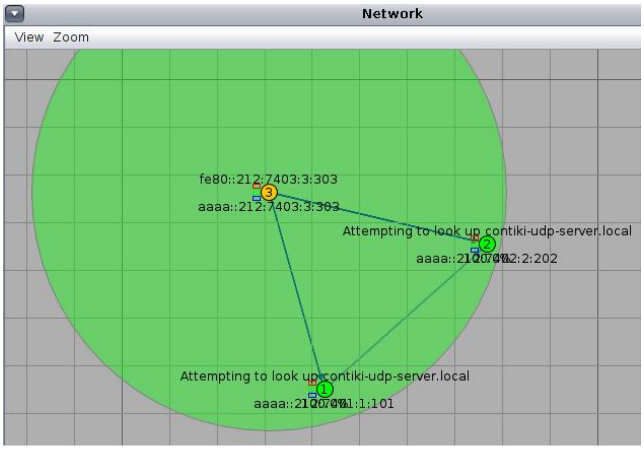
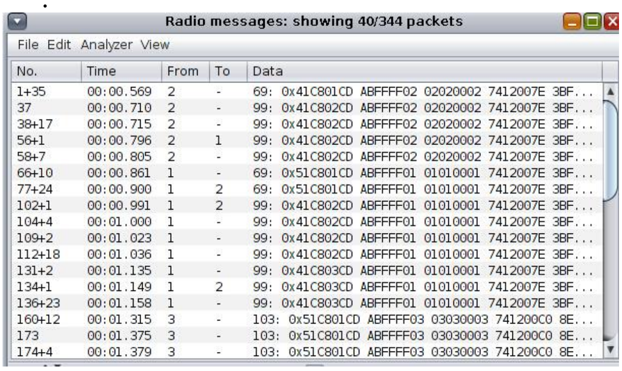


Figure 1. Test Network Topology

B. *Packet fragmentation and reassembly*

The data packets transmitted among the nodes 1, 2 and node 3 may be larger than the maximum transmission unit. The fragmentation will be built in the sender and restruction is performed in the received part. The test result is shown in Fig. 2.

.



Radio messages: showing 40/344 packets

File Edit Analyzer View

No.	Time	From	To	Data
1+35	00:00.569	2	-	69: 0x41C801CD ABFFFF02 02020002 7412007E 3BF...
37	00:00.710	2	-	99: 0x41C802CD ABFFFF02 02020002 7412007E 3BF...
38+17	00:00.715	2	-	99: 0x41C802CD ABFFFF02 02020002 7412007E 3BF...
56+1	00:00.796	2	1	99: 0x41C802CD ABFFFF02 02020002 7412007E 3BF...
58+7	00:00.805	2	-	99: 0x41C802CD ABFFFF02 02020002 7412007E 3BF...
66+10	00:00.861	1	-	69: 0x51C801CD ABFFFF01 01010001 7412007E 3BF...
77+24	00:00.900	1	2	69: 0x51C801CD ABFFFF01 01010001 7412007E 3BF...
102+1	00:00.991	1	2	99: 0x41C802CD ABFFFF01 01010001 7412007E 3BF...
104+4	00:01.000	1	-	99: 0x41C802CD ABFFFF01 01010001 7412007E 3BF...
109+2	00:01.023	1	-	99: 0x41C802CD ABFFFF01 01010001 7412007E 3BF...
112+18	00:01.036	1	-	99: 0x41C802CD ABFFFF01 01010001 7412007E 3BF...
131+2	00:01.135	1	-	99: 0x41C803CD ABFFFF01 01010001 7412007E 3BF...
134+1	00:01.149	1	2	99: 0x41C803CD ABFFFF01 01010001 7412007E 3BF...
136+23	00:01.158	1	-	99: 0x41C803CD ABFFFF01 01010001 7412007E 3BF...
160+12	00:01.315	3	-	103: 0x51C801CD ABFFFF03 03030003 741200C0 8E...
173	00:01.375	3	-	103: 0x51C801CD ABFFFF03 03030003 741200C0 8E...
174+4	00:01.379	3	-	103: 0x51C801CD ABFFFF03 03030003 741200C0 8E...

Figure 2. Radio Messages

C. *Energy Consumption Test*

The power consumption results are shown in Fig. 4-7. The highest level of power consumption is highlighted based on Power Tracker [6].

Fig. 3-6 shows that the energy consumption of node 2 is the highest each time. The answer can be found in Fig. 1, since the distance between the node 2 and node 3 is the longest, it requires higher power consumption to transmit the data packets. Further we know that the energy consumption is high when the node has just started, the energy consumption decreased when the connection is established.

PowerTracker: 3 motes

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Sky 1	51.33%	19.04%	17.49%
Sky 2	51.47%	16.92%	19.63%
Sky 3	50.49%	17.65%	17.45%
AVERAGE	51.09%	17.87%	18.19%

Print to console/Copy to clipboard Reset

Figure 3. Node Energy State When Time= 5.446s

PowerTracker: 3 motes

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Sky 1	29.51%	11.80%	9.31%
Sky 2	30.78%	12.05%	9.93%
Sky 3	28.19%	9.08%	10.36%
AVERAGE	29.49%	10.98%	9.87%

Print to console/Copy to clipboard Reset

Figure 4. Node Energy State When Time= 12.385s

PowerTracker: 3 motes

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Sky 1	16.80%	6.63%	5.10%
Sky 2	18.26%	7.71%	5.12%
Sky 3	16.37%	5.52%	5.50%
AVERAGE	17.14%	6.62%	5.24%

Print to console/Copy to clipboard Reset

Figure 5. Node Enegy State When Time= 24.412s

PowerTracker: 3 motes

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
Sky 1	10.34%	3.99%	3.02%
Sky 2	11.13%	4.58%	3.03%
Sky 3	10.07%	3.30%	3.27%
AVERAGE	10.51%	3.96%	3.11%

Print to console/Copy to clipboard Reset

Figure 6. Node Energy State When Time= 41.316s

D. *TimeLine Test*

Cooja TimeLine shows a time line for each node in the simulation in Fig. 7. On the time line, the power state of the radio transceiver of each node is shown in a color code: It is white if the transceiver is off, gray if it is on. Radio transmissions and receptions are shown in the same time line: transmissions are blue and receptions are green. When two simultaneous transmissions are reaching to a node, it results in radio interference, red is shown. Transmissions and receptions are shown with bit granularity. See Fig. 7 for a Cooja screenshot with a TimeLine. [7]

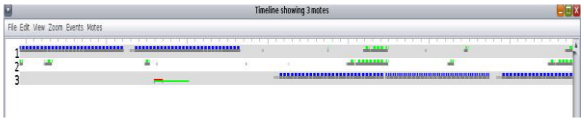


Figure 7. Timeline When Node Just Started

E. *UDP Packet Transmission Test*

Experiments are performed to show the UDP datagram transfer with the IPv6 address [8], in Fig. 8. Circular area represents the coverage of node 3 network. Nodes 1 and 2 are clients, node 3 is server. After the two nodes establish socket connection, they continuous send "Hello" string to node 3 by UDP protocol. Test results show that the wireless sensor network node can transfer the UDP datagram among them using the 6LoWPAN protocol stack. Fig. 8 shows the process that 3 nodes send message to each other and the process of the acknowledgment are covered by the node 1, 2 and node 3.

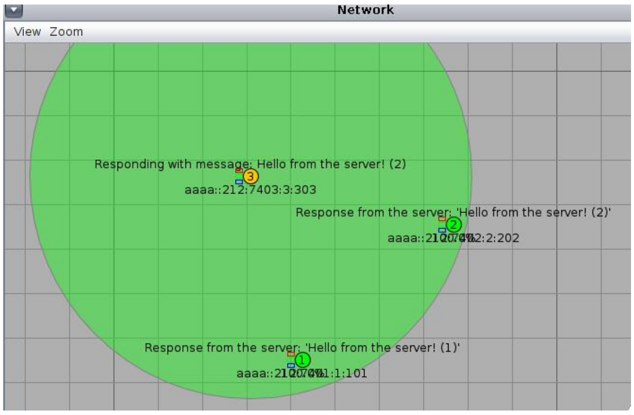


Figure 8. Process of Sending and Receiving Data

Mote output

File Edit View

Time	Mote	Message
00:01.208	ID:3	UDP server started
00:01.214	ID:3	Server IPv6 addresses: aaaa::212:7403:3:303
00:01.217	ID:3	fe80::212:7403:3:303
00:01.946	ID:2	Still looking up "contiki-udp-server.local"...
00:03.468	ID:1	Still looking up "contiki-udp-server.local"...
00:03.770	ID:1	Lookup of "contiki-udp-server.local" succeded!
00:03.770	ID:2	Lookup of "contiki-udp-server.local" succeded!
00:03.778	ID:1	Created a connection with the server fe80::212:7403:3:303 local/remo...
00:03.778	ID:2	Created a connection with the server fe80::212:7403:3:303 local/remo...
00:18.782	ID:1	Client sending to: fe80::212:7403:3:303 (msg: Hello 1 from the client)
00:18.784	ID:2	Client sending to: fe80::212:7403:3:303 (msg: Hello 1 from the client)
00:33.790	ID:1	Client sending to: fe80::212:7403:3:303 (msg: Hello 2 from the client)
00:33.792	ID:2	Client sending to: fe80::212:7403:3:303 (msg: Hello 2 from the client)
00:33.823	ID:3	Server received: 'Hello 2 from the client' from fe80::212:7401:1:101
00:33.828	ID:3	Responding with message: Hello from the server! (1)
00:33.929	ID:1	Response from the server: 'Hello from the server! (1)'
00:33.948	ID:3	Server received: 'Hello 2 from the client' from fe80::212:7402:2:202
00:33.952	ID:3	Responding with message: Hello from the server! (2)
00:34.035	ID:2	Response from the server: 'Hello from the server! (2)'

Figure 9. UDP Packet Transmission Test.

After node 1 and node 2 is powered, the protocol stack started first; then MAC layer is monitored and addresses allocated was followed next, last , the UDP client program is started and IPv6 address is assigned. See Fig. 9, the nodes will stop sending the message ‘look up ‘contiki-udp-server local’ until it was created a connection with the server.[9]

IV. CONCLUSIONS

This paper simulates the basic functions of 6LoWPAN protocol stack. [10] We tests functions of datagram fragmentation and reassembly, energy consumption, timeline and UDP data transmission by cooja tools, and so on. Experimental results show that the proposed scheme is able to complete the normal process of networking and accurate transmission of data, with some research and application value.

ACKNOWLEDGMENT

This research work is supported by the Chengdu University foundation project (No.2012XJZ07); Sichuan Province Science and technology support program (No.2013GZ0016); the open foundation of Key Laboratory about Pattern recognition and intelligent information processing.

REFERENCES

[1] Information on http://www.contiki-os.org/

[2] Fredrik Osterlind, Joakim Eriksson, Adam Dunkels: Demo Abstract: Cooja TimeLine: A Power Visualizer for Sensor Network Simulation (Zurich, Switzerland 2010).

[3] Hui J,ed. draft-ietf-6lowpan-hc,"Compression Format for IPv6 Datagrams in 6LoWPAN Networks" [S/OL].Available from:www.ietf.org/.

[4] Changlong Li: Research of 6LoWPAN Technology and Implementation of Header Compression (2011).

[5] D. Yazar and A. Dunkels. Efficient Application Integration in IP-Based Sensor Networks. In Proceedings of the ACM BuildSys 2009 workshop, in conjuction with ACM SenSys 2009, November 2009.

[6] RFC6775.Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks [OL].http://www.ietf.org/mail-archive/web/ieff-announce/current/msg10866.html, 2013.

[7] Abeill J,Mathide Durvy,Dawson-Haggerty. Lightweight IPv6 Stacks for Smart Objects:the Experience of Three Independent and Interoperable Implementations [J/OL]. http://wenku.baidu.com/view/1907d8cd05087632311212b6.html, 2013.

[8] MATTHIAS Kovatsch,SIMON Duquennoy,ADAM Dunkels. A low-power CoAP for Contiki [A].Valencia,Spain:IEEE, 2011.855-860.

[9] KOVATSCH M. Demo abstract:human-CoAP interaction with copper [A]. Barcelona,Spain: IEEE, 2011.27-29

[10] Olfa Gaddour, Anis Koubaa. RPL in a nutshell: A survey [J]. Computer Networks . 2012 (14)

