제21회 한국정보처리학회 춘계학술발표대회 논문집 제11권 제1호 (2004. 5)

블루투스를 이용한 보안을 위한 무선 센서네트워크의 구현

김재완, 김병국, 엄두섭
고려대학교 전자컴퓨터공학과
e-mail : kuzzang@final.korea.ac.kr

An implementation of wireless sensor network for security system using Bluetooth

Jae-Wan Kim*, Byoung-Kug Kim*, Doo-Seop Eom*
*Dept. of Electrical and Computer Engineering, Korea University

Abstract

We describe a Bluetooth wireless sensor network for security systems, which includes the implementation issues about system architecture, power management, self-configuration of network, and routing. We think that the methods or algorithms described in this paper can be easily applied to other embedded Bluetooth applications for wireless networks.

Ⅰ. Introduction

To overcome the restrictions of wired sensor networks, we propose to use Bluetooth technology for a wireless sensor network for security systems. The proposed network consists of sensor nodes, relay nodes and a control node. All nodes communicate with each other with the Bluetooth module. Sensor and relay nodes detect certain events (e.g. someone enters the security area without permission) and report the events to the control node. Then, the control node reports the information received from the sensor or relay nodes to the local security control system and replies to the corresponding node with an *ACK* message. If sensor nodes are not able to directly reach the control node, the relay nodes placed between them can relay the message from the sensor node to the control node. All nodes transmit and receive packets via a Bluetooth module that is embedded in them. In this paper, we introduce the implementation issues related to the proposed network, which includes system architecture, power management, self-configuration of network and routing. Since the network configuration of Bluetooth is based on the Piconets where each Piconet has one master and up to 7 slaves. So, the tree topology can be considered as a natural and appropriate choice for the networks using Bluetooth.

Ⅱ. The Overview of The Bluetooth System

A hardware architecture overview of Bluetooth is outlined in Figure 1. It consists of an analog part – the Bluetooth radio, and a digital part – the Bluetooth Host Controller. [1]

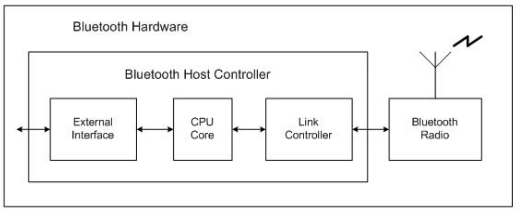


Fig. 1. Bluetooth hardware architecture overview

The Bluetooth radio consists of a transceiver and an antenna, and operates in the ISM band of 2.4GHz. Information is modulated using GFSK (Gaussian Frequency Shift Keying) at a rate of 1M symbols/second and transmitted in one of 79 1MHz channels. Signals are frequency hopped over the 79 channels at a rate of 1600 hops per second.

The Host Controller consists of three parts; the Link Controller (LC), a CPU core, and an external interface part. The link controller consists of hardware and software parts that perform the Bluetooth Baseband processing and physical layer protocols such as ARQ (Automatic Repeat request) and FEC (Forward Error Correction). The CPU core executes the Link Manager (LM) software. It allows the Bluetooth module to handle inquiries and filters Page requests without involving the host device. The external interfaces provide the communication channel between the host and the Host Controller, including RS232 and USB (Universal Serial Bus) interfaces.

The Host Controller Interface (HCI) driver on the host side provides a uniform interface method of accessing the Bluetooth hardware capabilities. The HCI firmware on the Host controller side implements the HCI commands for the Bluetooth hardware by accessing link manager commands, hardware status registers, control registers, and event registers. Figure 2 provides an overview of the Host Controller interface concept.

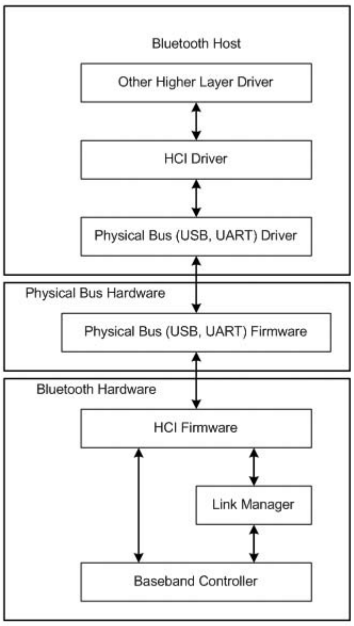


Fig 2. The Host Controller interface software layers

Ⅲ. The Proposed System Architecture

A. System Overview

The purpose of the proposed system is for detecting an invasion in a building that needs security service, and it consists of lots of sensor and relay nodes and a control node. Figure 3 illustrates the communication environment of the system.

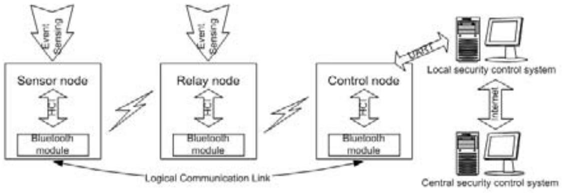


Fig. 3. Communication environment

B. Network Configuration

The flow for it as shown in Fig. 4 is performed by each node in a distributed manner. The procedure is as follows

1) Read memory in order to acquire address and other information which are required for set up when the power is turned on.
2) Perform an inquiry procedure to find adjacent nodes and save respondent node's information which is necessary for connection in an address table (e.g., *BD_ADDR*, clock information, etc...). The inquiry procedure can be continued until finding *N* adjacent nodes or for a fixed period.
3) Page the inquired nodes one by one for connection and exchange a packet including logical address and other information necessary for network configuration.
4) Compare its own logical address with the received logical address to determine the logical relationship between them, which includes parent, child node or none, and then save the result at the address table.
5) Sensor and relay nodes enter a power management mode and the control node enters a standby mode to wait for an event or paging.

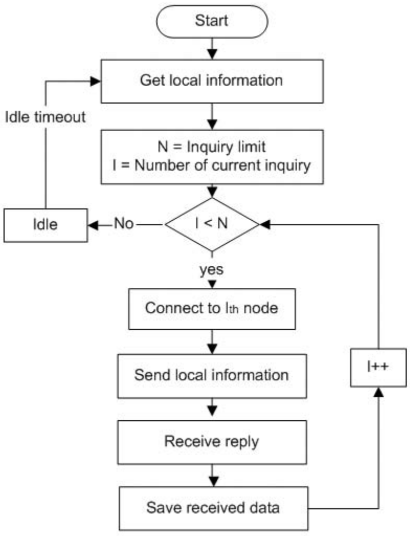


Fig. 4. Network configuration flow

After all nodes go through the above network configuration procedure, they can achieve a network configuration based on the tree topology as shown in Fig. 5. As the depth of tree becomes deeper, the length of logical address becomes longer because an additional value is added to the rear of the logical address for identification. For example, as shown in Fig.5, if the logical address of the control node is 3, the addresses of the nodes just under the control node are something like 3.1, 3.2, and 3.x, and the value of the nodes just under the node with the logical address of 3.1 is something like 3.1.1, 3.1.2, 3.1.x. Due to such a hierarchical addressing scheme, we can easily find multi-hop routes in networks with tree topology as will be explained in Section 3.C.

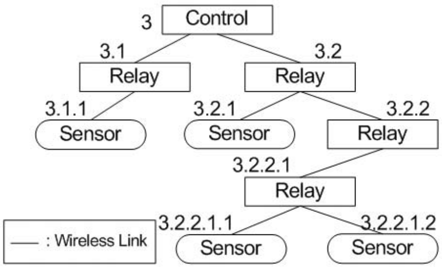
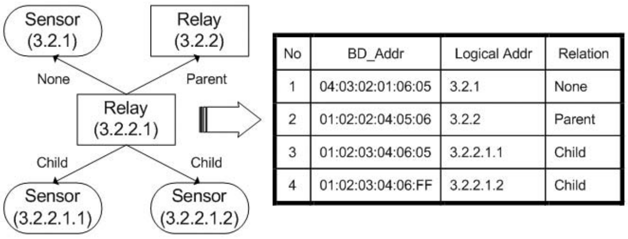


Fig. 5. Network Configuration

At the network configuration stage, each node exchanges a packet with its adjacent nodes, which includes the information necessary for network configuration such as logical address, and compares its logical address with the thing of its adjacent nodes to find the logical relationship between the two nodes. The relationship may become parent or child or none and the result is saved at the address table.



No	BD_Addr	Logical Addr	Relation
1	04:03:02:01:06:05	3.2.1	None
2	01:02:02:04:05:06	3.2.2	Parent
3	01:02:03:04:06:05	3.2.2.1.1	Child
4	01:02:03:04:06:FF	3.2.2.1.2	Child

Fig. 6. Address table

For example, the address table for the relay node with the logical address of 3.2.2.1 is shown in Fig. 6.

C. Routing

The routing of packets is a very simple matter in the proposed system because the data packets sent by sensor or relay nodes are always destined for the control node and a tree topology is adopted for the network configuration. In this case, when a senor or relay node sends a data packet, it just sends the packet to its parent node. Similarly, the parent node relays the packet to its parent node. In this way, the packet finally arrives at the control node. For example, if the sensor node with address 3.2.2.1.1 wants to send a packet to the control node, then it first finds its parent node from the address table and then sends the packet to the relay node with address 3.2.2.1. Then the relay node sends the packet to its parent node, i.e., the relay node with address with 3.2.2. In this way, the packet finally arrives at the control node. After the control node receives the packet, it can obtain the source address of the packet from the packet header for sending an *ACK* packet to the source of the packet, i.e., the sensor node with address 3.2.2.1.1. Since a hierarchical addressing scheme is used for routing, the control node can send the *ACK* packet to the relay with address 3.2 instead of the relay node with address 3.1. After the relay node receives the *ACK* packet, it looks up its address table for routing and compares the destination address of the *ACK* packet with the logical address fields in its address table. Then it finds out that the best match address is 3.2.2. It means that the relay node with address 3.2.2 is the next hop node for the *ACK* packet. In this way, the *ACK* packet finally arrives at the sensor node with address 3.2.2.1.1, i.e., the node that sent the packet corresponding to the *ACK* packet.

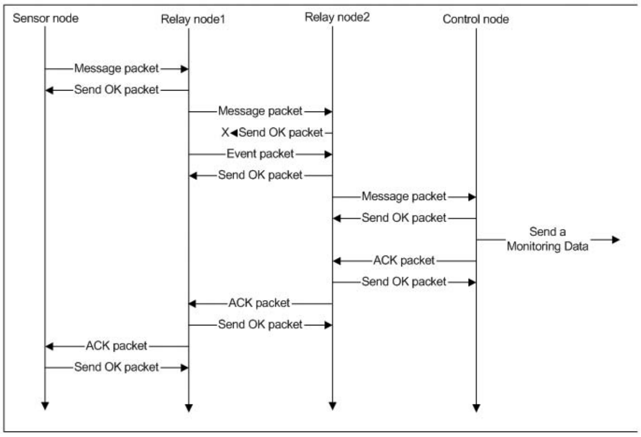


Fig. 7. Data flow chart

An example data flow chart for the proposed system is depicted in Fig. 7.

Figure 8 shows an example of a data transmission sequence for the network configuration shown in Fig.8. In this case, the sender node with address 3.2.2.1.2 sends a packet to the control node with address 3.

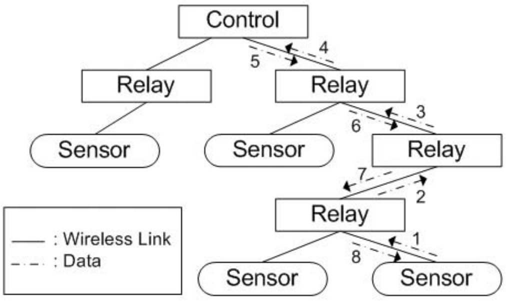


Fig. 8. Example of a data transmission sequence

Also, the packet formats used in the proposed system are summarized in Table I. The *Request* and *Reply* packets are used for exchanging data between two nodes after inquiry.

Table I. Packet formats

Packet	Code	Parameters
Request	0x01	Source address
Reply	0x02	Source address
Message	0x03	Source address, Destination address, Sequence ID, Length, Data [length]
Send OK	0x04	Sequence ID
ACK	0x05	Source address, Destination Address, Sequence ID

D. Operation of nodes

If a sensor or a relay node detects an event, it changes its mode to the active mode from the sniff mode and performs the specific operation corresponding to the event. It then reenters the sniff mode and waits for another event as shown in Fig. 9.

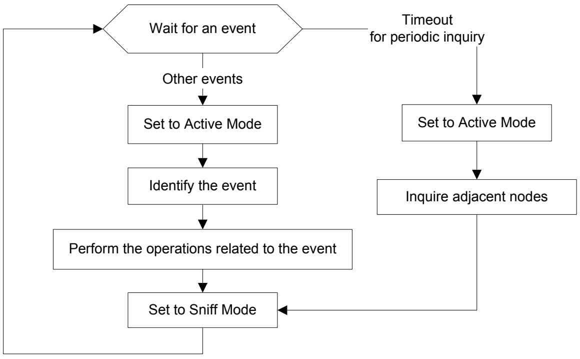


Fig. 9. Operating flow chart for the sensor or relay nodes

The operations related to the events are summarized as follows.

- External input event (e.g., a sensor detects an invader): We emulate this event by means of the on-off switch. The node detecting this event makes a *Message* packet corresponding to the event for reporting it to the local security control system. To send the packet, it searches the logical address of its parent node from the address table and then pages the parent node to send the packet.

- Retransmission timeout events: These events are further classified into the retransmission timeout for the link level ARQ and the retransmission timeout for the upper layer ARQ.
- Inquiry timeout event: This event occurs when the timer for periodic inquiry timeouts.
- Paging event: This event happens when a node is paged.

Ⅳ. Experiments and Discussion

A. Experiment Environment

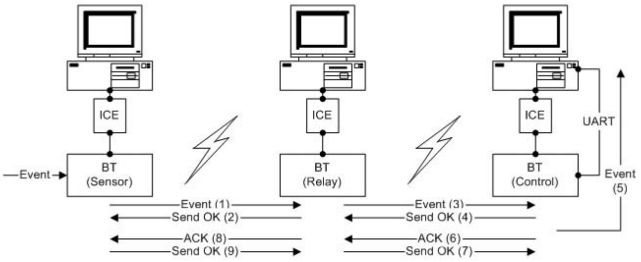
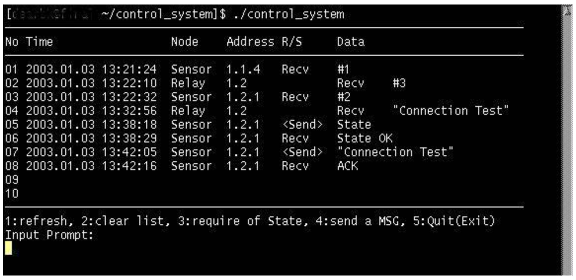


Fig. 10. Experiment environment

Figure 10 shows an example of our experiment environment. PCs are used to monitor nodes and debug the source codes for each node. The PC on the right hand side of the figure also plays the role of the local security control system. An event from a sensor node is relayed to the control node through relay nodes, and then it is reported to the local security system through the UART interface. Also, an operator can send a message from the local security control system to a specific node through the UART interface. When an event is reported by the control node, the local security control system informs the operator of it by means of the message displayed on screen or alarm. Figure 11 shows an example for the monitoring result.



```
[[illegible] ~/control_system]$ ./control_system
No Time                Node    Address R/S    Data
01 2003.01.03 13:21:24 Sensor  1.1.4   Recv   #1
02 2003.01.03 13:22:10 Relay   1.2            Recv    #3
03 2003.01.03 13:22:32 Sensor  1.2.1   Recv   #2
04 2003.01.03 13:32:56 Relay   1.2            Recv    "Connection Test"
05 2003.01.03 13:38:18 Sensor  1.2.1   <Send> State
06 2003.01.03 13:38:29 Sensor  1.2.1   Recv   State OK
07 2003.01.03 13:42:05 Sensor  1.2.1   <Send> "Connection Test"
08 2003.01.03 13:42:16 Sensor  1.2.1   Recv   ACK
09
10
1:refresh, 2:clear list, 3:require of State, 4:send a MSG, 5:Quit(Exit)
Input Prompt:
```

Fig. 11. Monitoring result

B. Experiment Results

We first measure the time taken for inquiring adjacent nodes and exchanging data between the inquiring node and the inquired node. The time consists of the inquiry time, the paging time and the data exchange time. We measure the period from the time when the inquiry procedure is initiated to the time when the data exchange is over. Figure 12 shows the result.

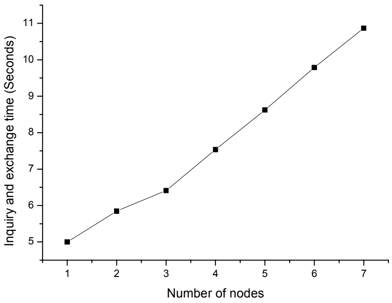
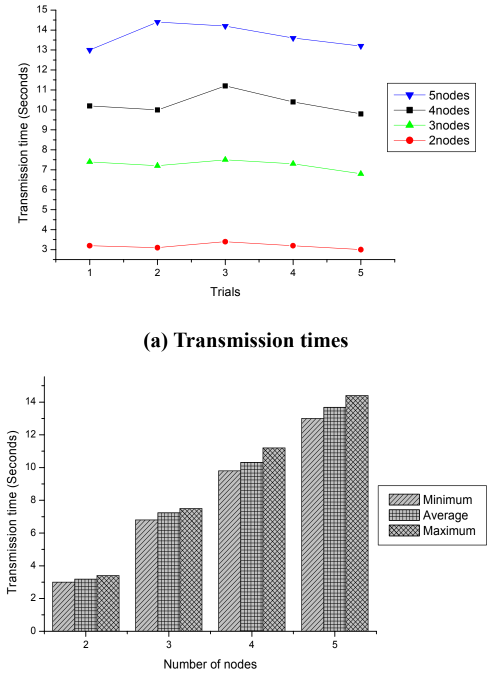


Fig. 12. Inquiry and exchanging data times according to the number of nodes

Second, we measure the time taken to complete the data transmission between a sensor node and the control node. Figure 13 gives the result that shows how the period is changed according to the number of relay nodes placed between two nodes



(a) Transmission times

(b) Variations of transmission times
Fig. 13. Transmission times according to the depth of tree

References

[1] *Specification of the Bluetooth System. Version 1.1*, February 22 2001

[2] Ian F. Akyildiz, Weilian Su, Yogesh Sankarabramaniam, and Erdal Cayirci, "A survey on sensor networks", *IEEE Communications Magazine*, pp. 102-114, August 2002

[3] D. Kaleshi and M. H. Barton, "Ensuring interoperability in a home networking system: A case study", *IEEE Trans. Consumer Electronics*, Vol. 45, No 4, pp. 1134-1143, November 1999.

[4] E.S. Eilley, "In-home digital networks and cordless options", *IEE Colloq. on ATM in professional and consumer electronics*, pp. 8/1-8/6, 1997.

[5] *Bluetooth 2000: To Enable The Star Generation*, Cahners In-Stat Group, MM00-09BW, June 2000

[6] S. Krco, "Bluetooth based wireless sensor networks-implementation issues and solutions," *10th TELECOMUNICATIONS FORUM (TELEFOR2002)* NOV. 2002.