Unified Modeling Language based Analysis of Security Attacks in Wireless Sensor Networks: A Survey

Sunghyuck Hong[1], Sunho Lim[2] and Jaeki Song[3]
[1]Office of International Affairs, Texas Tech University
Lubbock, TX 79409 - USA
[e-mail: sunghyuck.hong@ttu.edu]
[2] Department of Computer Science, Texas Tech University
Lubbock, TX 79409 - USA
[e-mail: sunho.lim@ttu.edu]
[3]Area of ISQS, Texas Tech University
[e-mail:jaeki.song@ttu.edu]
&
Service Sciences Management & Engineering, Graduate School of Business, Sogang University
Seoul, Korea
[e-mail:jaekisong@sogang.ac.kr]
*Corresponding authors: Jaeki Song

Received December 20, 2010; revised February 8, 2011; accepted March 5, 2011; published April 29, 2011

Abstract

Wireless Sensor Networks (WSNs) are rapidly emerging because of their potential applications available in military and civilian environments. Due to unattended and hostile deployment environments, shared wireless links, and inherent resource constraints, providing high level security services is challenging in WSNs. In this paper, we revisit various security attack models and analyze them by using a well-known standard notation, Unified Modeling Language (UML). We provide a set of UML collaboration diagram and sequence diagrams of attack models witnessed in different network layers: physical, data/link, network, and transport. The proposed UML-based analysis not only can facilitate understanding of attack strategies, but can also provide a deep insight into designing/developing countermeasures in WSNs.

Keywords: Security, wireless sensor networks, unified modeling language, standard attack models.

This research was supported in part by the Grants in Office of International Affairs at Texas Tech University, the Dept. of Computer Science at Texas Tech University, US National Science Foundation (CNS-1004210), and a grant from Sogang Business School's World Class University Project (R31-20002) funded by the Korea Research Foundation and the Sogang University Research Grant of 2010.

DOI: 10.3837/tiis.2011.04.010

1. Introduction

With the recent advent in wireless technology and mobile devices, Wireless Sensor Networks (WSNs) are rapidly emerging and becoming popular. Generally a WSN consists of a few hundred (or thousand) tiny, low-power, and multi-functional sensor nodes (later nodes) equipped with sensing, computing, and communicating facilities. Nodes continuously monitor the environment, sense an event of interests or ambient conditions (e.g., light, vibration, temperature, etc), and forward the sensed data to a sink, which could be a gateway to another network or an access point of a querying user. WSNs have been integrated with various applications in military and civilian environments, such as intrusion detection, tactical surveillance, habitat monitoring, structural health monitoring, smart farming, health-care, and home automation [3]. In particular, military applications require a high level of security.

However, providing a security service is challenging in WSNs because of following three major reasons:

- First, since WSNs are often deployed in an unattended or hostile environment, nodes are easily exposed to a lack of centralized coordination or physical protection. For example, one or multiple nodes could be compromised by an adversary, and thus nodes could behave differently against the originally designed communication protocol and degrade the networking performance.
- Second, nodes share a wireless medium for communication and collaborate by forwarding the sensed data through wireless links. Thus, an adversary located within the communication range of the data sender can overhear, duplicate, corrupt, or alter the data. Since the data is often routed to the sink by a single-hop basis broadcast (or point-to-point unicast), most routing protocols in WSNs are simple and susceptible to attacks.
- Third, due to the inherent resource constraints in terms of limited memory size, battery power, and computing capability, conventional security algorithms deployed in wired networks or mobile ad hoc networks cannot directly be applied to the nodes without modification.

In light of these reasons, a great deal of research effort has been devoted to developing various light-weight security algorithms/protocols and countermeasures [1][2][3][4].

In this paper, we revisit a set of selected attack models and analyze them by using a well-known standard notation, Unified Modeling Language (UML), in which UML has not been previously applied to this area. Here, UML has been widely used as the first step in developing an object-oriented design methodology to specify, visualize, and construct the artifacts of software systems. In particular, UML has been proven successful in modeling large and complex systems [13]. Our contribution is summarized in below:

- We suggest a UML-based analysis of various attack models witnessed in different network layers (e.g., physical, data/link, network, and transport) and provide a set of UML collaboration diagram and sequence diagrams to facilitate the understanding of attack strategies.

In this paper, we do not extensively present attack models because not all of them are eligible to be represented. Instead we select a set of representative attack models based on the authors' best knowledge.

The rest of this paper is organized as follows. A set of security attack strategies are investigated and analyzed through the proposed UML approach in physical, data/link, network, and transport layers in Sections 2, 3, 4, and 5, respectively. In Section 5, we conclude the paper with future directions.

2. Physical Layer: Jamming and Tampering

Unlike IEEE 802.15.4 [16] and ZigBee, most WSNs are limited to using spread spectrum capabilities such as frequency hopping and code spreading because of their inherent resource constraints. Also WSNs are often deployed in an unattended or hostile environment. Thus, nodes are easily exposed to threats of communication disruption and lack of physical protection. In this section, we investigate two attack models targeting the functionalities in physical layer: jamming and tampering.

2.1. Jamming Attack

A malicious node detects radio frequencies and disrupts the network. This jamming can interfere with legitimate nodes' communications and achieve similar results to a denial-of-service attack [14]. As shown in **Fig. 1**, a malicious node injects the same radio frequency signal to its neighbor nodes with a relatively high power. Then all neighbor nodes are not able to communicate with others but to stay in idle.

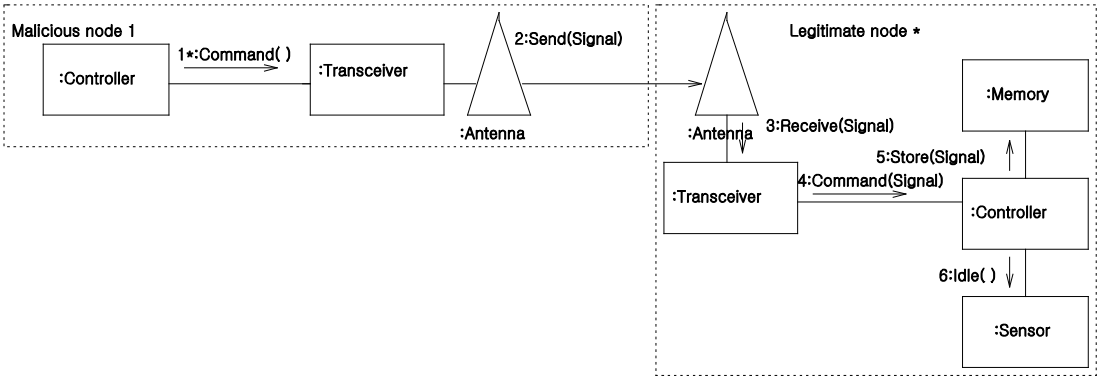


Fig. 1. UML collaboration diagram for jamming attack.

1-2 An adversary places a malicious node 1 (n_1^m) and initiates a jamming attack by injecting the same radio frequency signal to its neighbor nodes.

3 Legitimate nodes (n^*) identify this jamming signal and simply discard it.

4-6 n^* switch to a sleep mode and wakeup periodically to see whether the jamming signal is still on-going.

To deal with this attack, spread spectrum techniques can be used such as frequency-hopping spread spectrum (FHSS). In the FHSS, signals are transmitted by switching a carrier among multiple frequencies based on a pseudo random sequence, which is known to both sender and receiver [4]. In order to attack, the hopping sequence is required or a wide of section of the band should be jammed [6]. However, this countermeasure technique requires non-negligible design complexity and energy consumption.

2.2. Tampering Attack

An adversary can physically access a node, extract sensitive information such as a pre-distributed key or node identification, and place a malicious node. The adversary can execute several attacks through the malicious node. Then this malicious node can easily spoof

the on-going communication and disrupt the network. An example is shown in **Fig. 2**.

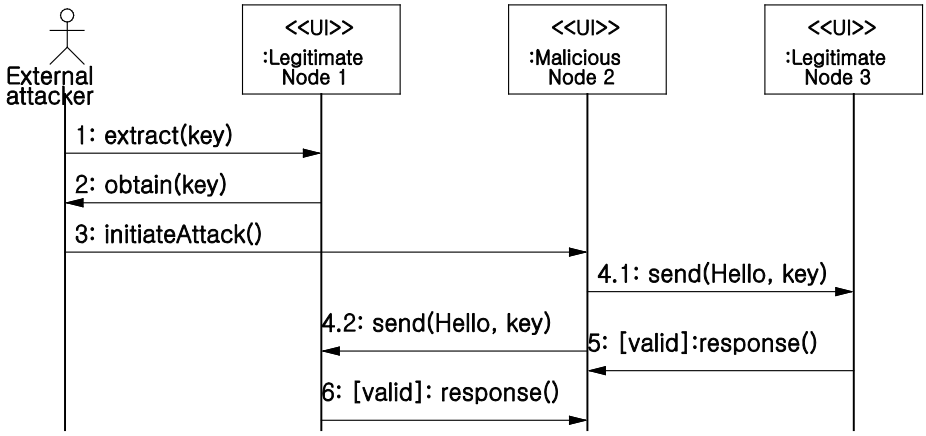


Fig. 2. UML sequence diagram for tampering attack.

1-2. An adversary captures and extracts sensitive information by accessing the legitimate node 1 (n_1) physically.

3. The adversary places a malicious node 2 (n_2^m).

4.1-4.2. n_2^m sends Hello packets to its neighbor nodes for updating routing tables.

5-6. The neighbor nodes respond to n_2^m with an ACK packet. When a node receives the Hello packet, it updates its routing table.

A possible defense approach of this attack is to use tamper proof nodes packaged physically or hide nodes in a more secure location. In reality, however, nodes in most WSNs are not tamper-proofed [7].

3. Data Link Layer: Collision, Resource Exhaustion, and Unfairness

The data link layer is primarily responsible for carrier sensing and packet transceiving. To deal with this, a medium access control (MAC) protocol coordinates nodes to access a wireless medium and resolves conflicts, in which multiple nodes compete to access the shared medium. When a collision occurs, the MAC protocol resolves it by using a contention resolution algorithm such as resending a packet later at a randomly selected time, or simply discarding a packet and leaving the decision of retransmission to the upper layer. In this section, we investigate three major attacks targeting the functionalities in data link layer: collision, resource exhaustion, and unfairness.

3.1. Collision Attack

To reduce collisions due to the hidden terminal problem, although an additional communication overhead occurs, a two-way handshaking of request-to-send (RTS) and clear-to-send (CTS) is often optionally used in IEEE 802.11-based MAC protocols. To attack this collision avoidance effort, a malicious node can interfere with an on-going

communication by transmitting a packet simultaneously, resulting in collision. In particular, the malicious node targets an ACK packet to maximize the communication delay and energy consumption (i.e., retransmission after an exponential back-off). An example of this attack is shown in **Fig. 3**.

1 An adversary places a malicious node 4 (n_4^m) and mounts a collision attack on n_4^m .

2 A node 2 (n_2) senses an event or has the sensed data to send.

3 n_2 sends a RTS to node 3 (n_3).

4 n_1 and n_4 who overhear either a RTS or CTS (e.g., n_2) should defer its transmission when the medium is busy, and set/update its Network Allocation Vector (NAV[1]) based on the duration specified in a RTS or CTS.

5 When n_3 receives the RTS, it replies a CTS back to the sender after a short Inter Frame Space (IFS) interval.

6 When n_4^m receives a busy indicate signal from n_3, it ignores the NAV and starts to send a RTS for collision with n_2.

7 After exchanging the RTS and CTS, n_2 sends the sensed data, and the collision attack is successful on n_3.

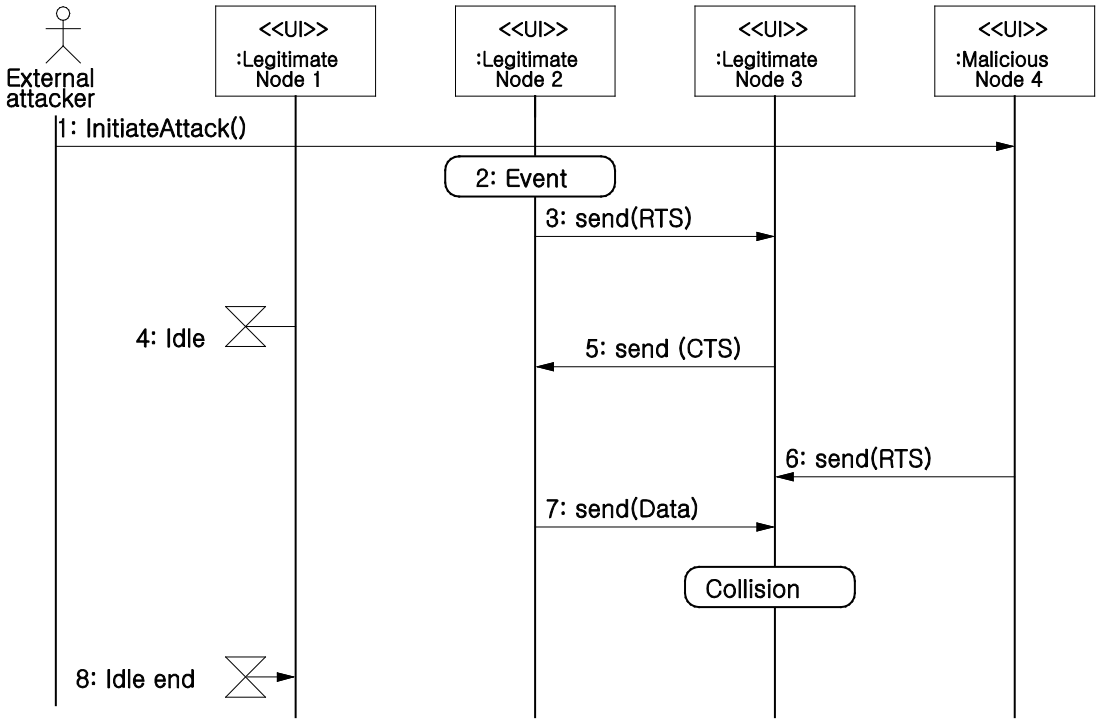


Fig. 3. UML sequence diagram for collision attack.

[1] The NAV contains an expected time and indicates the remaining time of following transmission sessions, and it always decreases regardless of the medium state while the back-off decreases only when the medium is idle.

Various error correcting code techniques can be used to defend this attack, but additional computation and communication overheads are expected. In fact, a reactive jamming attack [5][6] is similar to the collision attack, in which a malicious node keeps quiet when the channel is idle but transmits a jam signal when it senses an on-going communication. Because of this similarity to a normal packet collision, the malicious node is hard to be detected.

3.2. Resource Exhaustion Attack

Since wireless communication could be responsible for more than half of energy consumption [7], repeated aforementioned collision attacks can cause a series of retransmitting control and data packets and finally result in the battery depletion. A great deal of effort has been devoted to developing energy efficient MAC protocols. They primarily focus on how to judiciously place the nodes' radio in a low power or a sleep mode[2] as long as possible without degrading the communication performance, but they are quite vulnerable to resource exhaustion attacks. For example, denial-of-sleep (DoS) attack [8][9] keeps the radio in an active mode. A malicious node may repeatedly send RTSs, eliciting a CTS response from the targeted neighbor nodes. As a consequence, a set of involved nodes will eventually exhaust the energy. To prevent this unnecessary energy spending, MAC admission control can limit a number of same requests or ignore the excessive requests. However, because of the inherent promiscuous overhearing behavior, nodes may waste non-negligible energy for such a bogus RTS. An example of this attack is shown in **Fig. 4**.

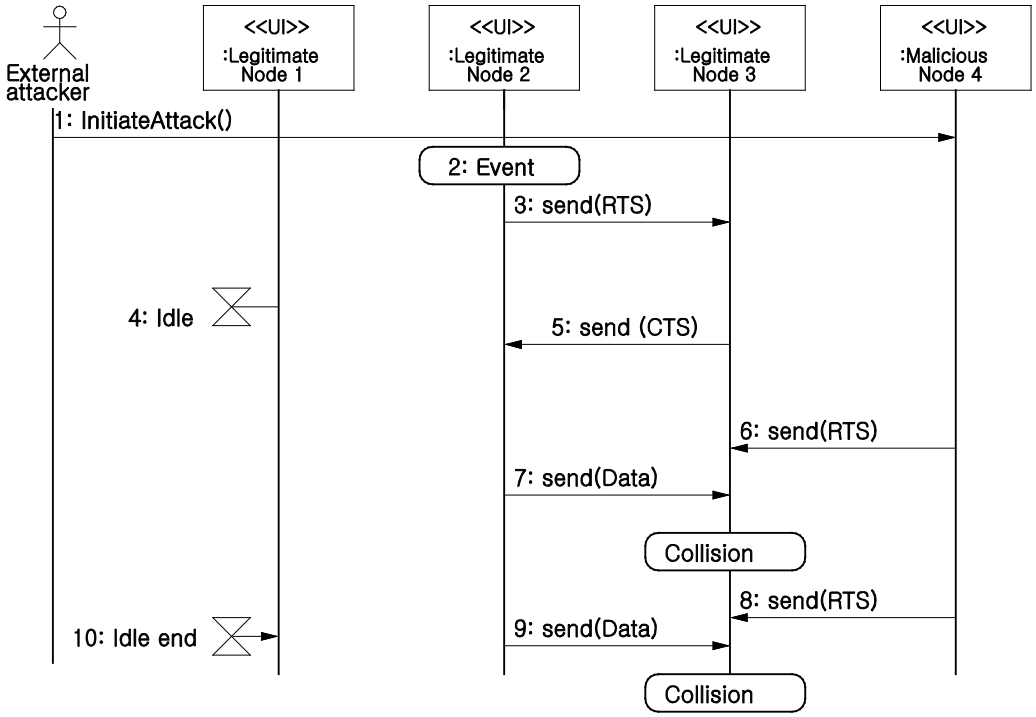


Fig. 4. UML sequence diagram for exhaustion attack.

[2] As pointed in [14], the Crossbow Mica2 consumes 36.81 *mW* in receive mode while 0.048 *mW* in sleep mode. It works over 4,000 days in sleep mode but only 10 days in receive mode under the two standard 3,000 *mAh* AA batteries.

1	An adversary places a malicious node 4 (n_4^m) and mounts a resource exhaustion attack on n_4^m .
2-3	A node 2 (n_2) senses an event or has the sensed data to send, and n_2 sends a RTS to node 3 (n_3).
4	n_1 and n_4 who overhear either a RTS or CTS (e.g. n_2) should defer its transmission when the medium is busy, and set/update its NAV based on the duration specified in the RTS or CTS.
5	When n_3 receives the RTS, it replies CTS back to the sender after an IFS interval.
6	n_4^m ignore the NAV and begin to send a RTS for collision with n_2.
7	After exchanging the RTS and CTS, n_2 sends the sensed data, and collision attack is successful on n_3.
8-9	n_4^m keeps sending a RTS repeatedly until n_3 uses up its battery and becomes exhausted.

3.3. Unfairness Attack

Due to the lack of centralized coordination, most MAC protocols heavily rely on a distributed contention resolution mechanism to ensure the fair share of the wireless medium. The implicit assumption of this mechanism is that all the nodes participating in a network follow (or cooperate) with the communication protocol. However, a malicious node may intentionally misbehave by ignoring the protocol to obtain the medium [10]. For example, the malicious node selects the back-off value from a smaller range (e.g., [0, CW/4]) or uses a different back-off scheme instead of the exponential back-off. Also, the malicious node specifies higher transmission duration than that of actual RTS and delays its neighbor nodes to compete with the medium. These simple yet effective unfairness attacks significantly degrade the communication performance of well behaved nodes. An example is shown in **Fig. 5**.

1	An adversary places a malicious node 2 (n_2^m) and mounts a fairness attack on node 2 (n_2).
2	n_2 senses an event or has the sensed data to send.
3	n_2 sends a RTS to node 3 (n_3). In the mean time, n_2^m increases the NAV on n_2 , causing that the neighbor nodes do not have a chance to communicate with other nodes.
4-5	When node 1(n_1) overhears the RTS, it becomes idle during the NAV time interval.
6	When n_3 receives the RTS, it replies CTS back to the sender.
7	After exchanging the RTS and CTS, n_2 sends the sensed data to n_3. As a result, n_1 will be isolated successfully.

4. Network Layer: Spoofed Routing Information, Selective, Sinkholes, Wormholes, Hello Flood, and Acknowledgement attacks

Major attacks witnessed in the network layer are dropping packets, sending packets in a wrong direction through the unreachable destination, establishing a route to a destination by including a malicious node as a part of the routing path, spoofing, and altering the routing

information.

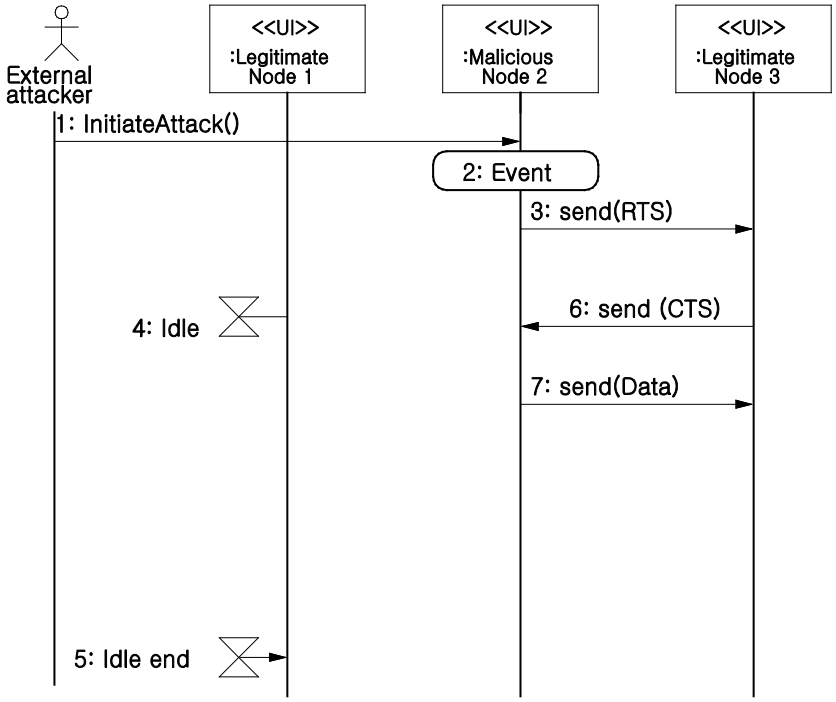


Fig. 5. UML sequence diagram for unfairness attack.

4.1. Routing Information Attack

Due to overhearing and multi-hop routing in WSNs, a malicious node located along the way where packets are routed can intercept and drop, spoof, alter, or replay the routing information piggybacked in the packets. Thus, the malicious node can create routing loops and generate fake error packets, resulting in a partitioned network and degraded communication performance [2]. Here, the implicit assumption of these attacks is that each packet contains a complete routing path and each node faithfully forwards the received packets. An example of attack is shown in **Fig. 6**.

1 An adversary places a malicious node 3 (n_3^m) and mounts a routing information attack on n_3^m .

2-3 n_1 detects an event and forwards the sensed data to n_2 located in the routing path, $n_1 \rightarrow n_2 \rightarrow n_3^m$ (malicious) $\rightarrow n_4$.

4 n_2 forward the received data from n_1 to n_3^m according to the routing path.

5 n_3^m alters the routing path as $n_3^m \rightarrow n_2 \rightarrow n_1$, resulting in a routing loop.

6 n_2 forward the received data from n_3^m to n_1 according to the altered routing path.

7 n_1 forward the received data from n_2 to n_1 according to the altered routing path. Repeat steps 3 thru 6.

4.2. Sinkhole Attack (Black Hole Attack)

A laptop class adversary equipped with a powerful transmitter can advertise zero-cost routes to network nodes to attract more traffic in their directions [15]. The adversary may also include itself into the part of routes, attract the packets, and simply drop all the received packets. An example of attack is shown in **Fig.7**.

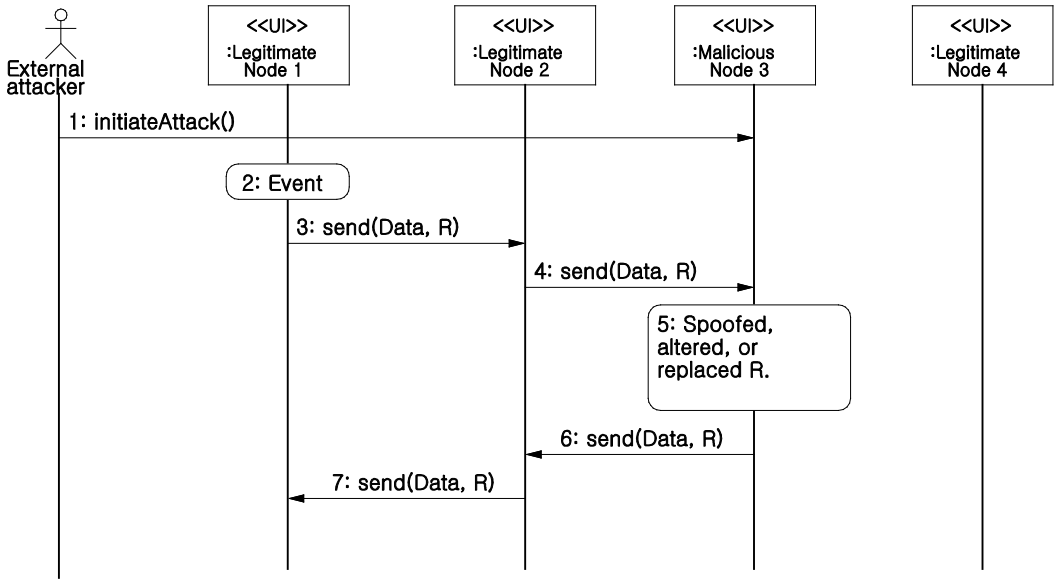


Fig. 6. UML sequence diagram for spoofed, altered, or replaced routing information attack.

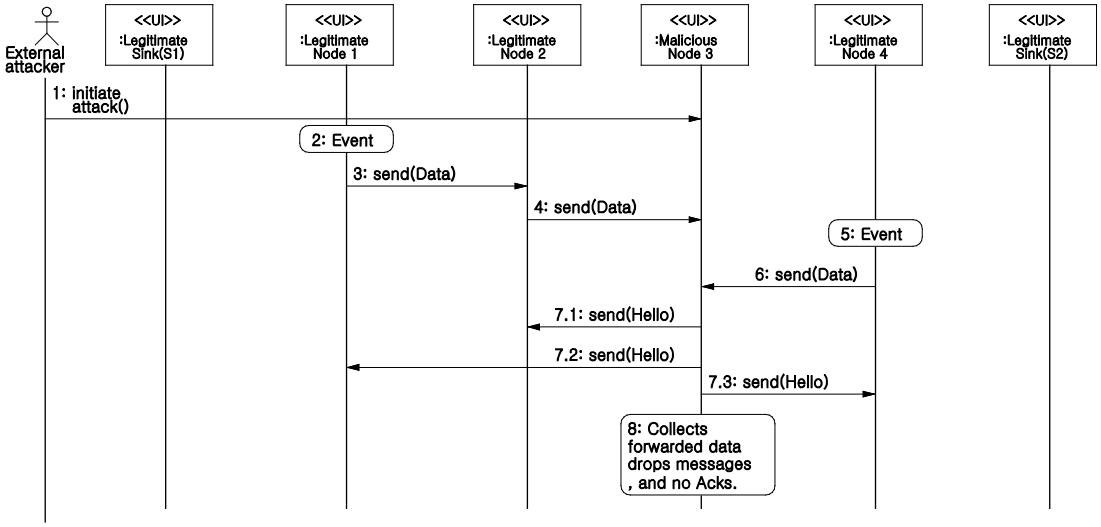


Fig. 7. UML sequence diagram for sinkhole attack.

1 An adversary places a malicious node 3 (n_3^m), and mounts a sinkhole attack on n_3^m .

2-3 n_1 detects an event and then forwards the sensed data to n_2, which is on the routing path ($n_1 \rightarrow n_2 \rightarrow n_3^m \rightarrow n_4 \rightarrow S_2$ (Sink)).

4 n_2 forward the received data from n_1 to n_3^m according to the routing path.

5-6 n_4 detects an event and forwards the sensed data to n_3^m, which is on the routing path ($n_4 \rightarrow n_3^m \rightarrow n_2 \rightarrow n_1 \rightarrow S_1$ (Sink)).

7.1-7.3 n_3^m receives the data from n_4 and then it does not respond to n_4. n_3^m repeatedly sends Hello packets to its neighbor nodes to update their routing information with the altered final destination as n_3^m.

8 n_3^m can refuse to forward packets and simply drop them, ensuring that they are not propagated any further.

4.3. Sybil Attack

A malicious node illegitimately takes on multiple identities. There are two ways to perform this attack. First, a sybil node communicates directly with legitimate nodes. Second, packets sent to a sybil node are routed through one of these malicious nodes that pretend to pass the packets to the sybil node [12]. To deal with this attack, nodes should check a list of "*known-good*" identities to validate a legitimate node. Also, the physical position is verified. The sybil node can be detected because it will appear at exactly the same position as the malicious node that generates identities. An example of this attack is shown in **Fig. 8**.

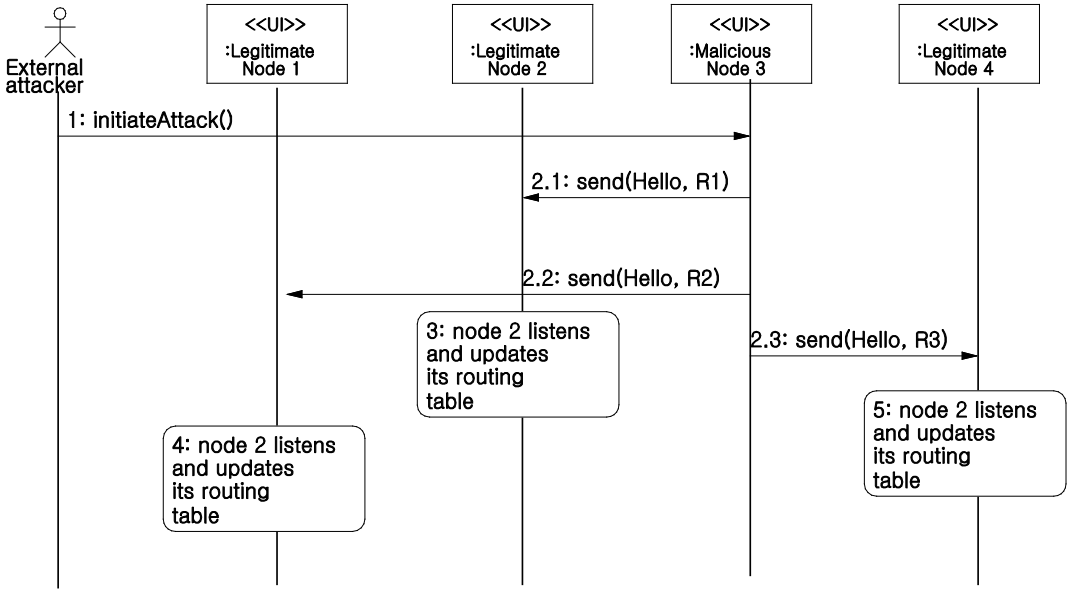


Fig. 8. UML sequence diagram for sybil attack.

1 An adversary places a malicious node 3 (n_3^m), and mounts a sybil attack on n_3^m.

2.1.-2.3 n_3^m assigns a set of channels (e.g., R1, R2, and R3) to its neighbor nodes to broadcast Hello packets. n_3^m can generate identities with a random value or steal from one of legitimate nodes.

3-5 If the channels are legitimate, then n_1, n_2, and n_4 listen and update their routing tables.

In this case, some identities can be dead or unreachable. However, the neighbor nodes of n_3^m are notified that all adjacent nodes are alive and reachable.

4.4. Wormhole Attack

An adversary records packets at one point in the network and tunnels them selectively to the other point, where the packets are retransmitted [17]. Here, the link could be a long-range wireless transmission or an Ethernet cable. This attack can easily disrupt the communication and interfere with any routing protocols that rely on the geographic proximity. An example of this attack is shown in **Fig. 9**.

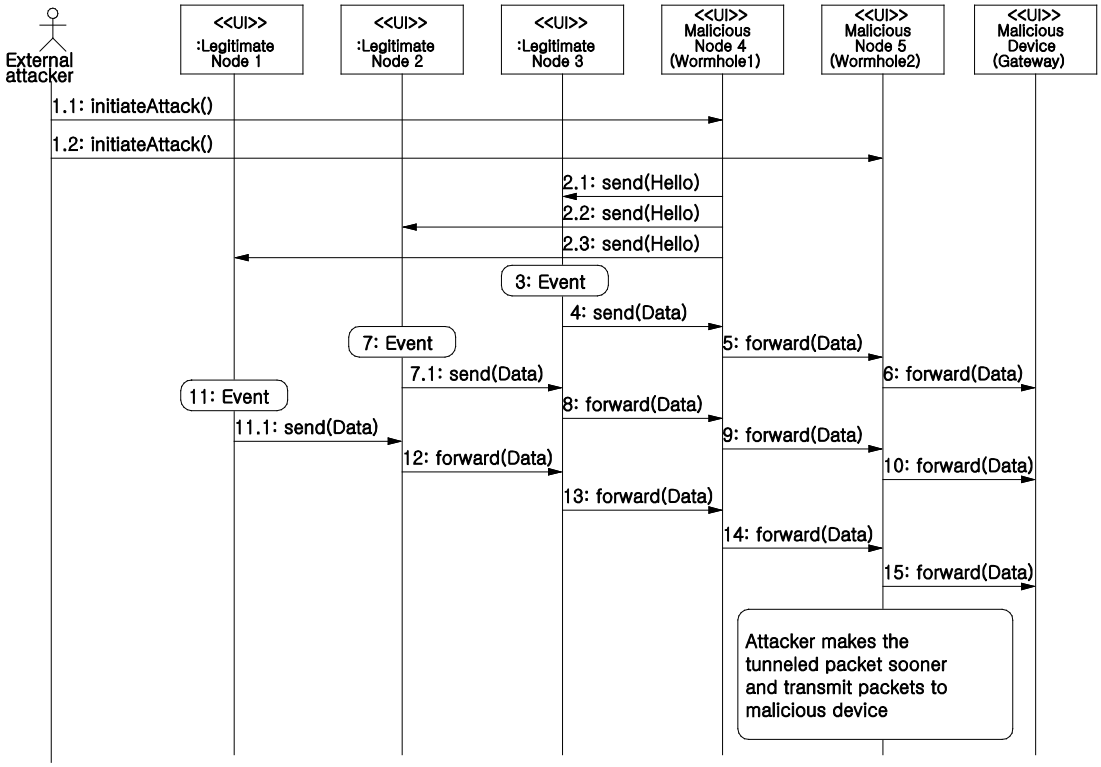


Fig. 9. UML sequence diagram for wormholes attack.

1.1-1.2 An adversary places a malicious node 4 (n_4^m) and 5 (n_5^m), and mounts wormholes attacks on n_4^m and n_5^m.

2.1-2.3 n_4^m sends Hello packets to its neighbor node packet in order to update their routing tables, in which the destination is changed to n_4^m.

3-4, 7-7.1, 11-11.1,13 The neighbor nodes of n_4^m send the sensed data whenever an event occurs.

5-6, 9-10, 14-15 n_4^m receives the data from its neighbor nodes and forwards n_5^m, in which a tunnel between two wormhole nodes (n_4^m and n_5^m) is established to send the data to a malicious device that plays a role as a gateway.

4.5. Hello Flood Attack

The implicit assumption of this attack is that each node uses a static routing information table and updates its table periodically by sending or receiving Hello and ACK packets. If a laptop-class adversary with a powerful transmitter broadcasts a Hello packet to its neighbor nodes, then a node receiving the packet may assume that it is located within the communication range of the sender. Therefore, the packet receiving nodes will update their routing tables accordingly. An example of this attack is shown in **Fig. 10**.

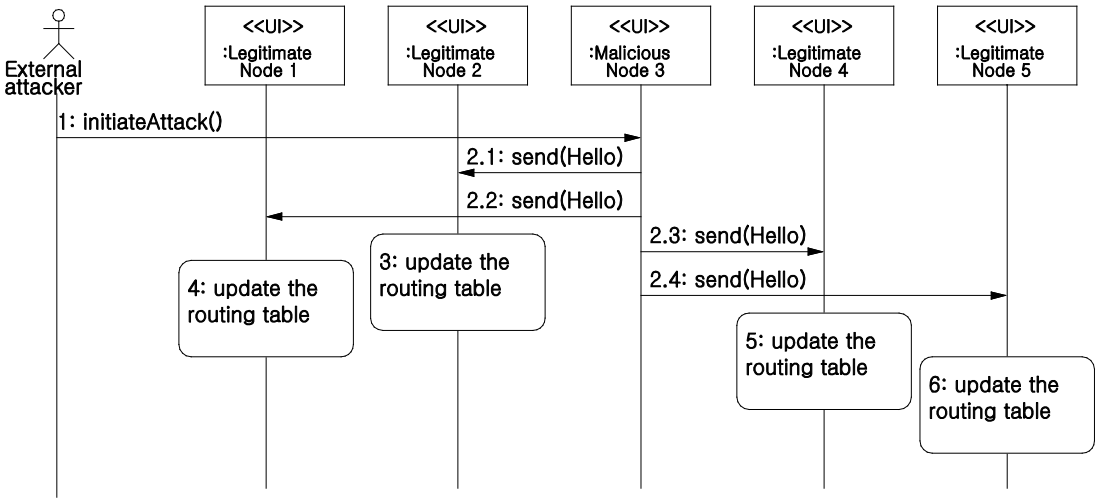


Fig. 10. UML sequence diagram for hello flood attack.

1. An adversary places a malicious node 3 (n_3^m), and mounts a hello flood attack on n_3^m.

2.1-2.4 n_3^m broadcasts Hello packets to all of its neighbor nodes located up to two hops away.

3-6 After receiving the packets, nodes update their routing tables.

4.6. Acknowledgment Spoofing Attack

Each node uses a static routing information table and updates its routing table periodically by sending or receiving Hello and ACK packets. Meanwhile, an adversary can spoof the acknowledgments of overheard packets destined for neighbor nodes in order to provide false information to them. An example of such false information is that a node is still alive when in fact it is dead [4]. An example of this attack is shown in **Fig. 11**.

1. An adversary places a malicious node 3 (n_3^m), and mounts an acknowledgement spoofing attack on n_3^m .

2,5,8,12 n_1 broadcasts Hello packets to its neighbor nodes to have them update their routing tables.

3,6 The received nodes send an ACK packet back to the sender.

9 If the link between n_4 and n_1 is weak, then the ACK packet cannot reach to n_4 .

13 If the link between n_5 and n_1 is weak, then the ACK packet cannot reach to n_5 .

10,14 If n_3^m notices weak ACK signals from n_4 and n_5, then it sends the ACK packet to n_1 so that the n_1 realizes that both n_4 and n_5 are reachable.

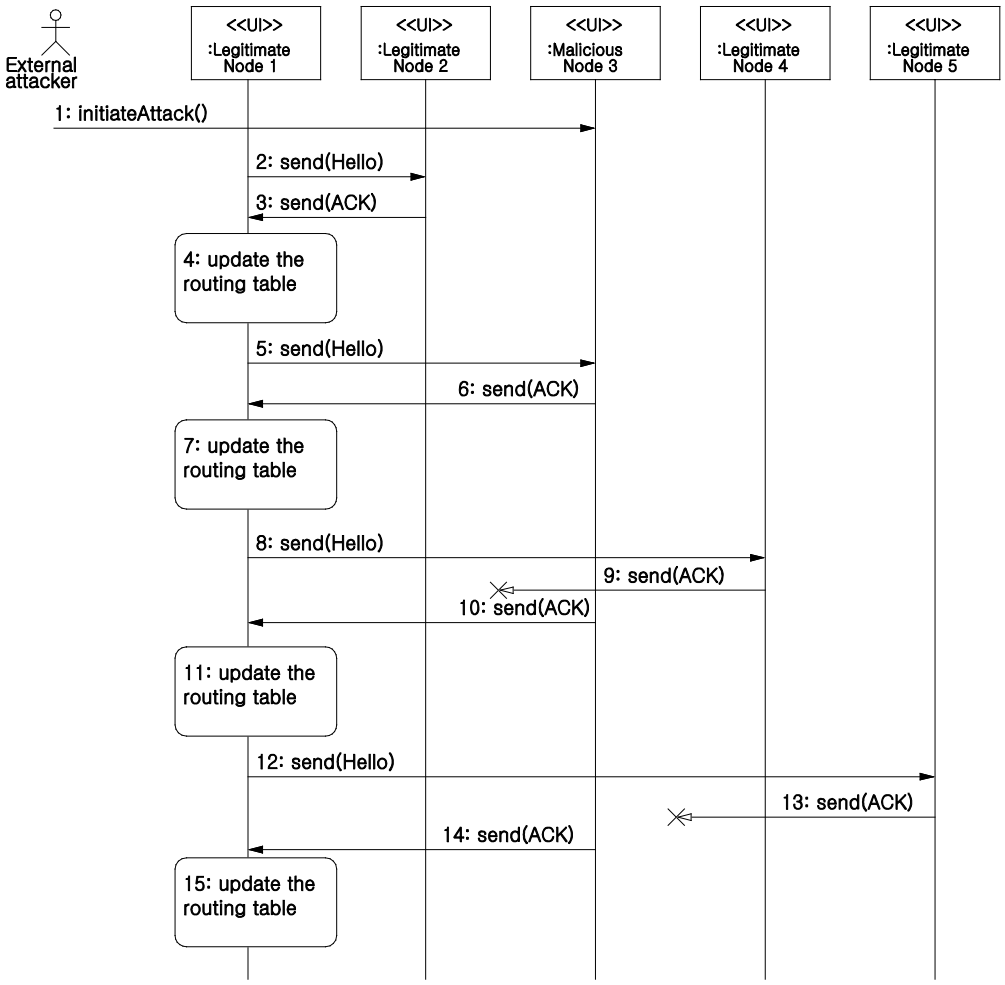


Fig. 11. UML sequence diagram for acknowledgment spoofing attack.

5. Transport Layer: Flooding and De-synchronization

The transport layer is primarily responsible for managing end-to-end logical connections and optimally provides services including reliable data communication, flow control, and congestion control. Most attacks that occur in this layer target a well-known TCP, but a TCP is not necessarily used in WSNs. There are two major attacks: flooding and de-synchronization.

5.1. Flooding Attack

Flooding attacks primarily exploit weaknesses in communication protocols, where connection information must be maintained at both ends of a connection. These protocols become vulnerable when a malicious node repeatedly transmits connection request packets and attempts to exhaust resources. For example, a TCP is a connection oriented communication protocol that requires the three-way handshake process to establish a connection. In the TCP SYN attack [11], a malicious node can send multiple connection establishment requests to the server with spoofed source addresses. This causes the server to keep allocating resources (i.e., memory) for bogus connections. When the maximum half-open connection limit is reached, any successive legitimate connection request will be refused. An example of this attack is shown in **Fig. 12**.

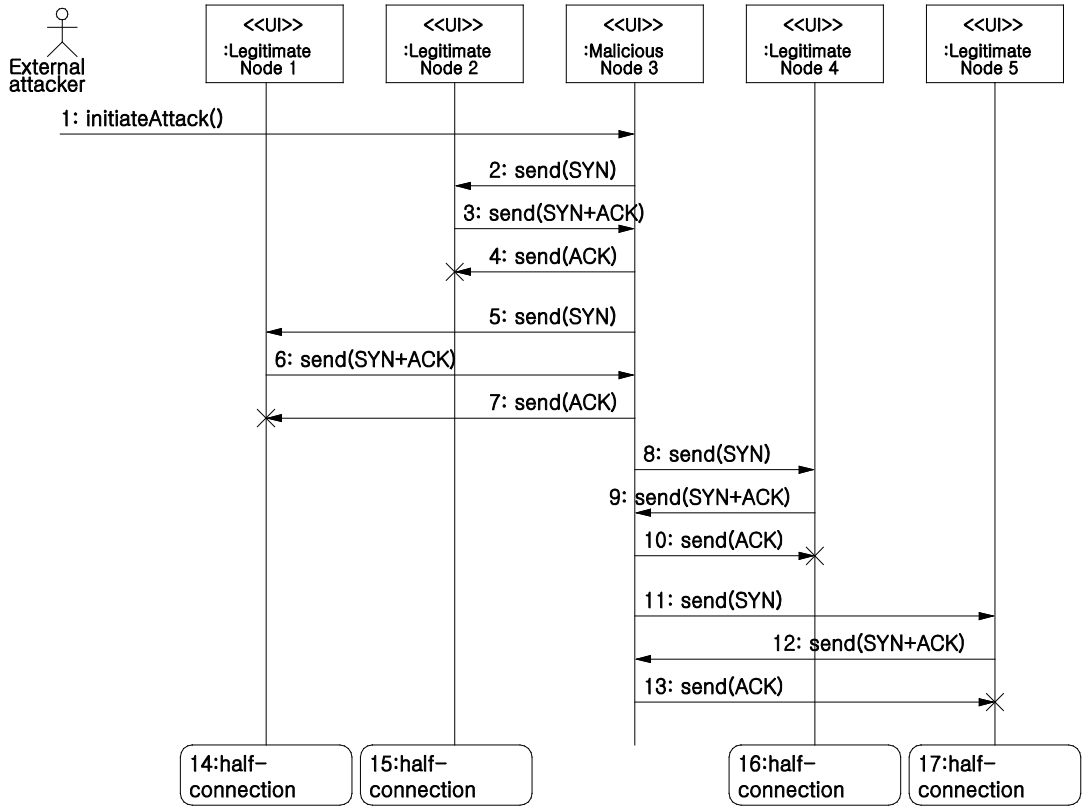


Fig. 12. UML sequence diagram of flooding attack.

1 An adversary places a malicious node 3 (n_3^m), and mounts a flooding attack on n_3^m .

2,5,8,11	n_3^m broadcasts a *SYN* packet to its neighbor nodes to establish an end-to-end connection. The malicious nodes would repeat the Steps 2 and 3 several times to overwhelm the targeted nodes.
3,6,9,12	When the neighbor nodes (e.g., n_1, n_2, n_4, and n_5) of n_3^m receives the *SYN* packet, they replies a *SYN+ACK* packet back to the sender n_3^m .
4,7,10,13	Even though n_3^m receives the *SYN+ACK* packets, it will not respond to its neighbor nodes with an *ACK* packet.
14-17	After a certain period of time, the neighbor nodes of n_3^m resend the *SYN+ACK* packet to n_3^m , resulting in all legitimate nodes in half-open connection. Thus, they cannot communicate with other nodes until the half-connections are done.

5.2. De-synchronization Attack

A malicious node may interrupt an ongoing connection by repeatedly transmitting forged packets containing sequence numbers or control flags to the victims, resulting in a de-synchronization between the two ends of the nodes. Then the node requests the retransmission of missed frames, resulting in resource exhaustion (i.e., energy). If the malicious node can maintain the correct timing, it can even prevent any further packet exchange between two ends of the nodes. This can be avoided by authenticating all the exchanged packets so that the malicious node cannot spoof the packets. However, packet authentication requires a non-negligible computational power.

6. Conclusion and Future Work

In this paper, we revisited various attack models in WSNs, analyzed them through an UML-based standard notation, and presented an UML collaboration diagram and sequence diagrams. We selected a set of representative attack models witnessed in different network layers for analysis: physical, data/link, network, and transport. The proposed UML-based analysis showed a set of interactions among legitimate and malicious nodes for better understanding of attack strategies. We believe that our approach can provide a potential space in designing/developing countermeasures for more secure WSNs.

There are many challenges that need further investigation to exploit the full potential improvement in our study. We first plan to include a robust UML diagram to analyze more sophisticated attack models, such as an activity diagram, state machine diagram, class diagram, composite structure diagram, and interaction diagram. Second, we plan to design/develop a countermeasure based on the proposed UML-based analysis.

References

[1] S. Hadim and N. Mohamed, “Middleware: Middleware Challenges and Approaches for Wireless Sensor Networks,” *IEEE Computer Society*, vol. 7, no. 3, pp. 1-23, Mar. 2006. Article (CrossRef Link)

[2] C. Karlof and D. Wagner, “Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures,” *Ad Hoc Networks*, vol. 1, no. 2-3, pp. 293-315, Sep. 2003. Article (CrossRef Link)

[3] L. Akyyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A Survey on Sensor Networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102-114, Aug. 2002. Article (CrossRef Link)

[4] Y. Wang, G. Attebury and B. Ramamurthy, "A Survey of Security Issues in Wireless Sensor Networks," *IEEE Communications Surveys & Tutorials*, vol. 8, no. 2, pp. 2-23, 2006. Article (CrossRef Link)

[5] W. Xu, W. Trappe, Y. Zhang and T. Wood, "The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks," in *Proc. MobiHoc*, pp. 46–57, 2005. Article (CrossRef Link)

[6] D. R. Raymond and S. F. Midkiff, "Denial-of-Service in Wireless Sensor Networks: Attacks and Defenses," *Pervasive Computing*, pp. 74-81, 2008. Article (CrossRef Link)

[7] R. Kravets and P. Krishnan, "Power Management Techniques for Mobile Communication," in *Proc. IEEE MOBICOM*, pp. 157-168, 1998. Article (CrossRef Link)

[8] T. Martin, M. Hsiao, D. Ha and J. Krishnaswami, "Denial-of-Service Attacks on Battery-powered Mobile Computers," in *Proc. PerCom*, pp. 309-318, 2004. Article (CrossRef Link)

[9] D. Raymond, R. Marchany, M. Brownfield and S. Midkiff, "Effects of Denial of Sleep Attacks on Wireless Sensor Newtork MAC Protocols," in *Proc. Workshop on Information Assurance*, pp. 297-304, 2006. Article (CrossRef Link)

[10] Anthony D. Wood and John A. Stankovic, "Denial of Service in Sensor Networks," *IEEE Computer Society Press*, vol. 35, no. 10, pp. 48-56, 2002. Article (CrossRef Link)

[11] C. L. Schuba, I. V. Krsul, M. g. Kuhn, E. H. Spafford, A. Sundaram and D. Zamboni, "Analysis of a Denial of Service Attack on TCP," in *Proc. IEEE Symposium on Security and Privacy*, pp. 208-223, 1997. Article (CrossRef Link)

[12] T. Aura, P. Nikander and J. Leiwo, "DOS-Resistant Authentication with Client Puzzles," in *Proc. Security Protocols Workshop*, pp. 170-177, 2000. Article (CrossRef Link)

[13] L. Zhao, X. Zhao, Q. Long and Z. Yan, "A type system for the relational calculus of object systems," *11th IEEE International Conference on ICECCS 2006*, pp.10. Sep. 2006. Article (CrossRef Link)

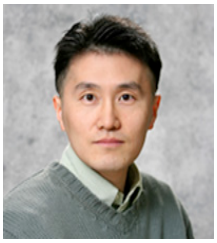
[14] S.K. Jain and K. Garg, "A Hybrid Model of Defense Techniques against Base Station Jamming Attack in Wireless Sensor Networks," *Computational Intelligence, Communication Systems and Networks, 2009. CICSYN '09. First International Conference on*, pp.102-107, Jul. 23-25, 2009. Article (CrossRef Link)

[15] S. Cheung and K.N. Levitt, "Protecting Routing Infrastructures from Denial of Services Using Cooperative Intrusion Detection", in *Proc. Workshop on New Security Paradigms, ACM Press*, New York, pp. 94 – 106, 1997. Article (CrossRef Link)

[16] Gutierrez, J.A.; Naeve, M., Callaway, E., Bourgeois, M., Mitter, V. and Heile, B., "IEEE 802.15.4: a developing standard for low-power low-cost wireless personal area networks," *Network, IEEE*, vol. 15, no. 5, pp.12-19, Sep/Oct. 2001. Article (CrossRef Link)

[17] Yih-Chun Hu; Perrig, A. and Johnson, D.B., "Wormhole attacks in wireless networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 2, pp. 370-380, Feb. 2006. Article (CrossRef Link)

Sunghyuck Hong received the Ph.D. degree from Texas Tech University in August, 2007 major in Computer Science.
Currently, he works at International affairs in Texas Tech University as a senior programmer/analyst, and his current jobs are development of ASP.NET web applications and maintenance of PC/Server. He is a member of editorial board in the Journal of Korean Society for Internet Information (KSII) Transactions on Internet and Information Systems. His current research interests include Secure Wireless Sensor Networks, Key Management, and Networks Security.



Sunho Lim received his B.S. degree (summa cum laude) from Dept. of Computer Science and M.S. degree in the Dept. of Computer Engineering from Hankuk Aviation University (a.k.a. Korea Aerospace University), Korea, in 1996 and 1998, respectively. He received the Ph.D. degree in the Dept. of Computer Science and Engineering from The Pennsylvania State University, University Park, in 2005. He is currently an assistant professor in the Department of Computer Science, Texas Tech University (TTU). Before joining TTU, he was an assistant professor in the Dept. of Electrical Engineering and Computer Science, South Dakota State University, from 2005 to 2009. His research interests are in the areas of Wireless Mobile Networks, Mobile Data Management, and Embedded Networked Systems. Also he is extending the research area into Network Security. He was a recipient of the Governor's 2010 Individual Research Seed Grant and NSF Grant at 2007 and 2008, respectively. He is currently leading the TTU Wireless Mobile Networking Laboratory. He is a member of the IEEE.

Jaeki Song is the Jerry S. Rawls Professor of MIS at the Rawls College of Business at Texas Tech University and a visiting associate professor at the Graduate School of Business at Sogang University in South Korea. His primary area of research is the adoption of Web-based technologies and Service Engineering Management. Specific research issues include service innovation, service productivity, service design, technology continuance, trust, business intelligence, and social aspects of adoption. His research findings have appeared more than 30 academic journals including *Management Science, Journal of Management Information Systems, IEEE Transactions on Professional Communication, Decision Support Systems, Information & Management, and other decent journals.*