Hindawi Publishing Corporation
Journal of Sensors
Volume 2016, Article ID 8797568, 11 pages
http://dx.doi.org/10.1155/2016/8797568

Research Article

From Wireless Sensor Networks to Wireless Body Area Networks: Formal Modeling and Verification on Security Using PAT

Tieming Chen,[1] Zhenbo Yu,[1] Shijian Li,[2] and Bo Chen[1]

[1]*College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China*
[2]*College of Computer Science and Technology, Zhejiang University, Hangzhou 310012, China*

Correspondence should be addressed to Tieming Chen; tmchen@zjut.edu.cn

Received 29 March 2015; Revised 14 July 2015; Accepted 27 July 2015

Academic Editor: Jian-Nong Cao

Model checking has successfully been applied on verification of security protocols, but the modeling process is always tedious and proficient knowledge of formal method is also needed although the final verification could be automatic depending on specific tools. At the same time, due to the appearance of novel kind of networks, such as wireless sensor networks (WSN) and wireless body area networks (WBAN), formal modeling and verification for these domain-specific systems are quite challenging. In this paper, a specific and novel formal modeling and verification method is proposed and implemented using an expandable tool called PAT to do WSN-specific security verification. At first, an abstract modeling data structure for CSP#, which is built in PAT, is developed to support the node mobility related specification for modeling location-based node activity. Then, the traditional Dolev-Yao model is redefined to facilitate modeling of location-specific attack behaviors on security mechanism. A throughout formal verification application on a location-based security protocol in WSN is described in detail to show the usability and effectiveness of the proposed methodology. Furthermore, also a novel location-based authentication security protocol in WBAN can be successfully modeled and verified directly using our method, which is, to the best of our knowledge, the first effort on employing model checking for automatic analysis of authentication protocol for WBAN.

1. Introduction

Formal modeling and analysis on security protocols have gained worldwide attention in recent years, particularly with the proliferation of formal method with model checking [1] since Lowe successfully found a bug on NSPK protocol through modeling and verification using CSP model checking tool FDR [2]. Recently, model checking also achieved a great success on security protocol analysis and verification for wireless sensor networks (WSN) using the Dolev-Yao attack assumption model and some specific tools [3]. For example, Ballarini modeled S-MAC protocol using probabilistic model checking tool PRISM [4], and Saxena found a flaw on TinySec protocol using AVISPA model checking tool LEAP [5].

As a new kind of wireless network security protocol, some novel wireless body area networks (WBAN) authentication protocols also appeared recently, especially in pervasive healthcare applications. The sensor on human body is more quite limited than that of WSNs; therefore, authentication schemes based on symmetric key cryptography for WSNs, such as Tinysec [6], MiniSec [7], and uTESLA [8], are not suitable because of the limitations of computation, energy, and memory in WBAN nodes. Noncryptographic security schemes are then explored to be workable for authentication purpose in WBAN. For example, Cherukuri et al. are among the first to discover biometric-based authentication in wireless networks of biosensors implanted in human body [9].

The same question comes with the popularization of WBAN, that is, how to guarantee the security property of the claimed authentication protocol. Formal method such as model checking mentioned above is first of all expected for implementing automatic verification on the noncryptographic authentication protocols. However, there are many

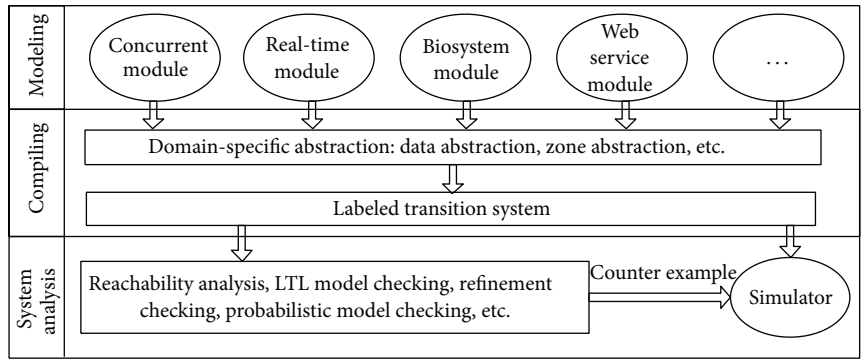

Figure 1: The architecture of PAT.

different model checking tools existing with different modeling languages, and the protocol modeling process for security verification is always rigorous and tedious that needs large professional knowledge and experience even if the final verification process could be automated using the corresponding tools. Therefore, developing some specific modeling language, as well as specific verification tool, for different application domains becomes a useful research topic and the trend of model checking technology. To the best of our knowledge, there is no research effort on formal specification and automatic verification tool specialized for security modeling and verification for both WSN and WBAN.

This paper just explores the latest model checking framework tool PAT [10], which supports building its own specific model checker, to develop a formal modeling and verification method for location-based specific kind of security protocols in WSN and WBAN. The rest of sections are organized as follows. Section 2 firstly introduces PAT and its modeling languages CSP#. In next section, a new data structure especially for modeling node mobility is proposed as well as the location supported specification language which is developed by extending CSP#. A detailed application on how to model and find a bug in one location-based authentication protocol of WSNs, using the proposed method, is illustrated in Section 4, and also a novel and successful application example on WBAN is finally discussed in Section 5.

2. Model Checking Using PAT

2.1. Basic Architecture of PAT. PAT (Process Analysis Toolkit) is a self-contained framework to support composing, simulating, and reasoning of concurrent, real-time systems and other possible domains. The architecture design of PAT is shown in Figure 1 [11], which has four layers separating the model checking process into three steps: compilation, abstraction, and verification. The four layers are introduced in the following:

(1) Modeling Layer: this layer identifies the domain-specific language syntax and well-formedness rules as well as formal operational semantics. For example, CSP# language can be used to model system in this layer.

(2) Abstraction Layer: abstract technique can be applied into systems in this layer. For instance, partial order reduction to reduce all possible interleave execution orders of parallel processing into a particular execution order is an effective abstract technique.

(3) Intermediate Representation Layer: this layer converts the input model to some form of internal representations, which is the basis of the model checking algorithms.

(4) Analysis Layer: this layer contains reusable model checking algorithms. For each intermediate representation in IRL, a set of algorithms are developed. For example, deadlock checking, reachability checking, LTL verification, and refinement checking have been developed for LTS. If the verification result is false, a counterexample can be produced, which is visualized via the simulator.

Because of the high scalability, we can abstract modeling language which can be recognized by PAT directly. The language we abstract can be converted to model language automatically. Therefore, in this paper, the modeling language specific for location-based wireless sensor networks security protocols can be proposed.

2.2. CSP# Modeling Language. The model language CSP# (Communicating Sequential Programs) offers great flexibility on how to model network protocol. It supports low-level shared variables, arrays, and general programming syntax such as for and while. Also, it employs high-level modeling operators including parallel composition, interleaving, and asynchronous message passing channel. Furthermore, CSP# semantic framework supports an interpretation of state/event Linear Temporal Logic.

Now, we will introduce a case study on security verification using CSP# approach, Needham Schroeder Public Key Protocol. The model should contain two honest agents which send and receive message by each other. The views of protocol sequences are as in Algorithm 1.

CSP# can describe the running sequence of protocol as the input model for PAT. The processes representing the communication between A and B are in Algorithm 2.

A's view:
Message 1: A sends to B: $\{A, Na\}_{\text{pk}B}$ //Na is the nonce produced by A and pk$B$ is the public key of B.
Message 2: A gets from B: $\{Na, Nb\}_{\text{pk}A}$ //Nb is the nonce produced by B and pk$A$ is the public key of A.
Message 3: A sends to B: $\{Nb\}_{\text{pk}B}$
B's view:
Message 1: B gets from A: $\{A, Na\}_{\text{pk}B}$
Message 2: B sends to A: $\{Na, Nb\}_{\text{pk}A}$
Message 3: B gets from A: $\{Nb\}_{\text{pk}B}$

Algorithm 1

```
//Initiator A
PIni(sender, receiver, nonce) =
ca! receiver. nonce. sender. receiver -> //{A, Na}pkB
ca? receiver. nonce. g1. sender ->
cb! receiver. g1. receiver -> //{Nb}pkB
//Responder B
PRes(receiver, nonce) =
ca? receiver. g2. g3. receiver ->
ca! receiver. g2. nonce. g3 -> //{Na, Nb}pkA
cb? receiver. nonce. receiver ->
```

Algorithm 2

```
public class Location = ExpressionValue {
    public int x;
    public int y;
    public int CalculateDistance(Location other) {...}
    ...
};
```

Algorithm 3

As we can see, the processes described by CSP# are simple and easy to understand. The protocol employs public key system, where pk$A$ and pk$B$ are the public key of A and B and Na and Nb are the nonce produced by A and B.

3. Extending CSP# to Model Location Behavior

3.1. Node Mobility Specification. Considering the mobility of node, we should add the location information to the model of WSN. Therefore, a method to abstract node mobility is proposed which is based on CSP# extension. To represent the location information, a two-dimensional coordinate as a natural way is required (see Algorithm 3).

Class location contains a two-dimensional coordinate of node; CalculateDistance() function is used to compute the Euclidean distance. We assume the location information of nodes A, B is (x_a, y_a) and (x_b, y_b). The node transmission distance is represented by Trans_R. If inequality $((x_a - x_b)(x_a - x_b) + (y_a - y_b)(y_a - y_b)) < \text{Trans}_R \times \text{Trans}_R)$ is satisfied, the distance between A and B is just within an acceptable range.

In order to represent all the possible mobility, the following statement is required:

$$[\,]x\colon \{0 \cdots \text{Trans}_R\};\ y\colon \{0 \cdots \text{Trans}_R\}\ @\ (P\text{Init}\ (\text{Node}A,\ \text{new Location}\ (x, y))\ |||\ P\text{Res}\ (\text{Node}B,\ \text{new Location}\ (x, y))).$$

The values of both x and y are between 0 and Trans_R, and the total range is within a square whose length and width are Trans_R. [ ] denotes value arbitrary choice, and ||| denotes processes PInit and PRes running interleaved.

3.2. Data Structure-Based CSP# Extension. Using the location class, the key of node can be presented as shown in Algorithm 4.

Class key contains key information based on ID and location, where "type" denotes key type with public or private (see Algorithm 5).

Class KnowledgeSet can store message knowledge in the process of protocol execution such as the random nonce and key information. MessageUnit denotes knowledge identification and object denotes what MessageUnit contains. For example, SortedList $\langle\langle \text{Location}, A\rangle, \text{loc}A\rangle$ denotes node A's location information.

Employing location, key, and knowledge, new data structure WSNNode can be further defined which is shown in Table 1.

WSNNode denotes an abstract WSN sensor, which contains information for the protocol such as node ID, Trans_R, location, knowledge, and random nonce.

4. On Location-Based Security Protocol for Wireless Sensor Networks

4.1. Location-Based WSN Protocol Introduction. Wireless sensor networks are often deployed in unattended and hostile environments, leaving individual sensors vulnerable to security compromise. Therefore, Zhang et al. proposed a location-based compromise-tolerant security mechanism [12], which was designed on identity-based cryptography. It can effectively restrict the threat of attack to a limited range. The mechanism assumes a trusted authority (TA) does the following operations in the initial stage:

(1) TA determines a q-order additive group G_1, a q-order multiplicative group G_2, and a pairing map $\widehat{e} : G_1 \times G_1 \rightarrow G_2$. Select an arbitrary generator P of G_1.

```
public class Key = ExpressionValue {
        public int Type;
        public int ID;
        public Location Loc;
        ...
}
```

Algorithm 4

```
public class KnowledgeSet = ExpressionValue {
        public SortedList <MessageUnit, Object> list;
        ...
}
```

Algorithm 5

Table 1: WSNNode class framework specific for WSN sensor modeling.

Class	Variable	Method
WSNNode	(i) ID: int	+getID(): int
	(ii) Loc: Location	+getLocation(): Location
	(iii) Nonce: int	+getNonce(): int
	(iv) TransRadius: int	
	(v) Knowledge: KnowledgeSet	
	(vi) PrivateKey: Key	
Location	(i) Axis$X$: int	
	(ii) Axis$Y$: int	
Key	(i) ID: int	+getID(): int
	(ii) Loc: Location	+getLocation(): int
	(iii) Type: int	
KnowledgeSet	Utype: enumeration	

(2) Pick a random $s \in Z_q^*$ as the network master secret and set $P_{\text{pub}} = sP$.

(3) Choose two cryptographic Hash functions: H_1 and H_2; $H_1 : \{0,1\}^* \rightarrow G_1^*$, $H_2 : G_2^* \rightarrow \{0,1\}^*$.

(4) Calculate for each node A an ID-based key (IBK for short): public key $Q_A = H_1(\text{ID}_A) \in G_1^*$ and private key $d_A = sQ_A \in G_1^*$.

Each node A possesses its public key $LQ_A = H_1(\text{ID}_A \parallel l_A)$ and private key $ld_A = sLQ_A = sH_1(\text{ID}_A \parallel l_A)$ denoted by ID_A and l_A.

After the initial process, any two neighboring nodes should validate each other's network membership. Figure 2 shows an instance of location-based neighborhood authentication, where node B is the neighbor of both A and C, while A and C are nonneighbors of each other. Nodes A and B are taken, for example, to explain the neighborhood authentication process:

(1) $A \rightarrow *$: ID_A, l_A, n_A,

(2) $B \rightarrow A$: $\text{ID}_B, l_B, n_B, h_{K_{BA}}(n_A \parallel n_B \parallel 1)$,

(3) $A \rightarrow B$: $h_{K_{AB}}(n_A \parallel n_B \parallel 2)$.

At start, node A locally broadcasts an authentication request including its ID_A, location $l_A = \langle x_A, y_A \rangle$, and a random nonce n_A. Upon receipt of such a request, node B first needs to ascertain that the claimed location l_A is in its transmission range by verifying if the Euclidean distance $|l_A - l_B| \leq R_B$. If the inequality does not hold, node B simply discards the authentication request; otherwise, B unicasts a reply to node A including ID_B, location l_B, a random nonce n_B, and an authenticated message computed as $h_{K_{BA}}(n_A \parallel n_B \parallel 1)$. Upon receiving the reply, node A also first checks if $|l_A - l_B| \leq R_A$. If the inequality is satisfied, A finally returns to B a message code computed as $h_{K_{AB}}(n_A \parallel n_B \parallel 2)$.

The keys used for generating the message codes actually satisfy the following equality:

$$\begin{aligned} K_{BA} &= e\left(ld_B, LQ_A\right) = e\left(sLQ_B, LQ_A\right) \\ &= e\left(sH_1\left(\text{ID}_B, l_B\right), H_1\left(\text{ID}_A, l_A\right)\right) \\ &= e\left(H_1\left(\text{ID}_B, l_B\right), sH_1\left(\text{ID}_A, l_A\right)\right) = e\left(LQ_B, ld_A\right) \\ &= e\left(ld_A, LQ_B\right) = K_{AB}. \end{aligned} \tag{1}$$

Therefore, we can derive the following two results: $h_{K_{BA}}(n_A \parallel n_B \parallel 1) = h_{K_{AB}}(n_A \parallel n_B \parallel 1)$ and $h_{K_{AB}}(n_A \parallel n_B \parallel 2) = h_{K_{BA}}(n_A \parallel n_B \parallel 2)$.

ld_A is possessed by node A alone, so does ld_B. Therefore, we can guarantee K_{AB} can be computed only by node A and K_{BA} can be computed only by node B. Subsequently, node A compares $h_{K_{AB}}(n_A \parallel n_B \parallel 1)$ computed by itself with $h_{K_{BA}}(n_A \parallel n_B \parallel 1)$ received through B. If the two results are equal, node A considers B an authentic neighbor. As the same, if $h_{K_{BA}}(n_A \parallel n_B \parallel 2)$ of node B is equal to what A sent, node B regards node A as an authentic neighbor as well.

4.2. Legitimate Sensor Node Modeling. The model of WSN security protocol contains two legitimate sensor nodes

```
//the process of the initial A:
PInitiator(SenderNode, locX) =
    IniRunning_A {...} ->    //initialize paramenters
    ca!SenderNode. everyone ->
    cb?Node_Y. Sk_Y.pk_Y. NonceA. NonceX. CONSTANT_1 ->
    cc!SenderNode. Node_Y_ID. Sk_Y. pk_Y. SenderNodeNonce. Node_Y_Nonce. CONSTANT_2 ->
    skip;
//the process of the responding B:
PResponder(ReceiverNode, locY) =
    ResInitial_B {...} ->    //initialize paramenters
    ca?Node_X.everyone ->
    cb!ReceiverNode. SkB. pkX. Node_X_Nonce. ReceiverNodeNonce. CONSTANT_1 ->
    cc?Node_X. ReceiverNodeID. Sk_X. pk_X. NonceX. NonceB. CONSTANT_2 ->
    skip;
```

Algorithm 6

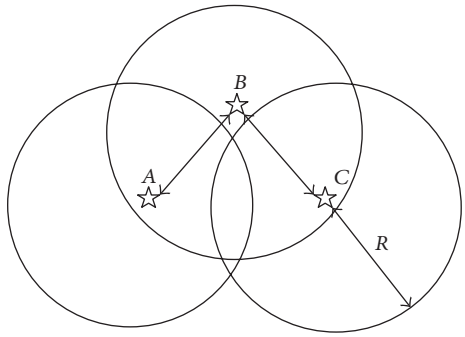

Figure 2: Node-to-node authentication between neighbors.

named A and B, where A acts as protocol initiator and B acts as protocol responder. We use two processes to describe them:

A: PInitiator(senderNode, loc$X$),

information of node A as input.

B: PResponder(receiverNode, loc$Y$),

information of node B as input.

To model Zhang's scheme, the finite-state machine is required to describe the state transition of the protocol. The state transition of initiator is shown in Figure 3, where the initial state A-Idle is supposed to skip to state A-Init automatically, then broadcasts an authentication request to all nodes in range Trans_R, and then reaches state A-Wait awaiting a response message from B. Upon receipt of such a response, A reaches state A-Check to verify whether B is in its transmission range. If it holds, A sends a confirmation message msg3 to B and reaches state A-Commit. After finishing aforementioned authentication process, A reaches state Idle again.

The same process as protocol responder is shown in Figure 4. Once receiving an authentication request, B skips to state B-Check1 to verify whether A is in its transmission range. If it does not hold, B returns to initial state B-Idle; otherwise, B responds by a message msg2 and reaches state B-Wait awaiting the response message code. If time is out, B returns back to state B-Idle. Upon the receipt of A, B reaches state B-Check2 to verify the message code received from A. If the code is confirmed, the authentication process is finished.

According to Figures 3 and 4, the main snippet codes of model are illustrated in Algorithm 6.

Processes A and B communicate to each other through channels ca, cb, and cc. The statement ca!Node_x.everyone sends a message with the values of two expressions listed to channel.

4.3. Moving-Free Adversary Modeling. Adversary model should be the most complex part because of the node moving-free scheme. Combining Dolev-Yao model [13] and node moving-free scheme, this protocol is easy to break by an intruder I in the following way:

(1) I intercepts the message sent between A and B.

(2) I sends to A or B messages by pretending to be a legitimate sensor node.

(3) I decodes $ld = sH(M)$ to find the plaintext M.

(4) I has much more powerful resources regarding energy and communication capacities than legitimate sensor nodes, so I can communicate over a high bandwidth to legitimate sensor nodes whose locations are out of range.

The abstract modeling data structure WSNNode supports adversary modeling. The type of attack can be classified as intercepting and retransmission. I can not only intercept every message sent from one node to another, but also resend messages intercepted or faked. The key snippet of adversary model is as shown in Algorithm 7.

Intruder I stores messages intercepted and decoded into object KnowledgeSet by a function Node$I$.addItem(). After that, I sends a new message combined by ID, Key, Location, and Nonce which are randomly selected.

The interfaces of message resending and faking are as follows: function RandID() is used to enumerate all the ID

```
PIntruder() =
//intercept messages from channel ca, cb, cc
    ca?Node_X.to1 ->
        InterceptChanA {
            ...
            NodeI.addItem(...); //intercept from channel ca and store knowledge decoded if possible.
        }-> PIntruder() []
    cb?Node_Y.to2.K_q_1.N1h.N2h ->
        InterceptChanB {
            ...
            NodeI.addItem(...); //intercept from channel cb and store knowledge decoded if possible.
        }-> PIntruder() []
    cc?Node_Z.to3.K_q_2.N4h ->
        InterceptChanC {
            ...
            NodeI.addItem(...); //intercept from channel cc and store knowledge decoded if possible.
        }-> PIntruder() []
//resend message intercepted or faked a new message from the knowledge decoded.
    ca!(No.RandID()).everyone.(No.RandLoc()).(No.RandNonce()) -> PIntruder() []
    cb!(No.RandID()).A.(No.RandLoc()).(No.RandNonce()).(No.RandSk()).pkA.Na.(No.RandNonce()).CONSTANT_1 ->
    PIntruder() []
cc!(No.RandID()).B.(No.RandSk()).pkB.Na.Nb.CONSTANT_2 -> PIntruder();
```

Algorithm 7

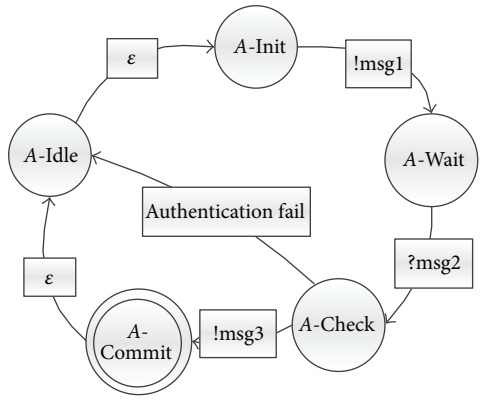

Figure 3: The state transition chart of initial *A*.

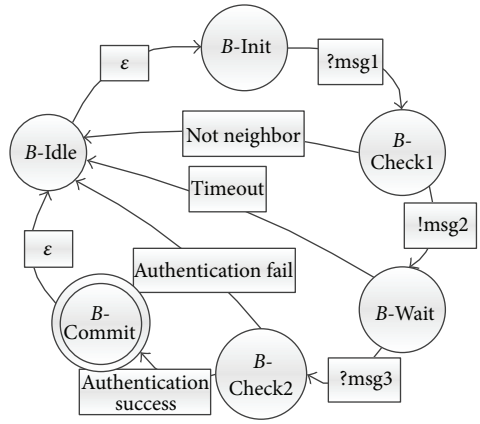

Figure 4: The sate transition chart of responder *B*.

Table 2: The symbol list for the abstract code.

Symbol type	Symbol introduction
ca, *cb*, and *cc*	Channel for message transition
Node_*X*	Message sender node
tox	Message receiver node
K_q_X	Security key of message sender node
Nxh	Nonce of message sender node

information *I* knows. Function RandLoc() is used to enumerate all the location information decoded from interception message. Function RandNonce() selects a random nonce from object KnowledgeSet. Function RandSk() is used to encrypt message by security key decoded from interception message. Using these functions can not only reduce the scale of modeling code, but also enhance the accuracy of modeling.

Intruder process runs infinitely to search all the state space. The symbols of the key code mentioned above are explained in Table 2.

4.4. Protocol Goal Property Specification. Linear Temporal Logic is applied in this paper to describe various properties that WSN security protocol should hold. The following variables are defined to facilitate the property specification:

#define iniRunning*AB* (req_*A* == true), set to 1 when initial *A* broadcasts the authentication request.

#define iniCommit*AB* (ack_2_*A* == true), set to 1 when responder *B* receives msg3.

#define resRunning*AB* (req_*B* == true), set to 1 when *B* receives the authentication request.

#define resCommitAB (ack_2_B == true), set to 1 when B receives and verifies msg3 successfully.

#define nodeMoved (moved_A == true || moved_B == true), set to 1 when A or B is moved.

#define MoveOutRadius (outRadius_A == true || outRadius_B == true), set to 1 when A or B is out of the transmission range of each other.

Using the variables mentioned above, the protocol property can be described as follows:

(1) Security: it requires that messages cannot be intercepted and decoded. We define formulas $G1$: [](([]!iniCommitAB) || (! iniCommitAB U resRunningAB)) and $G2$: [](([] !resCommitAB) || (! resCommitAB U iniRunningAB)) to ensure this property. If $G1$ does not hold, it implies that there is an adversary intercepted message from A to B. If $G2$ does not hold, it implies the adversary intercepted message from B to A.

(2) Internal authentication property: the underlying assumption here is that adversaries do not move out of the transmission range of legitimate sensor nodes. We define formula $G3$: [](nodeMoved -> <>(iniRunningAB -> resCommitAB) to ensure this property).

(3) External authentication property: the adversaries, however, might move out of the transmission range of legitimate sensor nodes and they can communicate over a high bandwidth to legitimate sensor nodes. We define formula $G4$: [](MoveOutRadius -> <>(iniRunningAB -> resCommitAB)) to ensure this property. It can judge adversary's legality by verifying the security key based on location.

Here, [] denotes always, U denotes until, and <> denotes eventually.

4.5. Verification Results and Analysis. We have performed this verification in model checking tool PAT using two legitimate processes, adversary process and LTL formulae as input. The results of $G1$ and $G2$ are true, but results of $G3$ and $G4$ are false. The counterexample of $G3$ and $G4$ is generated in Figure 5.

The counterexample illustrated in Figure 5 shows that the process stops when A sends the first message with location (3, 3). By comparing the initial location of A calls (0, 0), it is clear that node B cannot authenticate A because of the location change of A. Therefore, there is a bug in this location-based compromise-tolerant security mechanism so that it does not support the authentication when the node moves off.

5. Towards Authentication Protocol for Wireless Body Area Networks

5.1. Novel Authentication Scheme for WBAN. As we know, security in WBAN is the most important requirement to guarantee and protect the personal body parameters. Existing authentication schemes in WBAN typically use a single antenna, which are susceptible to environments. Recently, Chitra proposed SeAK, a light-weight secure device pairing protocol based on RSS obtained by dual-antenna transceivers [14]. The mechanism assumes there is one CU and one or more sensor devices to be authenticated. The CU, equipped with two antennas $A1$ and $A2$, is the only device to authenticate other devices by its two spatially separated antennas.

The formalizing description of this protocol is as follows:

(1) CU $\rightarrow$ E: Probe = $\{P_{\mathrm{CU}}, l_{\mathrm{CU}}, K_{\mathrm{CU}}\}$

(2) E $\rightarrow$ CU: Resp[0] = $\{P_0, l_0, K_0\}$.

$\vdots$

(i) E $\rightarrow$ CU: Resp[i] = $\{P_i, l_i, K_i\}$.

$\vdots$

(n) E $\rightarrow$ CU: Resp[n] = $\{P_n, l_n, K_n\}$.

The analysis steps of the protocol are introduced as below.

(1) The CU sends a probe packet Probe[i] to the device E to be authenticated from antenna $A1$, including sending power P_{CU}, the location of CU l_{CU}, and a random number K_{CU}.

(2) After receiving the packet sent by CU, device E measures the RSS indicator (RSSI) of the packet and sends a response packet Resp[0] to CU. The computational formula of RSSI is

$$P_r = \frac{P_i K_i}{d_i^{\alpha}}. \tag{2}$$

Here, d_i denotes the distance between CU and device E and α denotes an index of energy.

(3) Because of the mobility of device E, E should send a total of N packets, and CU responds with N packets by randomly switching between the two antennas $A1$ and $A2$. Let $R_1 = \{r1_1, r1_2, \ldots, r1_p\}$ represent the set of RSSIs measured by $A1$ and let $R_2 = \{r2_1, r2_2, \ldots, r2_q\}$ represent the set of RSSIs measured by $A2$.

(4) The average RSSI difference $\mathrm{RD}_{\mathrm{avg}}$ is calculated as follows:

$$\mathrm{RD}_{\mathrm{avg}} = \frac{\left((r_1 - r_2)_j + (r_1 - r_2)_{j+1} + \cdots + (r_1 - r_2)_n\right)}{n} \tag{3}$$

$j = \{1, 2, \ldots n\}$, n denotes the minimum of p and q.

(5) The CU compares $\mathrm{RD}_{\mathrm{avg}}$ with threshold $\mathrm{RD}_{\mathrm{th}}$. If $\mathrm{RD}_{\mathrm{avg}}$ is greater than $\mathrm{RD}_{\mathrm{th}}$, the device is confirmed as legitimate. Then, CU sends AssocResp[ACCEPT] message to the device.

Figure 6 shows the layout of CU and sensor devices. Node E denotes the attack node while node B denotes legitimate node. The distances between E and CU's two antennas are e_1 and e_2. As we can see, the difference between e_1 and e_2 is far smaller than d_1 and d_2, because legitimate node B is closer to CU than node E.

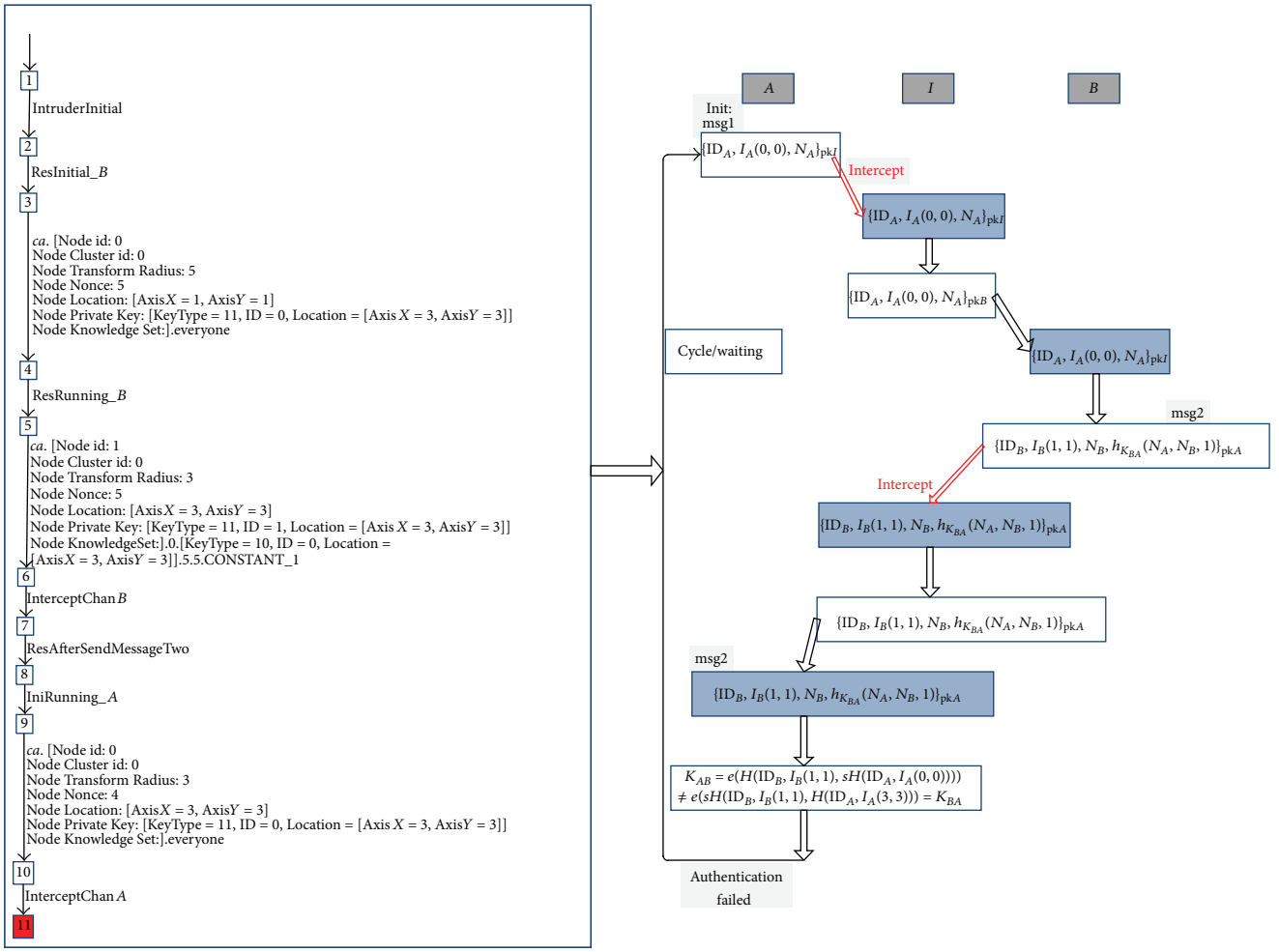

Figure 5: Counterexample of properties $G3$ and $G4$ extracted from PAT.

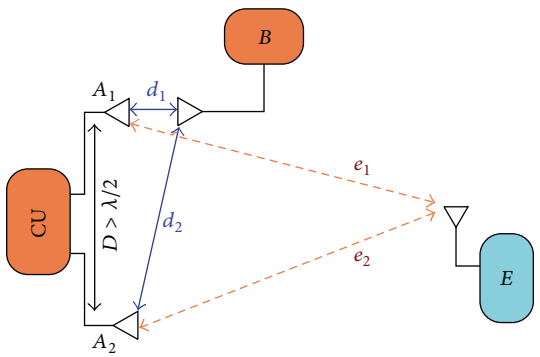

Figure 6: Layout of the SeAK protocol.

5.2. Security Modeling and Verification

5.2.1. Protocol Specification. As we can see from Figure 6, the two antennas $A1$ and $A2$ of CU are separated by D cm. The two antennas are responsible for capturing the radio signals and then calculating the received power ratio P_r which makes up the RSS value. As for devices B and E, the RSS values measured by $A1$ and $A2$ will result in a large difference as device B is placed at a distance of d_1 which is much closer than device E.

In the protocol, the RSS value is dependent only on the two distances between device and CU's two antennas. The RSS value difference is big when the device is nearby to CU, in contrast to similar RSS value difference when device is far away from CU. This allows legitimate nearby devices to be distinguished from attacker faraway devices.

5.2.2. Attack Model. In the attack model, we assume an attacker can easily vary its transmission power to get authenticated as a legitimate device. In addition, the location of an attacker is removable. This makes the attack model be the most complex part. By extending Dolev-Yao model [9], this protocol is easy to break by an intruder I using the following way:

(1) I increases the transmission power, so CU will consider it as an in-range node.

```
enum {CU_1, CU_2, ID_CU, ID_E};
#define RadiusCU 5;
#define RadiusE 3;
var<Location> lCU_1 = new Location (0, 0);
var<Location> lCU_2 = new Location (0, 1);
var<Location> lE = new Location (0, 6);
```

Algorithm 8

```
PIntruder() =
    //sends to CU messages by pretending to be a legitimate sensor node
    []ca!IDX.(No.RandLoc()).CU_1 -> PIntruder() []
    []ca!IDX.(No.RandLoc()).CU_2 -> PIntruder() []
    //intercept messages from the channel
    ca?ID_X.Loc_X.ID_E -> PIntruder() []
    cb?ID_Y.Loc_Y.ID_E -> PIntruder() []
...
```

Algorithm 9

(2) I moves into the transmission range of CU; the RSS value difference will be smaller than that out of the range.

(3) I intercepts the message sent between CU and device B.

(4) I sends to CU messages by pretending to be a legitimate sensor node.

The attack model can be divided into two parts; one is to intercept the message from the channel, and the other is to send messages to the channel which may be generated with intercepted knowledge. Here, we use the abstract modeling data structure WSNNode too (see Algorithm 8).

lCU_1 and lCU_2 are two spatially separated antennas $A1$ and $A2$. The distance between $A1$ and $A2$ is 1 cm. lE denotes the initial location information of attacker. Function enum() is used to enumerate all the ID information I knows. RadiusCU denotes the legitimate transmission range of CU. Radius$E$ denotes the available transmission range of intruder.

The behavior of intruder is uncertain, so the intrusion process may be actually an infinite loop. The key code of adversary model is as in Algorithm 9.

Here, RandLoc() is a function to enumerate all the Location information I knows. IDX is the ID who sends the message. Loc_X and Loc_Y are the location information of sender node. CU_1 and CU_2 denote the ID of two spatially separated antennas $A1$ and $A2$.

5.2.3. Security Property and Verification Results. As the same to WSN, we use LTL to describe the property which WBAN authentication protocol should hold. The following variables are again defined to facilitate the property specification:

#define iniRunningEU (req_A == true), set to true when the intruder sends an authentication request to CU.

#define resCommitEU (ack_2_B == true), set to true when the intruder completed all the N communication with CU.

#define MoveinRadius (outRadius == true), set to true when the intruder moves into the transmission range of CU.

After defining these variables above, protocol property can be described as follows:

G: #assert Protocol | = [](MoveinRadius − ><> !(iniRunningEU − > resCommitEU)).

If formula G results in True, it denotes that the intruder moves into the legitimate range of CU and has been authenticated with CU.

We have performed this verification in model checking tool PAT using two legitimate processes, adversary process and LTL formulae as input. The results of G are true. The process of authentication is in Figure 7.

As we assumed before, the initial location of intruder is (0, 6) which is out-range of CU's legitimate range. However, the intruder moved to location (0, 4) which is in-range of CU. Then, the intruder is authenticated with CU by faking itself as a legitimate device. Therefore, the authentication protocol based on dual antennas for WBAN cannot support the authentication when the intruder moves off.

6. Conclusions

In this paper, we have proposed an abstract modeling language based on CSP# to support node moving-free behavior modeling especially for wireless sensor networks, where both abstract data structure and attack model beyond Dolev-Yao are designed and developed in PAT. Using the proposed modeling method specifically for WSNs and WBAN, the automatic verification for different kinds of location-based

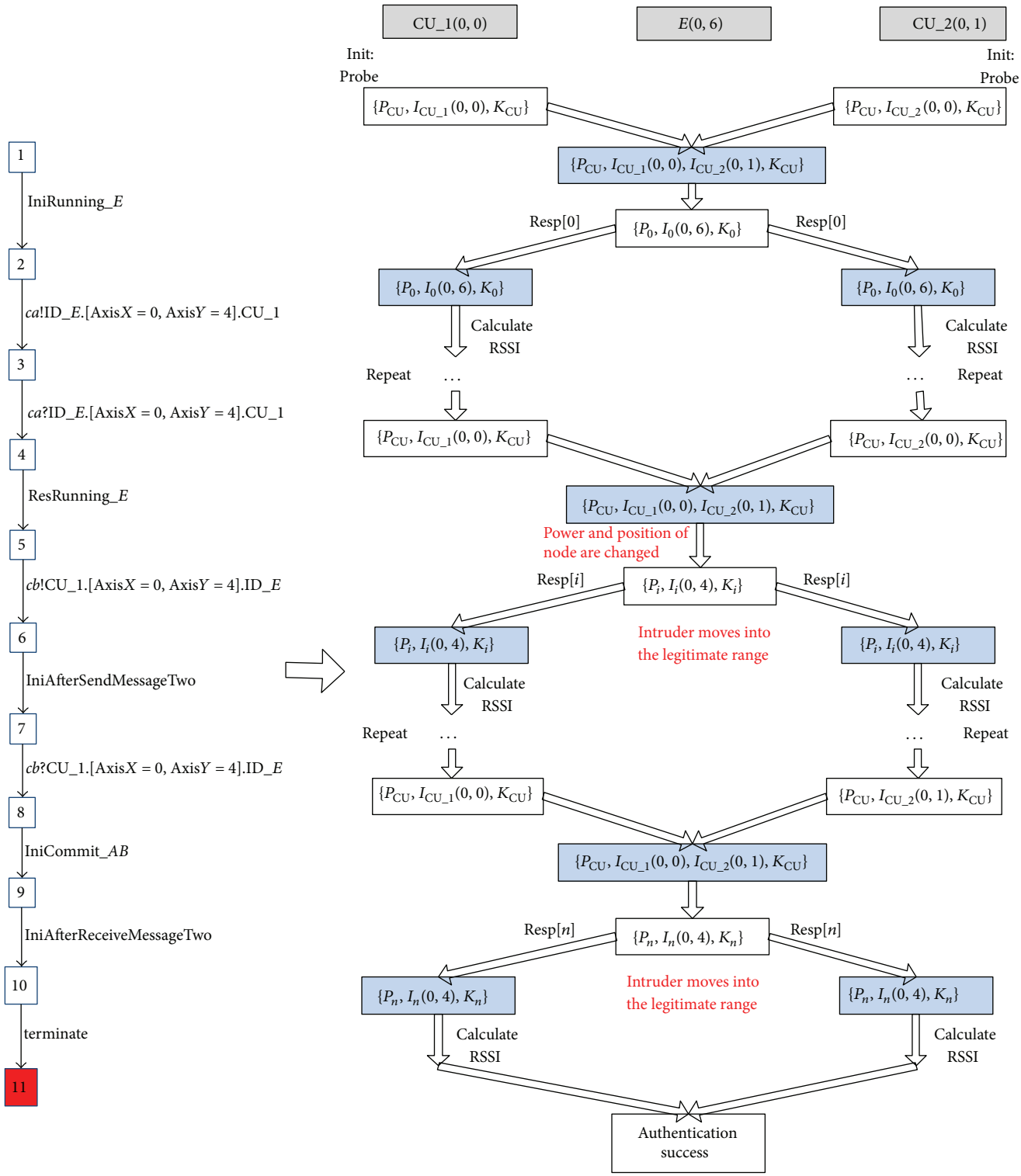

Figure 7: Verification process of property G extracted from PAT.

security protocols can be quickly implemented. At first, a bug for a compromise-tolerant and location-based cryptographic protocol on WSNs is found by employing our modeling and verification framework. Then, the latest and noncryptographic location-based authentication security scheme on WBAN is also modeled and analyzed as a successful demonstration to show the suitability and effectiveness of our proposed methodology. To the best of our knowledge, it is the first effort work on formally modeling and verifying WBAN security protocols using CSP.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work is supported by National Science Foundation of China with Grant no. 61103044.

References

[1] T. Chen, J. Jiang, X. Wang, and B. Chen, "Improved location-based compromise-tolerant security mechanisms for wireless sensor networks," *Chinese Journal of Sensors and Actuators*, vol. 25, no. 4, pp. 545–551, 2012.

[2] G. Lowe, "Breaking and fixing the Needham-Schroeder public-key authentication protocol sing CSP and FDR," in *Proceedings of the 2nd International Workshop on Tools and Algorithms for Construction and Analysis of Systems (TACAS '96)*, vol. 1055 of *Lecture Notes in Computer Science*, Springer, 1996.

[3] M. Novotný, "Formal analysis of security protocols for wireless sensor networks," *Tatra Mountains Mathematical Publications*, vol. 47, no. 1, pp. 81–97, 2010.

[4] P. Ballarini and A. Miller, "Model checking medium access control for sensor networks," in *Proceedings of the 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA '06)*, pp. 255–262, IEEE, Paphos, Cyprus, November 2006.

[5] N. Saxena, A. Roy, and J. Shin, "Dynamic duty cycle and adaptive contention window based QoS-MAC protocol for wireless multimedia sensor networks," *Computer Networks*, vol. 52, no. 13, pp. 2532–2542, 2008.

[6] C. Karlof, N. Sastry, and D. Wagner, "TinySec: a link layer security architecture for wireless sensor networks," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*, pp. 162–175, ACM, November 2004.

[7] M. Luk, G. Mezzour, A. Perrig, and V. Gligor, "MiniSec: a secure sensor network communication architecture," in *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks*, pp. 479–488, IEEE, Cambridge, UK, April 2007.

[8] A. Perring, R. Canetti, and J. Tygar, "The TESLA broadcast authentication protocol," in *RSA CryptoBytes*, pp. 2–13, UC Berkeley and IBM Research, 2002.

[9] S. Cherukuri, K. Venkatasubramanian, and S. Gupta, "Biosec: a biometric based approach for securing communication in wireless networks of biosensors implanted in the human body," in *Proceedings of the International Conference on Parallel Processing Workshops (ICPPW '03)*, pp. 432–439, IEEE, 2003.

[10] J. Sun, Y. Liu, and J. S. Dong, "Model checking CSP revisited: introducing a process analysis toolkit," *Communications in Computer and Information Science*, vol. 17, no. 7, pp. 307–322, 2008.

[11] Y. Liu, J. Sun, and J. S. Dong, "Developing model checkers using PAT," in *Proceedings of the 8th International Symposium on Automated Technology for Verification and Analysis (ATVA '10)*, pp. 371–377, Singapore, 2010.

[12] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Location-based compromise-tolerant security mechanisms for wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 247–260, 2006.

[13] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 350–357, 1981.

[14] J. Chitra, R. Girish, and L. Lavy, "SeAK: secure authentication and key generation protocol based on dual antennas for wireless body area networks," in *Radio Frequency Identification: Security and Privacy Issues*, Lecture Notes in Computer Science, pp. 74–89, Springer, 2014.