Unpredictable Software-based Attestation Solution for Node Compromise Detection in Mobile WSN

Xinyu Jin, Rodrigo Jose Salmeron, Pasd Putthapipat, Niki Pissinou, Deng Pan, Jeffrey Fan
Florida International University
Miami, Florida 33174
Email: {xjin001, rsalm001, pputt001, pissinou, pand, fanj}@fiu.edu

***Abstract*— Recent advancements in micro-computing have provided the exponential increase in the capabilities of a wide range of devices and have allowed the implementation of complex mobile wireless sensor networks (mWSNs). The common battery-powered sensor nodes require security techniques that eliminate redundant processing overhead for resource conservation, without compromising the overall network performance. To address this issue, this paper presents USAS: Unpredictable Software-based Attestation Solution, a node compromise detection algorithm in mWSNs. USAS deploys dynamic node attestation chains to decrease checksum computation time by almost 48% for selective attested nodes. By decentralizing the network, the attestation is unpredictable to prevent malicious data injection. The performance of USAS is estimated in terms of node compromise detection rate.**

I. INTRODUCTION

Node compromise is a crucial security issue in Wireless Sensor Networks (WSNs). Because sensor nodes have inherent constraints such as limited energy, processing capability and memory space, they are much more vulnerable than wired devices. Without any prevention solutions or hardware protection, sensor nodes are easily tampered with and system-critical information can be easily obtained. Researchers demonstrated the process to compromise sensor hardware in [1]. Common Internet-based security techniques are not well-suited to WSNs, due to the common assumptions of inherent schemes in which the end nodes, not the routers, are tampered with [2]. Resource limitations in WSNs also steer away from Internet marking schemes due to computational overhead. However, mission-critical applications, such as military and biomedical electronic applications, require effective techniques to protect the exchange of sensitive information.

Solutions to address node compromise focus on compromised node detection and increasing network resilience based on the inherent assumption that node compromise is an insider attack where attackers have obtained key information. The latter is under the condition that the detection of compromised nodes has occurred. The Proactive Secure Routing algorithm (PSR), which is a proactive solution that was proposed in [3], provides a secure routing protocol for transmission and improves the resilience of the entire WSN within a certain threshold.

Node compromise detection has been the subject of active investigation by researchers. Most of the previous works of node compromise detection are based on misbehavior detection or on attestation mechanisms [4]. There have been several proposed techniques using these principals and innovative research is ongoing. One such security solution is LiteWorp, which is a lightweight countermeasure for the wormhole attack that effectively detects and isolates malicious nodes in a multi-hop wireless network [5]. Comparatively, a Voting Mechanism is designed to detect misbehaving nodes based on a neighboring node monitoring mechanism in which all nodes within a relative proximity coordinate to determine if a node is compromised [6]. For physical attacks, Node Redeployment Detection is proposed based on the change of node neighborhood and the change of measured distances between nodes [7]. Using mobile agents to detect Node Compromise in PDoS attacks is proposed in [8]. Other hardware solutions include the utilization of powerful high-end sensors in Heterogeneous Sensor Networks to achieve better security and performance in [9].

Software solutions for the security of WSNs eliminate the need for excess hardware and are able to perform within the power constraints of the sensor network. Software-based ATTestation for embedded devices (SWATT) [10] is an example of a software-based security solution. SWATT is an external attestation scheme for WSNs using pseudorandom memory traversal. In SWATT, an external verifier challenges other nodes. A checksum was computed from the memory content of the device being challenged. Deviations from the memory content based on an expected value results in the detection of tampering. Every node in the network will have to perform a memory traversal using RC4 as the pseudorandom number generator (PRG). Researchers improved SWATT by decreasing iterations of memory traversal and deployed attestation schemes in distributed WSNs [11]. However, these schemes are designed for a static sensor network, need a large quantity of message transmissions between sensor nodes, and consume a considerable amount of power. We propose Unpredictable Software-based Attestation Solution (USAS), providing a both efficient and effective algorithm based on software attestation techniques, in order to improve the node compromise detection with the consideration of the large scale and mobility of mobile wireless sensor networks (mWSNs). USAS eliminates redundancy by reducing the number of nodes performing the RC4 pseudorandom number generation. The selection of the nodes performing the RC4 pseudorandom number generation will be unpredictable. This prevents certain nodes from being more susceptible to be compromised.

The contributions of the paper are as follows:

- Design an efficient software-based attestation with reduced energy consumption for battery-powered nodes.
- Create dynamic attestation chains to achieve the unpredictability of node verification to avoid creating areas of greater susceptibility in the network.
- Detect compromised nodes which are one hop away from the base station in mWSNs where the nodes are moving around without fixed neighbors.

The rest of the paper is organized as follows: Section II describes the system model, as well as assumptions. Section III explains USAS procedure implementation. Section IV describes the testing and analysis of the effectiveness of the algorithm. Finally, the discussion and a conclusion of the work are presented in sections V and VI, respectively.

II. System Model and Assumptions

USAS can be applied in both infrastructure networks and ad hoc networks. This work focuses on the applications in infrastructure networks for simplicity. The mWSN considered is of low cost and is battery- powered. Base stations are set up in proximity regions and are able to connect with mobile sensor nodes under their own coverage or to communicate with another base station. Base stations also act as network managers, which have access to a centralized database for all the sensor nodes in the network and they are protected by tamper-resistant hardware. It is assumed that the base station is a trustable entity in the mWSN, leaving the base station as the verifier for other nodes.

The network deploys a multi-level key management system. Communications are encrypted by proper keys based on LEAP proposed in [12]. However, USAS does not rely entirely on encryption for the security of the communication. A base station has the location information of each node that is currently in its coverage region. For a viable implementation of USAS, the assumption that the base station has full awareness of the capabilities of each node, including the processing speed and memory capacity, is required. Since USAS applies a dynamic node chain which is composed of one Initiator node (I-node) and multiple Follower nodes (F-nodes) to conduct attestation, the length of chain in the network must be designated. Normally, the nodes close to the base station are more vulnerable since attackers try to focus on these nodes to compromise more information sent from peripheral nodes. It is very important to detect any compromised nodes among those that are one hop away from the base station. Additionally, with the consideration of preventing malicious data injection, this work considers that the message transmission from an attested node to the base station is one hop in two-generation node chains. The relationship between the I-node and F-node will be discussed in latter section.

It is also assumed that the architecture of the mWSN does not employ virtual memory, as is the case with microcontrollers. The MEGA163L was used in the place of current architecture to allow for comparisons to be made with previous studies. The specifications for this microcontroller are as follows [13]:

- 8 bit processor architecture
- 16 KB program memory & 16 bit Addressing
- Maximum of 8 million instructions per second (MIPS)

USAS assumes that the addressing scheme used in the microcontroller is known. This will allow pseudorandom memory traversals to occur in the order of $nln(n)$ in accordance with the Coupon Collector's problem [14], where n is the number of memory blocks available. In this case, it is 8K blocks ($16 \times 8K$) [13].

III. USAS Procedure

In this section, we present USAS procedures in detail. The memory space of sensor nodes can be classified as program memory and data memory. After programming a microcontroller, the remaining program space will be filled with zeros. To prevent compromised nodes from copying the original memory pattern into remaining memory space to deceive attestation, the scheme proposed in [11] will be used; the remaining program memory will be filled with pseudorandom number before each node is deployed. Each node has a unique noise pattern which is derived from a noise generation seed called Su. Different from [11], Su will be deleted before deployment. Each node stores the one way hash function value $H(Su)$ instead of the Su value. A centralized database will store Su that is able to be accessed by base stations when it is necessary. Noise patterns for the future authentication and memory traversal of each node will also be accessible for base stations.

A. *Dynamic Node Attestation Chain*

As shown in Fig. 1, the base station triggers an attestation chain either randomly or as a result of detecting nodes misbehaviors in its coverage region. The base station will randomly select a node to be an I-node and send this node a challenge, which includes a random number as a seed for pseudorandom number generation. Along with this seed, authentication messages for the I-node and other F-nodes are included. Challenge messages to each F-node are generated by the I-node, including authentication messages and checksum values computed by the I-node. The design of dynamic node attestation chains offers the randomness for attestation initialization.

B. *Node Mobility Consideration*

To address node compromise issue in mWSNs, the mobility of nodes and dynamic network topology need to be considered.

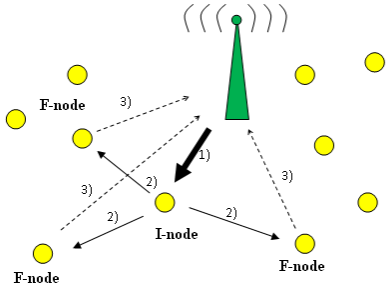

Fig. 1. Message transmissions during attestation

Solutions for static WSNs relying on fixed neighborhood relation or cluster grouping are not suitable here. USAS applies attestation solution in mWSNs by temporarily setting up dynamic node attestation chains based on the real time network topology. In our system model, each node in the network is freely moving around without a fixed relative location to other nodes. There is a constraint that USAS only attests nodes one hop from the base station which we have discussed the reason in Section II. USAS needs at least three message transmissions and two times of memory checksum computation to attest a two-generation node chain. The attested nodes are required to stay one hop away from the base station to avoid attestation message missing or malicious data injection by intermediate nodes. However, the time consumption for checksum computation is very low. We have a rough calculation later. Therefore, the impact on node velocity is trivial. There is a tradeoff between the performance of USAS and power consumption of the network. With higher power for message transmission, the node radio range is larger which leads to more attestable nodes in USAS.

C. Memory Checksum Computing

1) Message authentication

For all the attested nodes, message authentication through one way hash function computing is requested before running the memory traversal function. That is, each node computes $H(Su)$ upon receiving the challenge message, which includes unique noise generation seed Su and compares the result with the hash value stored locally. If these two values are identical, the memory traversal function is triggered.

2) Memory traversal function

USAS applies pseudorandom memory traversal. A function accesses program memory pseudo randomly and loads memory data to compute memory checksum. In order to traverse memory pseudo randomly, the pseudorandom access address is needed. In order to decrease computation time and the power consumption, USAS designs different method to generate this address for I-nodes and F-nodes.

Briefly speaking, I-nodes rely on stream cipher computation while F-nodes do not. Instead, F-nodes utilize the challenge messages from I-nodes directly. I-nodes generate memory access address by computing a stream cipher (e.g. RC4 is used as the pseudorandom number generator in SWATT [10]) based on the random number seed included in the challenge message given by the base station. The processor accesses this address to load an 8-bit memory data block. Therefore, 8 bits of a current checksum are updated in each round by using the previously computed checksum and performing an XOR operation with the current loaded memory. As a result, a 64-bit checksum is produced.

F-nodes combine a checksum sent from an I-node with the loaded memory and the updated checksum from the previous iteration to generate a 16-bit memory access address. The F-node does not have to perform a stream cipher for each round of memory traversal. The F-node procedure for memory traversal is shown in the form of pseudo code in Table I. After the message challenge from the I-node is authenticated, the F-node uses the checksum value of the I-node as the seed to traverse its program memory and compute the memory checksum. To be specific, the 64-bit checksum is separated into 8 vectors. In the first memory traversal iteration, each vector is used as the most significant byte of the memory access address. The least significant byte is a predetermined initial vector. The processor accesses this 16-bit address to load memory data and update the 8-bit checksum. After the first iteration, the most significant byte of the memory access is randomized by XORing with the updated 8-bit checksum from another vector computation. The least significant byte is updated by current loaded memory. This procedure produces a new 16-bit memory access address for the next iteration. Then the processor loads the memory data and updates the 8-bit memory checksum. Finally, the F-node produces a 64-bit checksum and sends it back to the base station.

D. Checksum Result Verification

The base station uses the same memory traversal seed to compute the program memory checksum of the I-node and then uses the result to compute the checksums of each F-node. The base station is able to traverse the original program memory pattern of each node, which is stored in the centralized database before node deployment, in order to verify the content. After comparing checksum values sent from the F-nodes with the checksum computed locally, the base station can detect compromised nodes. Although the I-node does not send its memory checksum back to base station, the base station can still detect if the I-node is compromised. Since the checksum of the I-node is used to generate pseudorandom memory addresses for the F-nodes, a genuine I-node message will allow F-nodes to compute correct checksums. As long as one F-node sends a correct response, the I-node and this F-node are considered to be trustable because the lower bound on the checksum collision probability is only 2^{-64}[10].

IV. Simulation Result and Analysis

In this section, we test the effectiveness of USAS by comparing the difference between checksum results computed based on the program memory of a compromised node and genuine program memory. Then we analyze the efficiency of USAS by calculating computation time consumption and node compromise detection rate.

TABLE I
F-node Memory Verification (Pseudo Code)

//Input: m number of iterations of the verification procedure
//Output: Checksum of memory
//Let C be the checksum vector, csfi is the vector of //checksum from I-node, and j be the current index //into the checksum vector
$for\ i \leftarrow 1\ to\ m\ do$
//Build address for memory read
$A_j \leftarrow \left(\left(csfi_j \oplus C_{((j+1)\ mod\ 8)}\right) \ll 8\right) + C_{((j-1)\ mod\ 8)}$
//Update checksum byte
$C_j \leftarrow C_j + (Mem[A_j] \oplus C_{((j-2)\ mod\ 8)}) + \left(csfi_j \oplus C_{((j+1)\ mod\ 8)}\right)_{i-1})$
$C_j \leftarrow rotate\ left\ one\ bit\ (C_j)$
//Update checksum index
$j \leftarrow (j+1)\ mod\ 8$
return C

A. *Memory Traversal Function Testing*

In order to test the effectiveness of USAS, simulations were performed for the memory traversal function procedure of each individual F-node. The input seed checksum from the I-node was given as the fixed input to the F-node. A 16 KB file was created as mock program memory of the F-node for the simulation. For simulating the compromised node, the program memory layout of the F-node varied from 1-bit difference to 16 KB difference. The memory traversal function ran 100,000 iterations in each round. For verification, the same checksum computing process ran in the original mock memory. Fig. 2 shows the number of bit differences from the genuine checksum computed by the base station versus occurrences. The difference is around the range of 20 to 40 bits even with a few bits change of the memory file.

B. *Computing Improvement Analysis*

With the functions available by the MEGA163L microcontroller, we obtain the number of operations required for proper implementation, as well as the number of clock cycles. In order to determine clock cycles, an analysis of each function used will have to be considered. From the microprocessor's technical document [13], F-nodes take 12 clock cycles in each round to update a checksum value for a single memory block in USAS. As a comparison, in SWATT, attesting each node takes 23 clock cycles [10] in each round. To put this in perspective, for 100,000 iterations of memory traversal, our implementation would take 0.15s assuming 8MIPS while SWATT would take 0.2875s. USAS decreases memory traversal computation time of selective nodes (F-nodes) by about 48%. Fig. 3 shows comparison result between SWATT and USAS. When the node density is higher, more F-nodes can be attested in one dynamic node attestation chain; therefore, more computation time can be reduced.

C. *Node Compromise Detection Rate Analysis*

Based on our system model, we derive the detection rate, which is the probability for base stations successfully detecting node compromise. The detection rate for two-generation chain attestation is considered. Assume that there are n nodes that are one hop away from multiple base stations in the entire mWSN. The probability for each node to be compromised is the same. In compromised nodes, the number of modified memory is m_c in bytes out of the total program memory size m.

The probability for base stations to detect program changes from the checksum result P_{ch} is:

$$P_{ch} = \left[1 - \left(\frac{m-m_c}{m}\right)^{m \ln m}\right](1 - 2^{-64}). \quad (1)$$

Consider that when the attested node is one of the F-nodes, successful detection of a compromised node is on the premise of the selected I-node is genuine. So detection rate P_d can be calculated by:

$$P_d = (1 - P_c).P_{ch}$$
$$= (1 - P_c)\left[1 - \left(\frac{m-m_c}{m}\right)^{m \ln m}\right](1 - 2^{-64}) \approx 1 - P_c. \quad (2)$$

where P_c is the probability for each node to be compromised.

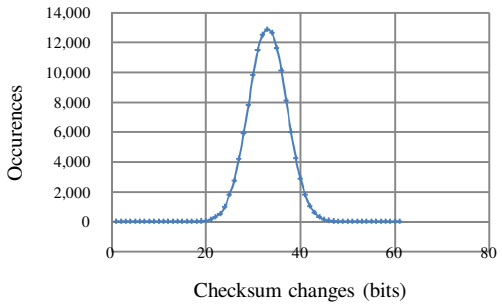

Fig. 2. Frequency description of the number of checksum

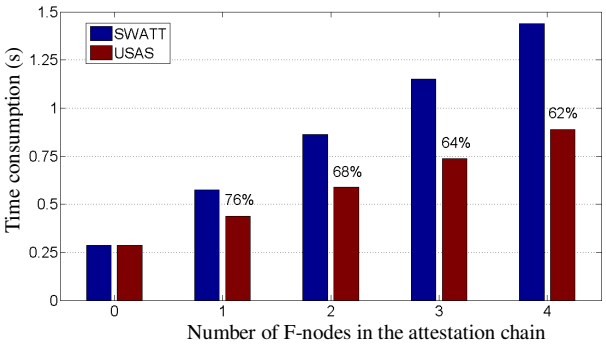

Fig. 3. Computation time comparison based on 100,000 iterations

From the view of the network, we need to select a genuine node as the I-node to detect node compromise successfully. The probability that there are i genuine nodes among the attestable n nodes is:

$$P_i = \binom{n}{i}(1 - P_c)^i P_c^{(n-i)}. \quad (3)$$

To select one of these genuine nodes as the I-node, the probability P_s is:

$$P_s = \frac{i}{n}\binom{n}{i}(1 - P_c)^i P_c^{(n-i)}. \quad (4)$$

We can derive the detection rate P_d from (1) and (4):

$$P_d = \sum_{i=k}^{n} \frac{i}{n}\binom{n}{i}(1 - P_c)^i P_c^{(n-i)}.P_{ch}$$
$$= \sum_{i=k}^{n} \frac{i}{n}\binom{n}{i}(1 - P_c)^i P_c^{(n-i)}\left[1 - \left(\frac{m-m_c}{m}\right)^{m \ln m}\right](1 - 2^{-64}). \quad (5)$$

where k is the lower bound of the number of genuine nodes among attested nodes n. In other words, at most $(n - k)$ nodes are compromised. Here we calculate the detection rate when $k = n/2$. Fig. 4 shows the detection rate when at most half of the attestable nodes are compromised with different number of attestable nodes, $n = 10,50,100$ respectively. $m_c = 100$. P_c varies from 0 to 1. When $k = 0$, it indicates the ideal detection rate calculated without considering the current status of the network. $P_d \approx 1 - P_c$. From this figure we observe the following: 1) the fewer attestable nodes, the less impact from the compromised nodes on the detection rate; 2) detection rate keeps high performance before $P_c = 1 - k/n$. Though we consider P_c and k as independent of each other, in the real network, $P_c$is the inherent security level of each node, while k depends on both P_c and network maintenance. The lower

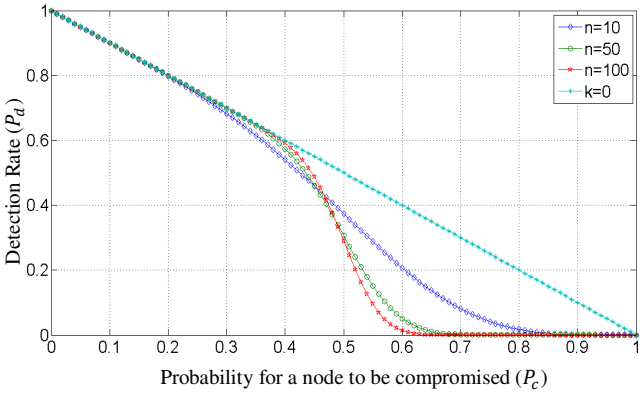

Fig. 4. Detection rate with $k = n/2$

probability for each node to be compromised (P_c) offers higher security level (k) for the network. Therefore, USAS has high performance in the realistic network status.

V. Discussion

A. Reducing Time and Power Consumption

Based on the assembly code analysis, the checksum computing for the F-node is simplified based on SWATT [10]. Therefore, the number of clock cycles for checksum function computation is reduced about 48%. Also, USAS is not constrained by the assumption that additional 'if' statements will detectably slow down the checksum computation. Furthermore, USAS does not rely on majority voting mechanisms which are commonly considered in WSNs. One of the drawbacks of majority voting is the considerable consumption due to the voting message transmissions and message distribution. In USAS, for verifying each node, only several message transmissions are required.

B. Dynamic Node Chain for Unpredictable Attestation

In our system model, the node chain relationship is dynamic and independent in different attestation. Not all nodes will be I-nodes or F-nodes at once, yet they can both store F-node and I-node algorithms since they are very similar. Also, the role of each node may vary in each time attestation. USAS helps to decentralize the network. By having a decentralized model, an attacker cannot predict which node will be the I-node. In this way, focusing on attacking a small number of nodes is not helpful to compromise the overall network. It reduces the probability of the attestation message interception and malicious data injection.

C. Applying Node Compromise Detection Solution in mWSN

In most of the previous works, node compromise detection has been studied in static WSNs. USAS utilizes the powerful base station and combines with the decentralized network structure to apply node compromise detection solution in mWSNs. It does not rely on fixed neighborhood relation. On the contrary, the mobility of the nodes benefits node compromise detection in two aspects: 1) the unpredictable I-node designation and the diffusion of attestation. 2) Each node has the possibility of being one hop away from base stations.

VI. Conclusion

In this paper, we studied node compromise detection in mWSNs. Based on our system model and assumptions, Unpredictable Software-based Attestation Solution (USAS) was presented. The simulation result shows the effectiveness of USAS for detecting any falsification of program memory of sensor nodes. By deploying dynamic node attestation chain, the attestation computation time and power consumption are improved. With the unpredictable I-node designation, the security level of mWSNs is increased.

This research has thus far only considered nodes one hop away from base stations. However, with the proper design of the dynamic node chain and the number of generations, USAS could be applied in multi-hop communications in the entire mWSN. Future work will focus on the design of attestation chain in specific applications.

References

[1] C. Hartung, J. Balasalle, and R. Han, "Node Compromise in sensor networks: the need for secure systems," *Technical Report CU-CS-990-05,* Jan. 2005.

[2] F. Ye, H. Yang, and Z. Liu, "Catching 'moles' in sensor networks," in *Proc. 27th IEEE International Conference on Distributed Computing Systems(ICDCS)*, 2007

[3] X. Chen, K. Makki, K. Yen, N. Pissinou, and Z. Liu, "A Proactive Secure Routing Algorithm Defense against Node Compromise in Sensor Networks," *IEEE Conference on Local Computer Networks,* 2008

[4] C. Kraub, M. Schneider, and C. Eckert, "On Handling Insider Attacks in Wireless Sensor Networks," *Information Security Technical Report*, vol. 13, pp. 165-172, Aug 2008.

[5] I. Khalil, S. Bagchi, and N. Shroff, "LiteWorp: Detection and isolation of the wormhole attack in static multihop wireless networks," *Journal of Computer and Telecommunications Networking*, 2007

[6] X. Lin, H. Zhu, B. Lin, P. Ho, and X. Shen, "A Novel Voting Mechanism for Compromised Node Revocation in Wireless Ad Hoc Networks," *IEEE GLOBECOM,* 2006

[7] H. Song and L. Xie, "Sensor Node Compromise Detection: The Location Perspective," *IWCMC'07,* August 2007

[8] B. Li and L. Batten, "Using Mobile Agents to Detect Node Compromise in Path-based DoS Attacks on Wireless Sensor Networks," *WICOM* 2007

[9] X. Du, "Detection of Compromised Sensor Nodes in Heterogeneous Sensor Networks," in *Proc. IEEE International Conference on Communications*, pp. 1446-1450, 2008.

[10] A. Seshadri, A. Perrig, L. van Doorn, and P. Khosla,"SWATT: SoftWare-based ATTestation for Embedded Devices," in *Proc. IEEE Symposium on Security and Privacy*, May 2004.

[11] Y. Yang, X. Wang, S. Zhu, and G. Cao, "Distributed Software-based Attestation for Node Compromise Detection in Sensor Networks", in *Proc. 26th IEEE International Symposium on Reliable Distributed Systems*, pp. 219-230, 2007

[12] S. Zhu, S. Setia, and S. Jajodia, "LEAP: efficient security mechanisms for large-scale distributed sensor networks," in *Proc. 10th ACM Conference on Computer and Communications Security*, ACM Press, 2003.

[13] Atmel, "8-bit AVR® Microcontroller with 16 K Bytes In-System Programmable Flash," ATmega163 and ATmega163L datasheet, 2003.

[14] M. Mitzenmacher and E. Upfal, *Probability and Computing:Randomized Algorithms and Probabilistic Analysis*, Cambridge University Press, 2005.