Unified Security Protocol for Wireless Sensor Networks

Abdalkahik W. Hussain[1], Mahmood K. Ibrahem[2]

[1]Network Engineering Department, College of Information Engineering Al-Nahrain University, Baghdad, Iraq
[2]Internet Engineering Department, College of Information Engineering Al-Nahrain University, Baghdad, Iraq

ABSTRACT

Securing Wireless Sensor Networks (WSNs) requires the integration of many security services due to the hostile environment, wide range of vulnerabilities and attacks, targeting this type of networks, most of the proposed security protocols for WSN focus on providing only some of those services and assume weak attacker model. The goal of this paper is to design a unified security protocol for WSN called USecWSN that integrate the necessary security mechanisms into a common architecture to achieve the required security services to defend against wide range of security attacks. USecWSN is an end to end security protocol that make use of an en-route filtering mechanisms for early detection of corrupted packets and intermediate nodes only required to compute single hash function to evaluate the packet authenticity and integrity. USecWSN uses a scalable and computation efficient replay protection mechanism which can be used to secure unicast and broadcast traffic. Moreover, USecWSN employ a load balanced cluster key distribution mechanism to reduce the overhead on the cluster head. In addition collaborative based intrusion detection system (IDS) for detecting attacks that cannot be detected by the cryptographic techniques.

Keywords: wireless sensor networks, security services, security mechanisms, end to end security, en-route filtering, key management, blom's scheme, intrusion detection system

HOW TO CITE THIS ARTICLE

Abdalkahik W. Hussain, Mahmood K. Ibrahem, "Unified Security Protocol for Wireless Sensor Networks", International Journal of Enhanced Research in Science, Technology & Engineering, ISSN: 2319-7463, Vol. 8 Issue 3, March -2019.

INTRODUCTION

Due to the numerous potential applications of wireless sensor networks in both military and civilian operations, WSNs have attracted a lot of attention. The network architecture of a WSN consists of a small and resource constrained sensor nodes that monitor their environments variables (such heat and humidity) then reports sensed data wirelessly usually through multihop communication to a centralized processing station called the sink which in turn make them available to a central information system for analysis, storage or reporting. Securing WSNs is very important especially for sensitive applications in surveillance and military fields, however ensuring secure WSN operation is very difficult to achieve due to its many vulnerabilities and sensor nodes limited resources.

In this paper we propose USecWSN, a unified security protocol for WSNs that integrate the necessary security mechanisms to secure the WSN operations. Many of proposed security protocol focused on providing only some of the required security services, which may not be enough for securing WSN against strong attacker model. In addition a storage and computation efficient modified replay protection mechanisms along with cluster key management scheme that distribute the load among member nodes are proposed.

This paper is organized as follow: section 2, review related work and highlight the main problems with their approach. Section 3 explains the attacker model and assumption. Section 4 provides USecWSN overview and contribution. Section

5 explains USecWSN design and its essential modules. Section 6 analysis the security of the proposed protocol and its resilience to security attacks. We conclude in section 7.

RELATED WORK

Many of the proposed security protocols for WSNs were designed in link layer [1, 2, 3, 4, 5, 6], which depends on hop-to-hop based security, meaning that, when a node send a packet, it encrypt and authenticate it using a secret key it shares with the next hop. The next hop can decrypt and verify received packets, then encrypt and authenticate that packet using the key it shares with its next hop and so on, until that packet reaches the sink. This approach enables intermediate nodes to have complete control over sent packets and they could compromise data confidentiality, integrity, etc. without being detected, in addition to the overhead of encrypting/decrypting and verifying/computing the message integrity code (MIC) at each hop.

An alternative approach is to design end to end security based protocol as proposed in [7, 8], where each node encrypts and authenticate their information using a secret key shared with the sink, although following this approach ensure the confidentiality, authenticity and integrity, however its vulnerable to DoS attack where attacker can inject false information into the network, that can't be detected until it reached the sink, for this reason those protocol incorporate an en-route filtering mechanisms for early detection of those types of attacks at the cost of some increase in the communication or processing overhead. An attractive en-route filtering techniques are the ones based on polynomials instead of MIC proposed in [9, 10] to ensure authentication, integrity and non-repudiation of sent data and acceptable resilience to node capture attack and its very efficient in removing falsely injected data into the WSN.

However most of those protocols are vulnerable to many security attacks launched from captured nodes (inside attacks) such as HELLO flooding, sinkhole,…, etc. because they focus only on implementing cryptographic mechanisms only. Clearly other techniques are needed to ensure a secure WSN operation like an intrusion detection system (IDS). In addition they employ some inefficient techniques for tackling security attacks namely for replay protection which result in extra overhead on the resource constrained sensor node.

ASSUMPTION AND ATTACKER MODEL

We assume that the intruder can capture some fractions of nodes chosen randomly and extract all secret information carried inside them or reprogram them to preform malicious activity; in addition attacker can eavesdrop on the traffic traveling through the network, inject, alter, replay or modify sent data. The sink is assumed to be protected and under direct control of the network administrator. On the other hand, we assume a short initialization period right after WSN deployment, where an intruder can't capture any sensor node or attack the WSN.

USECWSN DESIGN

USecWSN protocol consist of three main modules; crypto, intrusion detection system and controller modules. The description of those modules is as follow:

A. Crypto Module

This module acts as first line of defence for the WSN, by ensuring cryptographic services such as data confidentiality, authentication, integrity, non-repudiation, etc. crypto module provides several mechanisms, which includes:

1. Encryption and Decryption mechanism*:* Ensuring data confidentiality isn't enough on its own to prevent traffic analysis, an encryption scheme must also achieve semantic security preventing attacker from gaining any information about concealed data. USecWSN ensures the confidentiality and semantic security of sent application data by using RC5 in OFB mode due to their low memory overhead and energy efficiency. The IV is set such way it doesn't repeat for any secured data, the IV is 16 bytes in length and it's constructed as follow:

$$IV = DstID \| SrcID \| Seq\# \| Len \| E_i \| FCtr$$

Where *DstID* and *SrcID* are 16 bits sent data destination/source *id*, *Seq#* is a 16 bits number representing the packet sequence number, *Len* is 8-bits number representing application data length, E_i is 8 bytes number representing current epoch number (explained later), *FCtr* is 8-bits number that's increased for each packets sent in the same epoch and reset when an epoch ends, to guarantee a unique IV for each encrypted packet in every epoch and to provides weak freshness.

2. Replay Protection mechanism: For replaying attack mitigation, USecWSN employ modified scheme proposed in MiniSec, but differed in the fact that USecWSN checks only one epoch (single authentication and integrity checking operation instead of two) and it can secure unicast as well as broadcast traffic. It begins by separating time into Te length periods (epochs) $\{E_1, E_2, \dots \dots\}$, MiniSec bundle the current epoch counter E_i when computing MAC as nonce; if a packet replayed from prior window E_{i-1}, the verification process will fail. However because of network latency (τ) and motes synchronization error (D), this may causes genuine packets sent in past epoch to be excluded. So T_e values take D and τ into consideration and decryption need to be executed twice with two nominee epochs. If a node received a packet within $(t_i, t_i + \tau + 2 * D)$, two nominee for nonce are E_i and E_{i-1}, and if node acquired packet in ($t_i + \tau + 2 * D, t_{i+1}$), the two nominee values are E_i and E_{i+1}. However with preforming two validation operations for packet checking would causes extra overhead. USecWSN solution employ flag indicating odd or even epoch. Flag is 1 bit in length, if the it's set to 1 then the packet was sent during even epoch number $E_{i+1}, E_{i+3}, E_{i+5}, \dots, etc$ otherwise it's set to 0, let $T_e = \tau + 2 * D$, E_i be current period (epoch) number (even epoch), and t_i be starting of current epoch period. If received packet in $(t_i,\ t_i + \tau + D)$ and flag was set to 1, then that packet was sent during this period and nonce uses E_i , if flag was set to 0, then sending happened in previous epoch E_{i-1}. If packet within $(t_i + \tau + D, t_{i+1})$ and flag was set to 1 then packet sent during E_i , otherwise sent during E_{i+1}. Bloom filter is also employed to tackle vulnerability windows mentioned in the original MiniSec scheme. Figure 1 illustrates that idea.

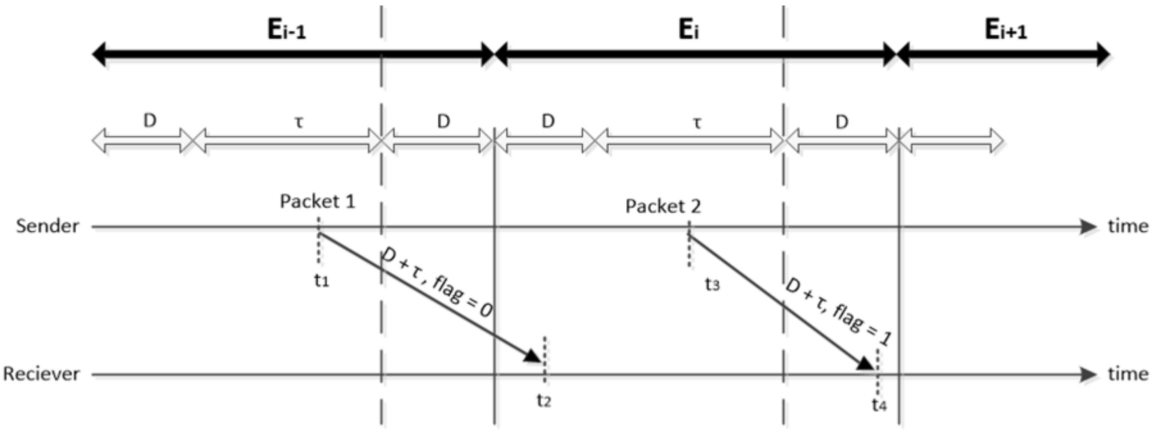

Fig. 1: USecWSN replay mitigation technique, packet 1 was sent during an odd epoch, so the flag was set to 0, and after D + τ it reaches its destination, while the flag is set to 1 in packet 2 because it was sent during an even epoch number.

3. En-Route Filtering mechanism: USecWSN implements en-route filtering scheme adapted from [10] for verifying packets authenticity and non-repudiation travelling throughout WSN. In this scheme administrator produces global polynomial $p(x, y, z)$ of degree x,y and z employed to produce two polynomials: an authentication and checking polynomials given in (1) and (2), and then preloaded into each sensor node:

$$auth^u(y, z) = \alpha * p(u, y, z) \quad (1)$$

$$chk^u(x, z) = \beta * p(x, u, z) \quad (2)$$

where *u* is nodes id, while α and β are variable chosen from specific set to increase resilience to node compromise. When a node (e.g. *id* = *u*) needs to send a packet, it first computes a hash function over packet fields as follow: Ѓ

$$\phi = H\left(M \,\|\, DestID \,\|\, SrcID \,\|\, Len \,\|\, Ei \,\|\, Seq\# \,\|\, FCtr\right) \quad (3)$$

where *H(.)* is the hash function, while *M* is the carried payload, then inserting ϕ into the authentication polynomial to find *MAP*, as follow:

$$MAP = auth^u(y, \phi) \quad (4)$$

Then MAP is attached to the packet and sent to next hop (e.g. *id* = *v*). The next hop validates it by inserting the hashing of the packet fields into its checking polynomial, as follow:

$$Verf = chk^{v}(x, \phi) \tag{5}$$

now setting x to packet source id in (5) and y to packet destination *id* and dividing it over *MAP* in (4). If result is within $\{1, 2, 4, 8, 16, 32, 64, 128\}$, then the packet is valid, otherwise it'll be discarded as follow:

$$q = \frac{chk^{v}(u, \phi)}{auth^{u}(v, \phi)} \Rightarrow \tag{6}$$

$$\frac{\beta * p(u, v, \phi)}{\alpha * p(u, v, \phi)} \Rightarrow q = \beta / \alpha \tag{7}$$

where β is randomly selected from {32, 64, 128, 256} and α is randomly selected from {2, 4, 8, 16}. Notice that intermediate nodes only needs to compute a single hash function to validate the sent packet which significantly reduces the computation overhead on them, in addition unlike the original scheme, packets doesn't carry any time stamp for protecting against replaying attack.

4. Key Management mechanisms: key management confirms the confidentiality of information wandering throughout the WSN. USecWSN has three key types each of size 128 bits each; (1) pairwise key, (2) individual key (key shared with the sink) and (3) group key. This scheme depend on Blom's mechanism [16] (modified by Du to make it more suitable for WSN [17]) for preparing nodes with secret information instead of a master key, because if the master key is known entire WSN is down, while in Blom's mechanism attacker need to capture ε nodes to completely compromise WSN. This submodule has two phases:

- Secret Information Initialization phase: Using Blom's mechanism, admin update WSN nodes with Blom's scheme related secret information.
- Key Establishment Phase: nodes in this phase begins to discover their neighbors and establish the required security keys:

A node starts this phase by discovering its neighbors, by disseminating a HELLO messages publicizing its *id* along with other information:

$$X = u \| * \| HELLO \| Len \| Ei \ \| FCtr$$

$$u \rightarrow * : X \ \| MAP(u, H(X))$$

where u is node *id* and *HELLO* is packet kind. When nodes finish propagating their HELLO messages, each node had built table containing its neighbor's *ids*. When a node in WSN wants to find common key with its neighbor, Blom's mechanism is used to produce initial mutual key between them, then passed through hashing function plus their *ids*, to create pairwise key.

$$K_{uv} = H(K_{m} \| u \| v)$$

Following the same procedure a sensor node, generate the sink key by passing the initial key through a hash function along with sink *id* to generate it:

$$K_{ub} = H(K_{m} \| u \| b)$$

In USecWSN the group key (K_G) distribution is load balanced among group members instead of relying on group (or cluster) head to distribute it. First assume that group head id is *u*, while group members have *ids* v_i (where $i = 1, 2, \ldots, n$), the steps of the proposed scheme:

A. First node u creates K_G and authenticates it using *MAP*.

B. After that *u* broadcast packet containing only *MAP* (without K_G) to all members.

C. After that *u* sent K_G to node v_1 encrypted using mutual key (K_{u,v_1}).

D. Receiver decrypts K_G and checks K_G using previously acquired MAP from *u*. If passed then v_1 send K_G to mote v_2 encrypted with common pairwise key.

E. Other members, repeat step 4 until all nodes in the group obtain K_G .

B. IDS Module

Crypto module can only protect the WSN against attackers with partial capabilities, so a second line of defence is needed, an IDS module which monitor and analyse network events to disclose any intrusion. The designed IDS module is a cooperative based IDS, where every node listens promiscuously to neighbours communication, builds behaviour profile for nodes, and compare them to stored attacker profile (signatures), if a match is found, then this information is shared among neighbours and reported to sink. The description of this module is as follow: a node monitor packets passed through the channel within its radio range and progressed for collection and computing statistics. Information extracted from packets analysed and stored in table called the Audit List (A_List) which contains:

1) Received Signal Strength Indicator (*RSSI*): average for all RSSI values of packets sent by a neighbor per collection period. Monitoring this metric helps in revealing HELLO flooding and wormhole attack.
2) Packet Receiving Rate (*PRR*): average number of packets received by a neighbor node per-collection period. Monitoring this metric helps in revealing HELLO flooding and sinkhole attack.
3) Packet Dropping Rate (*PDR*): average number of packets per detection period, the IDS heard and their destination was a neighbor, but they weren't perceived to be progressed by that neighbor. Monitoring this metric helps in revealing blackhole and selective forwarding attackers
4) Packet Corruption Rate (*PCR*): represent the average number of packets sent by neighbor per collection period their destination was that neighbor, but one of the packet fields was affected such as carried payload or for just delaying packet. Monitoring this metric helps in revealing path based DoS and packet delaying attack.
5) Packet Retransmission Rate (*PRTR*): average number of captured packets sent more than once by a neighbor per collection epoch. Monitoring this metric helps in revealing packets replaying attack.

After the collection period elapsed, the A_List containing nodes behaviors are subtracted from the Statistics List (S_List). S_List values are set during initialization period after the WSN was deployed, nodes keep listening to neighbors and computes mean value from several A_List, then stored in S_List, when a node wants to detect intruder it subtract its current *A.RSSI, A.PRR, A.PDR, A.PCR,* and *A.PRTR* (from A_List) from values in S_List and compare it to threshold values for these metrics stored in the Threshold List (T_List). A node that has value greater value than any threshold in T_List it's considered as malicious, and added to the Suspect List (U_List). T_List built by simulating WSN in presence of attackers then analyzing collected data in each mote to extract attacker profile (signature) containing: *T.PRR, T.PCR, T.PRTR, T.PDR* and *T.RSSI* values, then saved in T_List and preloaded into each node. Algorithm 1 shows how the IDS module uncovers any WSN intruders. When any node uncovers malicious activity, it exchanges knowledge by broadcasting its U_List. Every node sets timer for receiving reports from other nodes and buffer them until timer expires then analyze received reports. A node in U_List is considered an attacker if it was accused by half (or more) of reports and added to the M_List and reports it to the sink, otherwise it's considered as normal node.

Algorithm 1: INTRUDERS DETECTION

```
Input: Audit List (A_List)
Output: Suspect List (U_List)
begin:
   if T.PCR < A.PCR- S.PCR then
Add(U_List, A.NodeID, "Packet corruption or delay attack")
end
   if T.PRR <A.PRR- S.PRR then
Add(U_List, A.NodeID, "Sinkhole or HELLO flooding ")
end
if  T.RSSI < A.RSSI-S.RSSI then
Add(U_List, A.NodeID, "HELLO flooding or Jamming")
end
   if  T.PRR < A.PRTR-S.PRTR then
      Add(U_List, A.NodeID, "Replay attack")
   end
    if  T.PDR < A.PDR-S.PDR then
      Add(U_List, A.NodeID, "Selective forwarding or Blackhole")
    end
return U_List
```

SECURITY ANALYSIS

A. Confidentiality and Semantic Security

USecWSN uses RC5 in OFB mode with 16 bytes IV for ensuring confidentiality and semantic security. USecWSN bundle the IV with 8 bytes epoch counter E to keep track of the current epoch window counter, however since this counter only increases every Te which is the order of several seconds, two packets sent during the same epoch windows to the same destination may have the same IV value, here where FCtr comes which get incremented every packet get sent within the same E value. FCtr reset to zero at the beginning of a new E.

B. Authentication, Integrity and Non-repudiation:

USecWSN attach an MAP to any sent packet by the sensor nodes. Every intermediate or final destination node has to compute single hashing function to validate the attached MAP for authentication, integrity and non-repudiation, one attractive feature of the employed en-route filtering mechanism, is that even if an attacker has the authentication/checking polynomials obtained from captured nodes, he can't modify other nodes sent reports without being detected.

C. Weak Freshness and Replay Protection

USecWSN uses the *FCtr* counter within the same epoch windows to order the received packets. USecWSN uses sliding windows and bloom filter for protection against packets replaying for unicast and broadcast traffic by dividing the time into series of epochs indexed by *E* of length *Te*, every epoch *Ei* is assigned a bloom filter where packets sent during that epoch are inserted, this will ensure with some probability that previously sent packet within this epoch can't be replayed.

D. Availability

USecWSN employ network based cooperative based IDS installed at each node, for monitoring their neighbor's behavior and analyze it to uncover any possible intrusion such as packet dropping, report modification, etc.

E. Resilience to Security Attacks

To demonstrate the resilience and effectiveness of USecWSN in detecting and recovering from security attacks, the proposed protocol was simulated with 60 nodes deployed WSN in a 100x100 m^2 grid using Castalia, a wireless sensor and body area networks simulator [18] with fours nodes were captured by an intruder and reprogrammed to preform inside active attacks, namely HELLO flooding and delaying attack. The comparison was performed with WSN that doesn't have any security enabled with evaluation metric includes average packet delivery ratio (*PDR),* end to end delay and consumed energy. Table 1 shows the selected simulation parameters.

Table 1: Simulation parameters

Parameters	Value	Parameters	Value
Path Loss Exponent	2.4	Type	CC2420
Signal Delivery Threshold	-100dbm	Rate	250Kbps
Sink	1	Transmission power (T_p)	-7dbm
Sigma	4	Protocol	TMAC
Nodes	60	Packet Rate	0.2 packet/sec
Area	$100 * 100\ m^2$	Routing protocol	GPSR
Deployment Type	Grid	MAC protocol	TMAC
x, y and z	2	T.PCR	2
T.PRR	20	T.PDR	17
T.RSSI	7	T.PRTR	12
Application layer overhead	16 bytes	Network layer overhead	38 bytes
Data link layer overhead	18 bytes	Physical layer overhead	6 bytes

1. Delay attack*:* After capturing four nodes, the intruder reprograms them to delay every received packet. USecWSN detected delaying by analyzing packet designated for its neighbors and comparing its receiving and sending time, by collecting and comparing them to TPCR values stored in the T_List, if exceeded the threshold value, then that node is considered as an attacker. In addition in the sliding windows approach, every packet has a restricted time frame, if passed it and then it becomes un-authenticated because node uses the determined *E* value as nonce when verifying the MAP which provides protection against this type of attacks. Figure 2 shows the average energy consumption, while Figure 3 shows the end to end delay comparison between secured and unsecured WSN under this type of attacks.

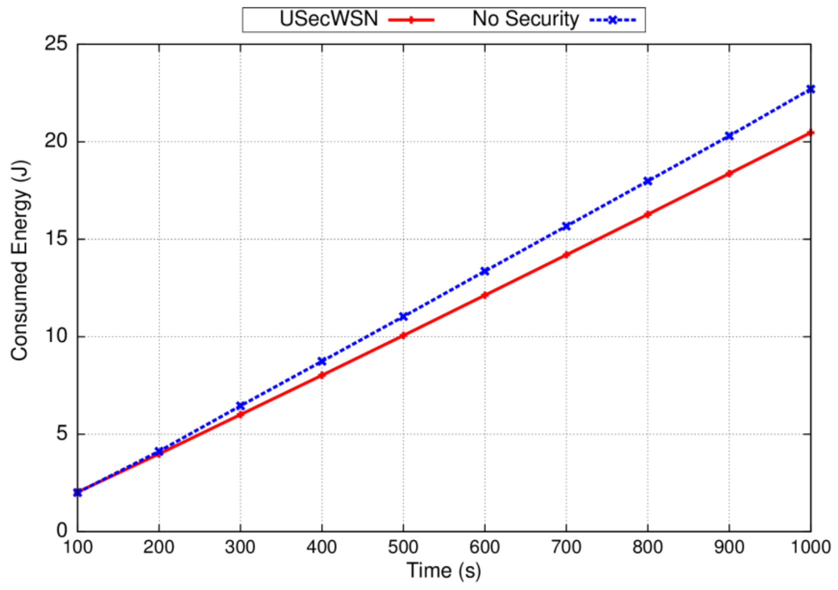

Fig. 2: Average consumed energy for the delay attacker scenario

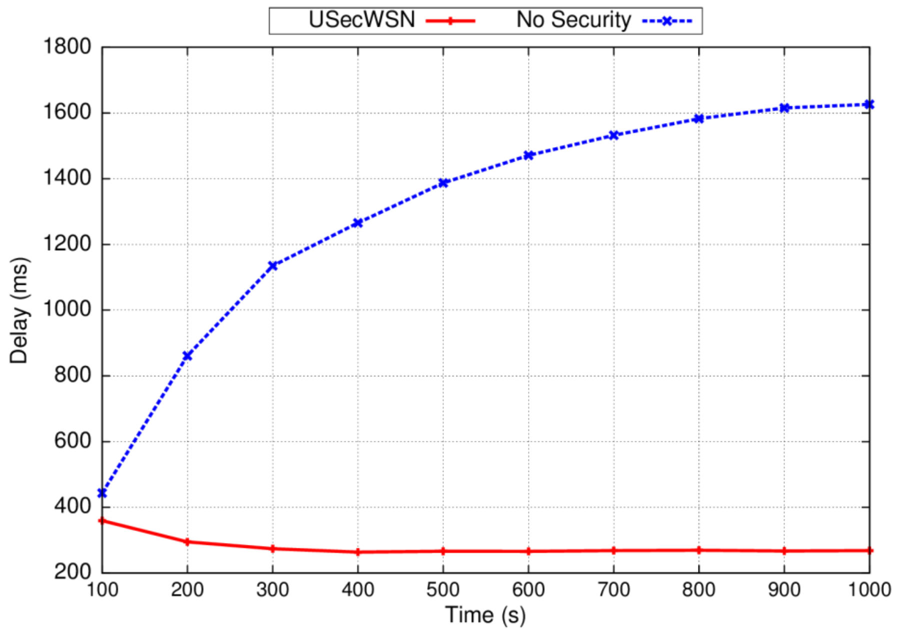

Fig. 3: Average end to end delay for the delay attacker scenario

2. HELLO Flooding Attack*:* After capturing four nodes, the intruder reprogram them to broadcast a HELLO advertisement packet announcing optimal route to the sink, in GPSR protocol this can be done by announcing a location near the sink in the HELLO packet and broadcasting it using very high transmission power to attract packets from nearby and faraway nodes. Distant nodes using their low transmission power sending packets to the attacker will cause a lot of retransmission, as these nodes will not receive an acknowledgment from the attackers causing extra energy consumption and dropping in the packet delivery ratio. USecWSN forces motes to receive HELLO only from registered neighbor to tackle high transmission power of attackers. In addition the IDS module constantly gathers RSSI and PRR values from neighbor nodes, after that compare it to those nodes stored profile and if any node have values exceeded T_{prr} and T_{rssi} stored in the T_List, then it's considered as an attacker. USecWSN consumed less energy and had better PDR than unsecured WSN as displayed in Figure 4 and Figure 5 respectively.

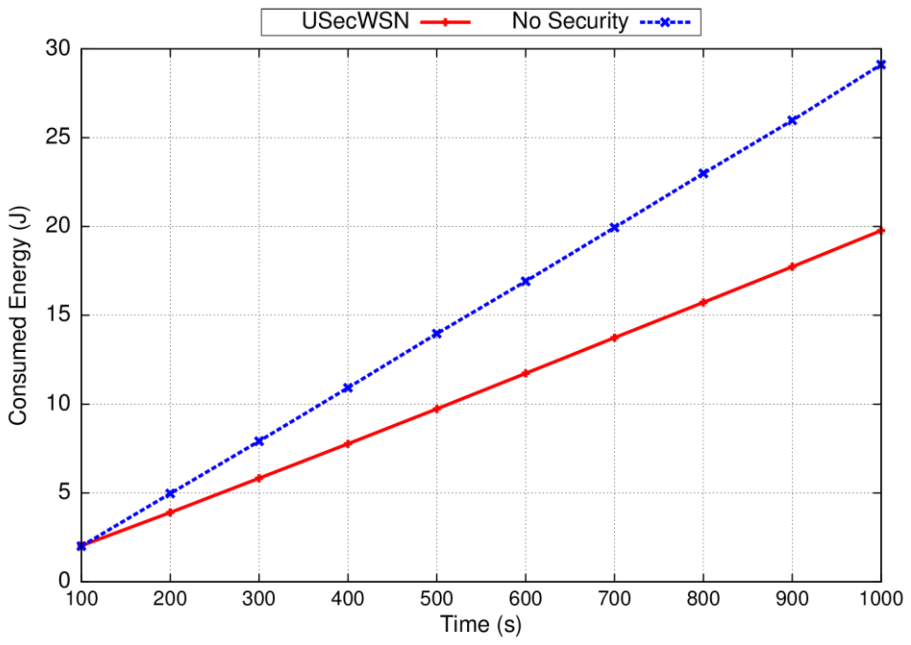

Fig. 4: Average consumed energy for HELLO flooding scenario

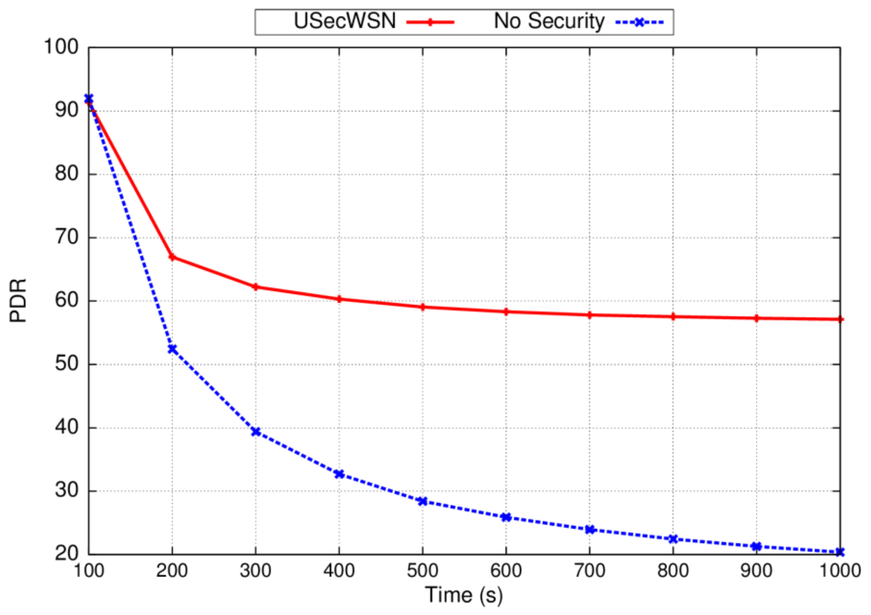

Fig. 5: Average PDR for HELLO flooding attack scenario

CONCLUSION

Due to WSN many vulnerabilities and weakness that made them an easy target for attackers, so any security protocol meant for WSN must tackle them. USecWSN is a WSN security protocol integrating many security mechanisms including cryptographic and IDS techniques into unified architecture to ensure maximum security taking into account limited capabilities of sensor nodes. The simulation of two inside attackers preforming delaying and HELLO flooding insider attackers showed that USecWSN can successfully detect and recover from security attacks. USecWSN ensures end to end secrecy of sent information is that only the final destination can decrypt the sent reports. Moreover USecWSN incorporate a storage and computation efficient replay protection mechanism for securing unicast and broadcast traffic. Another attractive feature is that intermediate nodes only needs to compute single hash function to validate any received packets and detect any injected/corrupted reports along with load balanced group key management scheme for distributing the overhead of key establishment on all group members with scalable and computationally efficient replay protection scheme. USecWSN integrate an IDS module for detecting any malicious activity within the WSN that can’t be prevented by using cryptographic techniques.

REFERENCES

[1] Zhu, S., Setia, S., Jajodia, S.: 'LEAP: efficient security mechanisms for large-scale distributed sensor networks', Pro. 10th ACM Con. on Computer and Communications Security, USA, Washington, October 2003, pp. 62 - 72.

[2] Karlof, C., Sastry, N., Wagner, D.: 'TinySec: A link layer security architecture for wireless sensor networks', Proc. 2nd Int. Con. on Embedded Networked Sensor Systems, USA, Baltimore, November 2004, pp. 162 - 175.

[3] Park, T., Shin, T.: 'LiSP: A lightweight security protocol for wireless sensor networks', ACM Transactions on Embedded Computing Systems, 2004, 3, (3), pp. 634 - 660.

[4] M., Luk, Mezzour, G., Perrig, A., Gligor, V.: 'MiniSec: A Secure Sensor Network Communication Architecture', Int. Symposium on Information Processing in Sensor Networks, UK, Cambridge, April 2007, pp. 479 - 488.

[5] Moh'd, A., Aslam, N., Phillips, W., Robertson, W., et al., 'A dual-mode energy efficient encryption protocol for wireless sensor networks', ScienceDirect, 2013, 11, (8), pp. 2588 – 2604.

[6] Tsou, Y., Lu, C., Kuo, S., et al.,: 'MoteSec-Aware: A Practical Secure Mechanism for Wireless Sensor Networks', IEEE Transactions on Wireless Communications, 2013, 12, (6), pp. 2817 - 2829.

[7] Ren, K., Lou, W., Zhang, Y. et al.: 'LEDS: Providing Location-Aware End-to-End Data Security in Wireless Sensor Networks' IEEE Transactions on Mobile Computing, 2008, 7, (5), pp. 585 - 598, May 2008.

[8] Alzaid, H., and Alfaraj, M.: 'MASA: End-to-End Data Security in Sensor Networks Using a Mix of Asymmetric and Symmetric Approaches', New Technologies, Mobility and Security, Morocco, Tangier, November 2008, pp. 1 - 5.

[9] Zhang, W., Subramanian, N., Wang, G.: 'Lightweight and Compromise-Resilient Message Authentication in Sensor Networks', 27th IEEE Con. on Computer Communications, USA, Phoenix, April 2008.

[10] Yang, X., Lin, J., Yu, W., Moulema, P., Fu, X., Zhao, W., et al.: 'A Novel En-Route Filtering Scheme Against False Data Injection Attacks in Cyber-Physical Networked Systems' IEEE Transactions on Computers, 2015, 64, (1), pp. 4-18.

[11] Perrig, A., Szewczyk, R., Wen, W., Culler D., Tygar, J., et al., 'SPINS: Security Protocols for Sensor Networks' Wireless Networks, 2002, 8, (5), pp. 521 – 534.

[12] Uluagac, A., Li, Y., Copeland, J. et al.: 'VEBEK: Virtual Energy-Based Encryption and Keying for Wireless Sensor Networks' IEEE Transactions on Mobile Computing, 2010, 9, (5), pp. 994 – 1007.

[13] Ren, J., Lighfoot L., Li, T., et al.: 'An energy efficient link-layer security protocol for wireless sensor networks', IEEE Int. Con. on Electro/Information Technology, USA, Chicago, May 2007, pp. 233 - 238.

[14] David, R., Marchany R., Midkiff, S.: 'Scalable, Cluster-based Anti-replay Protection for Wireless Sensor Networks', Information Assurance and Security Workshop, USA, West Point, June 2007, pp. 127-134.

[15] Jinwala, D., Patel, D., Dasgupta, D., et al.: 'FlexiSec: A Configurable Link Layer Security Architecture for Wireless Sensor Networks', Journal of Information Assurance and Security, 2009, pp. 582 - 603

[16] Blom, R.: 'An Optimal Class of Symmetric Key Generation Systems', Pro. of EUROCRYPT 84, A Workshop on the Theory and Application of of Cryptographic Techniques, France, Paris, 1984, pp. 335-338.

[17] Du, W., Deng, J., Han, Y., Varshney, P., et al.: 'A pairwise key predistribution scheme for wireless sensor networks', ACM Transactions on Information and System Security, 2005, 8, (2), pp. 228 – 258.

[18] Pediaditakis, D., Tselishchev, Y., and Boulis, A.: 'Performance and scalability evaluation of the Castalia wireless sensor network simulator', Con. on Simulation Tools and Techniques, Belgium, Brussels, 2010, p. 53.

[19] Vogt, H.: 'Protocols for Secure Communication in Wireless Sensor Networks', PhD Thesis, Swiss Fedrel Institute of Technology, Zurich, 2009.