Mitigating Location Spoofing Attack in Underwater Sensory Networks

Lakesh Biyala, Samyak Jain, B. R. Chandavarkar
Department of Computer Science and Engineering
National Institute of Technology Karnataka
Surathkal, Mangalore, India
lakeshbiyala126@gmail.com, samyakjain677@gmail.com, brcnitk@gmail.com

***Abstract*—In a Sensory Network, the location of a node is critical information hence if that is compromised it can lead to adverse effects on the entire network. Therefore safeguarding it from attacks and the discovery of spoofing is very important in underwater communication especially where the environment is such attack prone due to high bit error rate, low bandwidth and high delays of the acoustic channels. It is already a challenging task to detect any spoofing attacks in UWSNs. In this paper, we propose an anti-spoofing algorithm for mitigating localization spoofing attack in UWSNs that makes use of the AES encryption algorithm along with SHA-1 hash function and implemention of it in UnetStack.**

***Keywords*—UWSNs, location spoofing attack, AES, SHA-1, UnetStack**

I. Introduction

Underwater wireless sensor networks (UWSNs) have shown a growing interest in recent years [1]. A variety of applications can be implemented where each operation is important in its own context, but some of it may improve maritime exploration to meet a wide range of underwater applications, including a disaster awareness program (i.e., tsunami and earthquake tracking), assisted navigation, maritime data collection, tactical surveillance and water monitoring, etc. Underwater Acoustic Sensors networks (UASNs) provide a new communication platform. The challenges that arise in UASNs are due to low bandwidth and variation in latency surrounding impacts because of doppler shift, water currents and multi-path fading, etc. cause problems such in localization, routing and security. Muhammad Khalid et al [2] gave a detailed comparison between all the routing protocols in UWSNs. Localization based protocols need the prior geographic location of all the nodes in the network where localization free protocols work without location information and work on flooding-based mechanisms [3]. Vector Based Forwarding (VBF), Location-Aware Source Routing (LASR), Focused Beam Routing (FBR), etc. In almost all routing algorithms, the physical location of the nodes in the network plays a crucial role in the routing of the data inside the network. Therefore it is of utmost importance to ensure the location information of the nodes is protected from the attackers.

The remaining paper is organised in four subsections namely, Section II Literature Survey, which briefs about the previous work done related to UWSNs, Section III which is Understanding of Location Spoofing and its impacts, In Section IV, the Proposed Methodology suggests our way to mitigate the Location Spoofing issue. Following that, we describe a detailed implementation of the Proposed design in UnetStack. Finally, the last Section V presents the conclusion of the paper.

II. Literature Survey

This section of the paper-primarily outlines the research of the different potential attacks, different efforts made in the past to mitigate and existing mitigation measures. In this article, [4] we have focused on the unique characteristics of the acoustic communication channel and studied possible attacks and countermeasures of communication protocols in UASNs.The research of UASNs is still in the initial stage. In the design of communication protocols, only limited kinds of malicious attacks (e.g. jamming attacks, wormhole attacks) are considered. In addition, most studies do not take node mobility into consideration, and their computational complexity is relatively high, which ultimately introduces high communication overheads and energy consumption. In order to obtain an efficient and secure communication protocol, the following basic requirements should be satisfied.

1) Security
2) Robustness
3) Energy efficiency
4) Lightweight Protocol

Secure communication protocols are always designed based on interactions among sensory nodes to detect malicious nodes and ensure the integrity of sensory data. In addition to integrity and correctness, the privacy and confidentiality of data packets should also be protected. Under such a disaster-prone environment, the attacks to obtain and manipulate the location of the nodes becomes a facile task. This article [5] presents a survey of different technical approaches to prevent or at least to detect location spoofing in the network. This paper [6] shows a unique way to mitigating localization and neighbour spoofing attacks in UASN. Their implementation uses three pre-shared symmetric keys to hide the location data and hash functions to later verify the integrity of the location data shared to the location requesting node. In [7] Yan Li, Liang Xiao, Qiangda Li and Wei Su propose an authentication scheme that applies the hypothesis tests to detect spoofing attacks in UWSNs. Here interactions between a spoofer

and a surface station in UWSNs was formulated as a zero-sum authentication game. Their spoofing detection method was based on Q-learning which was proposed for dynamic underwater sensor networks.

III. Location Spoofing attack and its Impacts

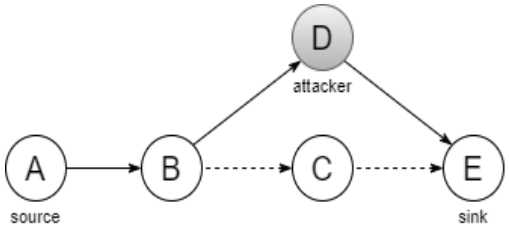

Fig. 1: Understanding Location Spoofing

For the communication to occur in the network, the nodes have to form routes with other nodes and then the exchange of data takes place. This process is initiated first by a single node requesting the location of its neighbouring nodes to form routes with it. As the response to the request, the nodes send their address and a route is added in the routing table of the initiating node. This process is carried forward and the network is formed and communication takes place through the routes formed. However, if the exchanging of the location data is intercepted by a malicious node it can lead to malpractices and eventual failure of the network.

In Location Spoofing, the attacker sends the false location data to other genuine nodes when they request the location of the nodes to form routes. With this, it gains their confidence. Whenever two distance nodes want to communicate they need to set up a route using genuine intermediate nodes between them and data is transmitted hop to hop, However, when a malicious node imitate an intermediate node and alter its location in such a way all packets comes at the malicious node then it might modify the data and before forwarding to next hop. Now this genuine node replies with its location information to attend to the request. This received location information of the genuine node is used by the malicious node and now it shows itself acts as the genuine node and reply back to the requesting node. Requesting node has a sense that it is directly communicating with the genuine node as however it is connected to the malicious node acting as a genuine node with its location information.

In Fig. 1 we see the source A wants to send data to sink node E. The node B being the closest to node A, node A forms a route with it. Then node B tries to make a route with its neighbouring nodes. Here the node D spoofs the location of node C and thus node B forms a route with attacker node D. Here the connection is formed between node B and node E through node D even though node C would provide the optimal distance.

With the location spoofing, the entire network is unaware of the presence of a malicious node. This node can cause attacks such as Denial of Service attack and Man-in-middle attack.

IV. Proposed Methodology

1) Proposed Design: Anti-Spoof works with three symmetric keys (a, b, c) shared with all deployed nodes as a prerequisite. Here its real-time location coordinates (x, y, z) are encrypted before sharing using the AES (Advanced Encryption Standard) cryptographic algorithm [8]. Each key is used to encrypt each coordinate. This results in encrypted location coordinates (x', y', z'). In addition to the encryption done, the hash of actual location coordinates is calculated using the hash function SHA-1 (Secure Hash Algorithm 1) [9] cryptographic hash function. The use of the Hash function provides integrity when the sharing of the location coordinates data. The encrypted location coordinates and the hash value are shared with the requester for verification.

Algorithm 1 Proposed Algorithm

Require: Pre-set shared keys a, b, c
Ensure: Hiding Actual Location x, y, z
Modified Location m
$x' \leftarrow$AES.enc(x, a)
$y' \leftarrow$AES.enc(y, b)
$z' \leftarrow$AES.enc(z, c)
$h \leftarrow$SHA-1(x, y, z)
$m \leftarrow x', y', z', h$

2) Mitigating localization spoofing attack: In this case, the malicious node transmits false localization data to the requesting node to form a route to it in the disguise of a genuine node by spoofing it. However, Anti-spoof depends on three pre-allocated keys in the encryption process and computing these keys by the malicious node by using brute force is impossible. Since the malicious node doesn't know the keys it tries to guess the key set, it ends up using a completely different key set (a', b', c'). This will lead to hash inconsistencies at the attacker side, therefore requesting node will discard the received location and the node with not be considered for route formation.

3) Implementation: For the development of the UASN simulation, UnetStack has become a go-to tool for researchers and developers because of its efficiency and seamless transition to the real world field deployment of the development projects

A. Overview on UnetStack

Unetstack [10] is a collection of software agents representing different stack layers. It contains a component called an Agent which offers services with respect to different layers in the network stack. These agents interact with other agents by using messages. In Unetstack message is of 3 types: requests, responses and notifications. Request message asks an agent to perform a task whereas a response message is a reply to the requesting agent. The notifications are unsolicited information generated by agents. A service is a collection of messages (requests, responses and notifications) and parameters that an agent honors. Services often define optional capabilities. The major services offered by Unetstack are Datagram service, Physical service, Baseband service, Link service, MAC service, Routing service, Transport service, Ranging ser-

vice and Node Information service, Remote service, Address resolution service and Route maintenance services.

B. Topology of Node Network

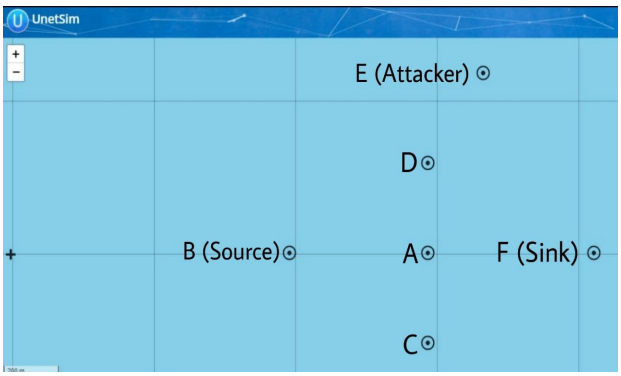

Fig. 2: *Topology in UnetStack*

Fig 2. shows the topology implemented in UnetStack for the demonstration of Location Spoofing Mitigation. Here are six nodes (A, B, C, D, E, F) deployed in 3-dimensional underwater space. Node B is the source node and node F is the sink node. Node B want to send some data to node F since they are located far apart, it required a certain number of hops to communicate with node F. So the source tries to develop routes to the sink. The remaining nodes E, D, A, C are the possible nodes to form routes. Here node E is assumed to become the attacker node. So node E tries to spoof the location of the other nodes (D or A or C) while B is trying to form the routes.

C. Implementation of Attacker Node

In order to implement a malicious node in UnetStack, one of the nodes in the topology are treated as the attacking node. We build an agent in such a way that it emulates the malicious working of the attacker node and add that to the malicious node. The attacker modified location data with the address of some other node in the network, Then it relays that address to the requesting node. In this way, the attacker spoofs the address of the legitimate node after it responds to the incoming request by the requesting node with its spoofed location data resulting in gaining the trust of the requesting node and establishing itself as a genuine node in the network.

D. Implementation of Solution

The Anti-Spoof in UnetStack implemented in two parts; one at the neighbouring nodes which send its location information as shown in Fig 3 and the other at the requesting node which receives the information as shown in Fig 4.

To implement the mitigation at requesting node, an agent extending UnetAgent is defined. The startup() method is overridden to subscribe to various agent service like PHYSICAL, ROUTING, NODE_INFO, to fetch node address, send `DatagramReq` and a request for location data, is broadcast using `DatagramReq` packet and waits for the reply from available neighbours. Upon receiving location data from the neighbours having MAC Protocol [11], which is sent as encoded Protocol Data Unit (PDU), which is then decoded and data is extracted.

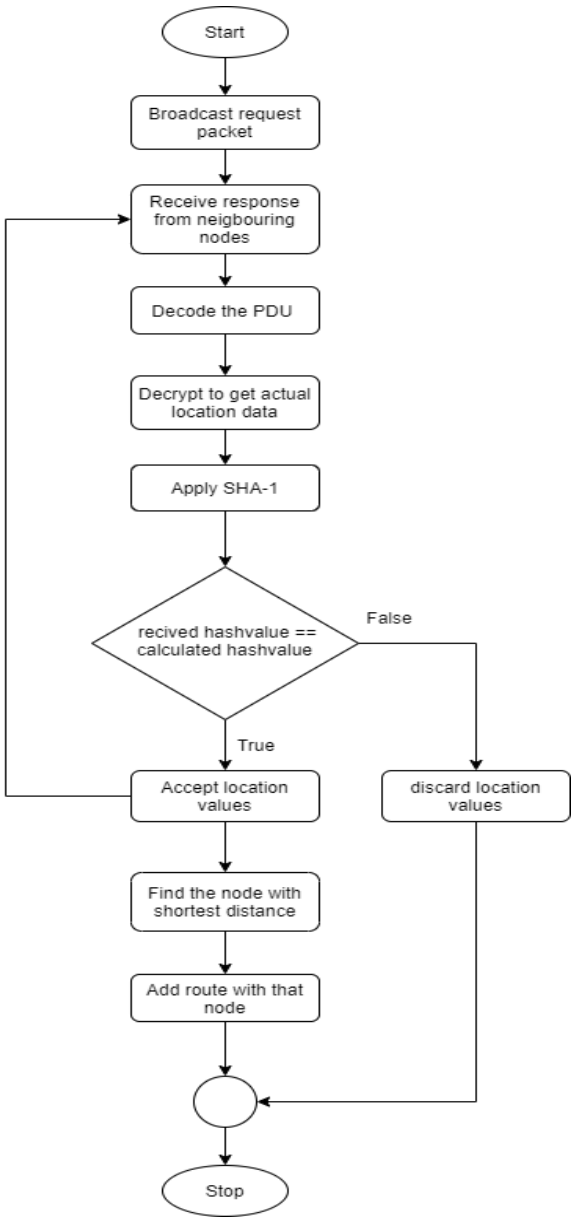

Fig. 3: Working at Source Node

Then the AES-encrypted location coordinates are decrypted using the shared key set. The decrypted coordinates are used to regenerated the hash value using SHA-1, The SHA-1 hash algorithm is implemented in the agent using Java Security API, which generates a base16 hash string, further, a hash code is generated for this hash string to reduce the size of hash. Thereafter, the recomputed hash is compared with the received hash; if both are matched, the location data is accepted, indicating the data is received from the legitimate neighbouring node, else it is dropped. Further, upon accepting location coordinate values, the distance is calculated using the Euclidean distance formula, and the routing table is updated with node address as next-hop having shortest distance for the In Location Spoofing, the attacker sends the false location data to other genuine nodes when they request the location of the nodes to form routes. With this, it gains their confidence.

Algorithm 2 Working at Source node

At startup(): BROADCAST location request
Received response from neighouring nodes
while Timeout period ≥ 0 **do**
  Decode the PDU and fetch data
  decrypt the location data
  $x \leftarrow$AES.dec(x', a)
  $y \leftarrow$AES.dec(y', b)
  $z \leftarrow$AES.dec(z', c)
  Recomputing hash ch using SHA algorithm
  $ch \leftarrow$f(x, y, z)
  $rh \leftarrow$ Recieved hash
  if ch == rh **then**
    Accept location values
    Add route with the shortest distance nexthop
  else
    Discard location values
  end if
end while
return nexthop

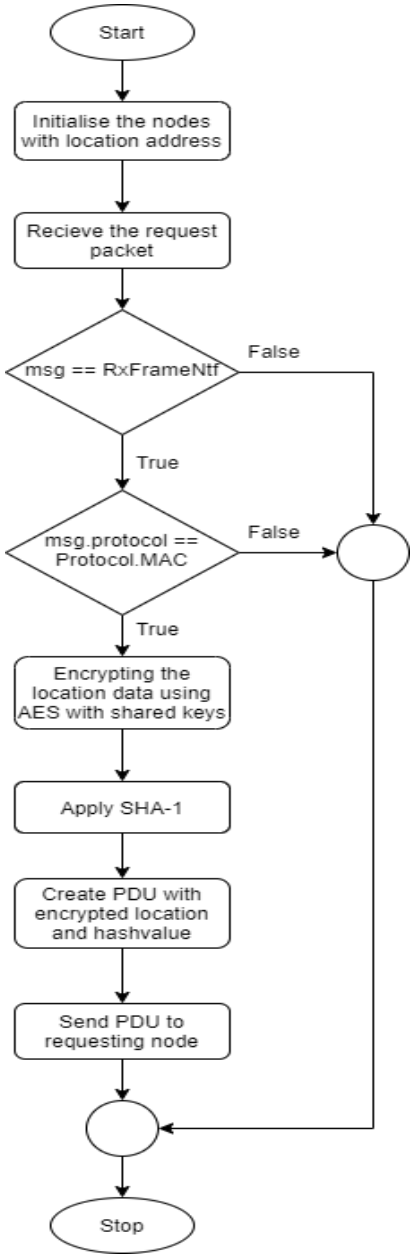

Fig. 4: Working at Neighboring node

On the other part, to implement mitigation at neighbouring nodes, an agent extending UnetAgent is defined. The UnetAgent startup() method is overridden to fetch node address and initial location values and then the location values are encrypted using the shared keys. This location information remains the same and is sufficient to send for a requesting node if the node remains stationary. To receive the request from requesting node the processMessage() method is overridden in which upon receiving a `RxFrameNtf` having MAC Protocol containing location request, the current location values are fetched and encrypted, then the hash is generated to the plain location data. Hash is generated using the SHA-1 hash algorithm which is implemented in the agent using Java Security API, which generates a base16 hash string, further a hash code is generated for this hash string to reduce the size of the hash. The encrypted location coordinates data and the hash code is put together in the PDU defined. Finally, the PDU created and is sent to the requesting node. To avoid a collision from multiple neighbours, the neighbours made to wait for a certain amount of time and send their data based on their distance to the requesting node. Algorithm 3 describes the detailed working of the neighbouring node.

Algorithm 3 Working at Neighbouring node

At startup(): Fetch node address and location values
At processMessage(): Received Location request PDU from requesting node
Decode PDU and extract location data
if msg == `RxFrameNtf` && msg.protocol== Protocol.MAC **then**
  Encrypting the location data
  $x' \leftarrow$AES.enc(x,a)
  $y' \leftarrow$AES.enc(y,b)
  $z' \leftarrow$AES.enc(z,c)
  $h \leftarrow$f(x,y,z)
  Create a PDU $P \leftarrow$(nodeid,x',y',z',h)
  return PDU P
else
  return NULL
end if

V. Conclusion

In this paper, we proposed a design to mitigate location spoofing in Underwater Wireless Sensory Networks(UWSNs). We have implemented the proposed design in UnetStack. However, in this design, the use of the AES cryptographic algorithm provides strong encryption of the location coordinates but it imposes a heavy computational cost on the nodes. Therefore we would suggest the use of some light-weight cryptographic algorithms like PRESENT [12] which is notable 2.5 times smaller than AES, KATAN and KTANTAN [13] - small and efficient hardware-oriented block ciphers or Tiny Encryption Algorithm (TEA) [14] for the encryption process.

REFERENCES

[1] M. Chitre, S. Shahabudeen, M. Stojanovic, Underwater acoustic communications and networking: Recent advances and future challenges, Marine technology society journal 42 (1) (2008) 103–116.

[2] M. Khalid, Z. Ullah, N. Ahmad, M. Arshad, B. Jan, Y. Cao, A. Adnan, A survey of routing issues and associated protocols in underwater wireless sensor networks, Journal of Sensors 2017 (2017).

[3] D. Shin, D. Hwang, D. Kim, Dfr: an efficient directional flooding-based routing protocol in underwater sensor networks, Wireless Communications and Mobile Computing 12 (17) (2012) 1517–1527.

[4] G. Han, J. Jiang, N. Sun, L. Shu, Secure communication for underwater acoustic sensor networks, IEEE communications magazine 53 (8) (2015) 54–60.

[5] M. Decker, Prevention of location-spoofing-a survey on different methods to prevent the manipulation of locating-technologies, in: International Conference on e-Business, Vol. 1, SCITEPRESS, 2009, pp. 109–114.

[6] B. Chandavarkar, A. V. Gadagkar, Mitigating localization and neighbour spoofing attacks in underwater sensor networks, in: 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), IEEE, 2020, pp. 1–5.

[7] Y. Li, L. Xiao, Q. Li, W. Su, Spoofing detection games in underwater sensor networks, in: OCEANS 2015-MTS/IEEE Washington, IEEE, 2015, pp. 1–5.

[8] A. Abdullah, Advanced encryption standard (aes) algorithm to encrypt and decrypt data, Cryptography and Network Security 16 (2017).

[9] D. Eastlake, P. Jones, Us secure hash algorithm 1 (sha1) (2001).

[10] M. Chitre, R. Bhatnagar, W.-S. Soh, Unetstack: An agent-based software stack and simulator for underwater networks, in: 2014 Oceans-St. John's, IEEE, 2014, pp. 1–10.

[11] S. Shahabudeen, M. Motani, M. Chitre, Analysis of a high-performance mac protocol for underwater acoustic networks, IEEE Journal of Oceanic Engineering 39 (1) (2013) 74–89.

[12] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. Robshaw, Y. Seurin, C. Vikkelsoe, Present: An ultra-lightweight block cipher, in: International workshop on cryptographic hardware and embedded systems, Springer, 2007, pp. 450–466.

[13] C. De Canniere, O. Dunkelman, M. Knežević, Katan and ktantan—a family of small and efficient hardware-oriented block ciphers, in: International Workshop on Cryptographic Hardware and Embedded Systems, Springer, 2009, pp. 272–288.

[14] D. J. Wheeler, R. M. Needham, Tea, a tiny encryption algorithm, in: International workshop on fast software encryption, Springer, 1994, pp. 363–366.