CENDA: Camouflage Event Based Malicious Node Detection Architecture

Kanthakumar Pongaliur Li Xiao Alex X. Liu
Department of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824, U.S.A.
{pongaliu, lxiao, alexliu}@cse.msu.edu

***Abstract*—Compromised sensor nodes may collude to segregate a specific region of the sensor network preventing event reporting packets in this region from reaching the basestation. Additionally, they can cause skepticism over all data collected. Identifying and segregating such compromised nodes while identifying the type of attack with a certain confidence is critical to the smooth functioning of a sensor network. Existing work specializes in preventing or identifying specific type of attack and lacks a unified architecture to identify multiple attack types. Camouflage Event Based Malicious Node Detection Architecture (CENDA) is a proactive architecture that uses camouflage events generated by mobile-nodes to detect malicious nodes while identifying the type of attack. We exploit the spatial and temporal information of camouflage event while analyzing the packets to identify malicious activity. We simulated CENDA to compare its performance with other techniques that provide protection against individual attack types and the results show marked improvement in malicious node detection while having significantly less false positives. Moreover, CENDA is able to identify the type of malicious activity and is flexible to be configured to include other attack types in future.**

I. Introduction

Sensor network finds application in different disciplines ranging from academic research to practical on field military interests. Provided the vast scope, securing a sensor network is critical given the deployment scenario wherein the nodes may be left unattended over long durations of time. The sensor network can be under attack from different types of adversaries, some intentional adversary like a malicious user and some unintentional adversary like environmental conditions.

In this paper we look at collaborative adversaries whose goal is to segregate part of the network in order to prevent event reporting by either dropping the packet or corrupting it. Also, if the events are sporadic, the basestation will not be able to differentiate between non-occurrence of event and non-report of an event due to malicious activity. This is important if you consider military applications such as detecting heavy artillery movement. There have been a lot of studies to protect the sensor network from different kinds of adversaries trying to launch varied types of attacks. Protocols like TinySec [1], SPIN [2], TinyPK [3], SERP [4] have been developed to provide security and maintain integrity of the communication data. Pirzada *et al.* [5] use a trust model like a reputation scheme to identify the sinkholes and wormholes in the network. In this scheme, they guarantee that the packet reaches the basestation using a trusted path, but do not detect the malicious nodes in network.

The research so far have provided solutions which work in an inside-out fashion, *i.e.* the onus of identifying such mal-intent is left to the sensor nodes. There have been numerous reputation based systems [6] [7] [8], which use this methodology attempting to solve the problem of intrusion detection and mal-behavior of sensor node. In [9], Ngai *et al.* provide a scheme to analyze packets to detect the intruder. The existing approaches tend to prevent a specific attack type. Even an IDS looks at identifying an intrusion without pinpointing the type of attack. What is lacking is a proactive architecture that can detect a malicious attack while also being able to identify the type of attack.

The threat model considered is a collaborative attack to segregate a region so as to disallow event packets in this region from reaching the basestation. The adversary is a laptop class attacker which can perform four attack types, namely sinkhole attack, selective forwarding, wormhole and sybil attack to achieve the same.

In this paper we look at solving the problem in an outside-in fashion in which the onus of identifying the mal-intent of a sensor node is left to the basestation. We present Camouflage Event Based Malicious Node detection Architecture (CENDA), a multiphase proactive architecture in which the basestation exploits the spatial and temporal information of the camouflage event to detect the malicious node.

A camouflage event is a reputable event generated in response to a basestation request. A simple solution is for the basestation initiating a camouflage event from the node by sending a message, but a powerful adversary can overhear the communication and allow such event packets. The camouflage event generator is a mobile-node which can be mounted on robot or an unmanned aerial vehicle. This mobile-node traverses the path decided by the basestation and generates the camouflage events at regulated intervals of time. These events are called camouflage events since they mimic the real events, but are not real events in the true sense. The nodes which are currently sensing and are in the sensing range of the camouflage event location will detect the event and report back to the basestation. We provide a lightweight address encoding scheme wherein each node encodes the relative address of the node from which it received the packet. Additionally, each node also maintains the information about the overheard packet transmissions by the neighbors. To cater to memory constraints we use a variation of the bloom-filter

This work was supported in part by the US National Science Foundation under grants CCF-0514078, CNS-0551464, CNS-0721441, and CNS-0716407.

978-1-4244-5113-5/09/$25.00 ©2009 IEEE

[10] based information storage system. This information is used for verification while branding a node as malicious. We also identify the type of attack and currently can classify this to four attack types. In future it can be expanded to include identification of other attack types.

We simulated CENDA to compare its performance with [11] [9] that provide protection against individual attack types and is seen to show marked improvement in malicious node detection while having significantly less false positives. Moreover, CENDA is able to identify the type of malicious activity and is flexible enough to be configured to include other attack types.

The paper is organized as follows. We discuss our classification of nodes and definition of metrics in Section II. Threat model is presented in Section III. As part of the architecture, we design the mobile route for the camouflage event generator in Section IV. Then we present the security architecture in Section V. Finally we present the simulation results in Section VI, followed by the conclusions in Section VII.

II. Node Classification and Metrics Definition

Let n be the number of nodes and f_i is the count of the number of nodes for which node i forwards the packet. This parameter f_i is the $Forwarding\ Number$ of node i. f_{mean} is the mean of the forwarding number of all nodes. At network initialization, the forwarding number of a node is set to be inversely proportional to its distance from the basestation.

$$f_{mean} = \sum_{i=1}^{n}(f_i/n)$$

The nodes are classified as follows:

1) Regular nodes: Nodes which forward packets for two or less nodes including itself.
$$Regular\ Nodes = \{n_i\} \forall\ i\ where\ f_i \leq 2$$
2) Routing nodes: Routing nodes forward packets for greater than two nodes but less than the f_{mean}.
$$Routing\ Nodes = \{n_i\} \forall\ i\ where\ 2 < f_i < f_{mean}$$
3) Backbone nodes: These are the nodes which forward packets for greater than or equal to f_{mean}.
$$Backbone\ Nodes = \{n_i\} \forall\ i\ where\ f_i \geq f_{mean}$$

$Critical\ Index\ (CI)$ is a per node metric, which is the availability of sensor coverage over an area. The sensing area is divided into m unit regions. Each node i senses over a set of unit areas called $Sensing\ Area$. U_i is a set of unit area's over which node i has sensing coverage. Each unit area will be monitored by a set of nodes called $Sensing\ Nodes$ represented as C_a where a is the unit areaid. C_a is a set of node's which sense over the unit area a. The $Coverage\ Inheritance$ of a node i is defined as the average of C_a where area a belongs to the set U_i.

$$Coverage\ Inheritence_i = \frac{\sum_a C_a}{|U_i|} \text{ Where } a \in U_i$$

We compute the $Critical\ Index$ of a node based on the $Forwarding\ Number$ and the $Coverage\ Inheritance$ of the node. The $Critical\ Index$ of the node is represented as follows:

$$CI_i = \alpha * f_i + \beta * \frac{1}{Coverage\ Inheritence_i} + \gamma b_i$$

where α, β and γ are constant coefficients, b_i is a boolean variable identifying if node i is a cut-vertex. If a node is a cut-vertex, then the importance of the node increases manifold, as the loss of this node can partition the network and make some nodes unable to reach the basestation.

III. Threat Model

The adversary is a laptop class attacker and on capturing a node, it has access to everything on the node. Also, the malicious entity would want to capture the nodes such that the minimum number of node captures result in the maximum damage. Being a more powerful node, it is reasonable to assume that the malicious entity can overhear communication over a larger area compared to the sensor node and can make informed decisions about the nodes it wants to attack. The malicious entity captures a node and does one or more of the following types of attacks: Sinkhole attack, Wormhole attack, Sybil attack or Selective Forwarding attack.

The goal of the attacker is to segregate a region such that event packets do not get reported from the region. The attacker may allow other communication. Let x_1, x_2, ..., x_n be the set of n nodes. The goal of the threat model is to increase the value of $Cumulative\ Attack$, where cumulative attack is defined as follows:

$$Cumulative\ Attack = \frac{\sum C_p}{|p|}\ Where\ C_p - \ critical\ index\ of\ node\ p$$

The adversary wants to compromise a node and perform attack from the list above in order to achieve the goal of region segregation. The attacks can be composite in nature *i.e.* multiple different types of attacks simultaneously.

IV. Mobile-Node Route Design

The mobile node in our scheme is the camouflage event generator. The route followed by this node governs the location and time of the camouflage event which in turn determines the nodes that will detect this event. The design of an optimized mobile node route is critical due to the following reasons:

1) A poorly designed mobile node route can cost the network dearly in the form of event detection by unwarranted nodes and having the network flooded by these packets. This consumes energy which is a limiting factor in sensor nodes.
2) The second reason is the wastage in time and energy of the mobile node to follow a non-prime path.

This section introduces the design decisions in the route to be followed by the mobile node and provides an algorithm for the same. It includes a preliminary step of selecting the nodes that needs to be observed.

A. Node Selection

Simulation results indicated that it is costly to protect all the nodes. Under attack, we need to be able to identify and protect the nodes which are critical to the network functioning. Hence in this step we prioritize the nodes based on their role/importance. This importance is dictated by different factors like location, power etc. Based on the classifications and definitions in section II, we want to select a subset of nodes such that they satisfy the following requirements.

1) Step 1: Includes all nodes with critical index of τ.
2) Step 2: Include nodes such that we have at least a unit coverage over the maximum possible area.

B. Route Design

M is the mobile route, $L_1, L_2..., L_n$ are the locations on route M where the event generation occur and n is the number of stops in a mobile route. Let X be the set of all nodes and

Algorithm 4.1 Node Selection

$\{CovArea\} \Leftarrow C_m, \{CN\} \Leftarrow \phi, X_n = \{Nodes\}$
for all $x_i \in X_n$ and $CI_i > \tau$ **do**
  $\{CN\} \Leftarrow \{CN\} + x_i$
  $\{CovArea\} \Leftarrow \{CovArea\} - U_i$
end for
while $\{CovArea\} \neq \{\phi\}$ **do**
  $MCN \Leftarrow x_i$ where $(((x_i \in X_n)$ and $(x_i \notin CN))$
    and $x_i = (max(U_n - \{CovArea\})))$
  $\{CovArea\} = \{CovArea\} - U_i$
  $\{CN\} = \{CN\} + MCN$
end while
return $\{CN\}$

$\{CN\}$ be the set of nodes to be observed. If we know the routing paths of the packets, we find the subset of nodes in X which should detect the event such that all nodes in CN either detect the event or are on the routing path of node detecting the event. Let this subset of nodes which detect the event be the set of privileged node represented by X_p. If we do not know the routing paths of the packets, then the set CN is the set of privileged node represented as X_p.

Goal: Find set of locations L_n in the field such that all nodes in X_p are at a distance of sensing range or less from at least one location in L_n. This is presented in Algorithm 4.2.

Algorithm 4.2 Camouflage Event Location Selection

$L_i \Leftarrow \{\phi\}$
while $(\{X_p\} \neq \{\phi\})$ **do**
  for all $A_m \notin \{L_i\}$ **do**
    $SenseCov_m \Leftarrow Count(Distance(X_p, A_m) \leq \tau)$
  end for
  $MaxCovArea = Max(SenseCov_m)$
  $\{L_i\} = \{L_i\} + MaxCovArea$
  $\{X_p\} = \{X_p\} - \{Node\ X_i\ Sensing\ MaxCovArea\}$
end while

In the event of having multiple mobile vehicles, the set X_p is geographically partitioned into the number of mobile vehicles available and the mobile route can be planned for each of the partitions independently. The next step is to calculate the shortest path to cover the locations in set L_i. This is formulated similar to the traditional 'Traveling Salesman Problem' but with an optimization exception to allow the mobile node to revisit any prior visited location. For this a simulated annealing approach is used to find the shortest route.

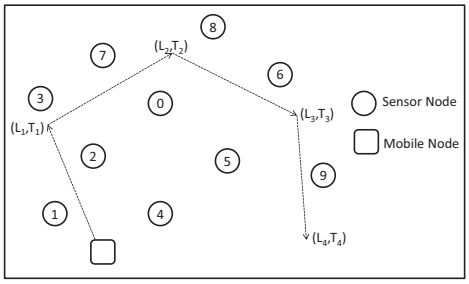


Fig. 1. Mobile Route

C. Mobile Route Representation

A mobile route is a directed graph, defined as a 2-tuple (L, T) where L is a set of locations and T is a set of time of event occurrences. Figure 1 demonstrates a scenario containing twelve sensor nodes and one mobile node. The mobile node moves to location L_1 and generates the event at time T_1. Then it moves to location L_2 followed by to location L_3 and L_4 while generating the camouflage events at times T_2, T_3 and T_4 respectively. Let v be the speed of the mobile vehicle. The mobile route is represented as follows:

$$M = (L, T)\ Where\ \forall i,\ L_i \in A_m\ ,\ T_{i+1} > T_i,\ T_{i+1} - T_i \geq \frac{L_{i+1} - L_i}{v}$$

V. Secure Architecture

The secure architecture is a standalone module, which consists of the following steps:

A. Camouflage Events

The events generated by the mobile nodes are called camouflage events. With regards to the base station, the camouflage events possess some distinct properties which are not characteristic of real world events. These properties are as follows:

1) The base station is aware of the location of the event occurrence (L).
2) The base station knows the precise time of the occurrence of the event at each location (T).
3) With the knowledge of the location and time of event occurrence, the base station knows the set of sensor nodes that detect the event (SN).

The camouflage event is represented as (L, T, SN). For a particular location the active nodes can vary if the nodes are following a sleep cycle, hence at different times the set SN can differ for the same location.

The mobile node starts the route designed by the base station. While traversing the path, it stops at the designated location and generates the camouflage event. The nodes sensing the event, respond back to the base station. The sensor nodes cannot differentiate this event from an actual event and hence it is handled like a real world event. We need to emphasize this because event type anonymity is crucial to this methodology. If a malicious node can differentiate this event from a real world event, then it can treat just these events like a well-behaved node to escape detection. In CENDA, the onus of detecting malicious behavior is on the basestation. The sensor nodes only need to record overheard packet transmissions by the neighbors.

The base station collects all the packets that were generated and have traversed through the network. For each packet received by the base station in response to a camouflage event, we have two types of nodes involved. The event notifier node which reports the occurrence of the event and the intermediate nodes through which the packet traversed to reach the base station. At a higher level every packet received by the base station provides one bit information about each of the nodes involved in the delivery of the packet.

Depending on routing, there are two distinct cases that we consider. One in which the packet follows the same path for a set period of time. In this case the basestation is aware of the path the packet traverses. In the second case, instead of following a particular route, the packet follows geographical routing. For the basestation to know the path traversed by the packet, the intermediate nodes need to append additional information. In the absence of this additional information the basestation can only make an educated guess about the intermediate nodes using the knowledge of the location of the nodes and their sleep cycles.

B. Embedding Route Information

In the absence of fixed route, we device a scheme in which the nodes append the route information to the packet before transmitting over the next hop. This address information being attached to the packet can be of two types.

1) Absolute address: In this case the node appends the absolute address of the neighbor from which it received the packet. The drawback of this scheme is the amount of space it takes to represent the absolute node address.
2) Relative address: We take advantage of the knowledge of the node distribution by the basestation and present a scheme in which the nodes appends the relative address of the neighbor node from which it received the packet.

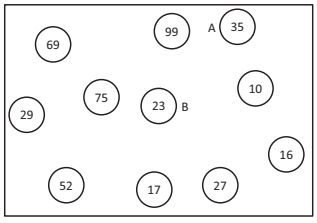
Absolute address is suited, if the number of nodes is few, resulting in shorter address. But the number of nodes in the network can be large, making the absolute address of the nodes longer. The length of the relative address is dictated by the number of neighboring nodes. To analyze the overhead disparity between the absolute address and relative addresses, we ran simulations to study the distribution of the nodes based on the following parameters- number of nodes, area of field, type of distribution. The type of distribution can be uniform or non-uniform. If the distribution is non-uniform, for densely distributed areas, the nodes are programmed such that only a set of nodes are active at a point of time. Table I displays

TABLE I
NODE DISTRIBUTION

Node	Area(m^2)	Average Hops	AAL^*	RAL^{ψ}
512	1000 X 1000	8	72	32
1000	1000 X 1000	8	80	34
2000	1000 X 1000	7	77	38
3000	1000 X 1000	7	84	38

* Absolute Address Length (bits)
ψ - Relative Address Length (bits)

the comparison between the absolute address length and the relative address length for different area sizes and different node densities. The absolute address length is calculated as the average number of hops multiplied by the number of bits needed to represent the absolute address.



Absolute Nodeid	Relative Nodeid
10	0
16	1
17	2
27	3
29	4
35	5
52	6
69	7
75	8
99	9

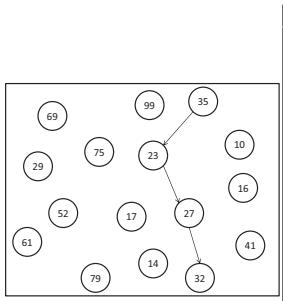
Fig. 2. Relative Neighbor Address

1) Relative neighbor address: The address and representation of the neighbor A for node B is based on two factors. The absolute nodeid of A and the number of neighbors of node B. Each node maintains a table with two fields namely, the nodeid and the relative address. The relative address are assigned in the increasing order of neighbors nodeid. In Figure 2, the relative address of neighbor nodes for node B is listed in table. We see that the relative address can be represented using four bits while the absolute address can require up to seven bits.

2) Encoding the path address: When a node receives a packet it appends the relative address of the node from which it received the packet and encrypts it after appending the relative address before forwarding it over the next hop. If node n received the packet from node m. The encryption is represented as follows:

$$[Address||RA_{m|n}]_{PK_n} \quad Where\ A||B\ -\ Append\ B\ to\ A$$
$$RA_{m|n}\ -\ Relative\ address\ of\ m\ w.r.t\ n$$

In Figure 3, the event is detected by node 35 and it is transmitted to the basestation via nodes 23, 27, 32.



Node 23

Aid^*	Rid^{ψ}
10	0
16	1
17	2
27	3
35	4
75	5
99	6

Node 27

Aid	Rid
14	0
16	1
17	2
23	3
32	4
41	5

Node 32

Aid	Rid
14	0
27	1
41	2

* Absolute Node id
ψ - Relative Node id

Fig. 3. Address Encoding

Table in Figure 3 lists the relative address for the three nodes receiving the packet. Each node before forwarding the packet over the next hop, encodes the relative address information in the packet as follows. Node 23 will encode the relative address of node 35 which is 4 before transmitting to node 27. Node 27 will encode the relative address of node 23 which is 3 before forwarding the packet to node 32 and so forth.

$$[Addr]_{PK_{35}}$$
$$\downarrow$$
$$[[Addr]_{PK_{35}}||100]_{PK_{23}}$$
$$\downarrow$$
$$[[[Addr]_{PK_{35}}||100]_{PK_{23}}||11]_{PK_{27}}$$
$$\downarrow$$
$$[[[[Addr]_{PK_{35}}||100]_{PK_{23}}||11]_{PK_{27}}||01]_{PK_{32}}$$

3) Decoding the path address: When the basestation receives the packet, it knows the sender of the last hop of the packet. In the example in Figure 3, the last hop forwarder being node 32. The basestation also knows the neighbors of node 32. On decoding the packet using the key specific to node 32, the basestation knows the relative address of the node which forwarded the packet to node 32. Looking up the table, the basestation identifies that the packet was received from node 27. Recursively, the basestation can trace the complete path back to the sender. This is represented as follows:

$$[EncryptedAddress]_{PK_{32}} \Longrightarrow [EncryptedAddress_1\ ||\ 01]$$
$$\downarrow$$
$$[EncryptedAddress_1]_{PK_{27}} \Longrightarrow [EncryptedAddress_2\ ||\ 11]$$
$$\downarrow$$
$$[EncryptedAddress_2]_{PK_{23}} \Longrightarrow [EncryptedAddress_3\ ||\ 100]$$
$$\downarrow$$
$$Node35$$

C. Packet Meta-analysis

Let X_i be the set of packets that are generated as a result of a camouflage event. Let R_i be the set of packets that were received by the basestation and L_i is the set of packets that were not received.

$$R_i \subseteq X_i, \quad L_i \subseteq X_i, \quad R_i\ +\ L_i = X_i$$

The basestation maintains records of characteristics for the different attack types for each node. This record consists of a set of parameters which are updated by the basestation whenever a packet is generated for a camouflage event. As a packet is received or not-received, the record for the nodes gets updated accordingly.

The neighbors of each node are classified as either downstream neighbor or upstream neighbor. A neighbor node which helps propagate the packet towards the basestation is a downstream node and the nodes in the neighborhood which are not downstream nodes are upstream nodes. It must be noted that, a node which is farther away from the basestation as compared to the neighbor node can be a downstream node.

We maintain a set of four parameters for each node; packets generated, packets received, packets lost, packets garbled. Packets generated is a parameter which keeps track of the packets generated by the node in response to the camouflage event. Packet received is a parameter of the node which tally's the number of packets successfully received when the node was either the source of the packet or an intermediate node forwarding the packet. Packet lost parameter of a node tracks the packets that were lost and is statistically determined to see what path the packet should have taken to reach the basestation. Using this information, the intermediate nodes are also marked for the packet lost. The packet garbled parameter tracks the number of packets which were successfully received, but cannot be traced back to the source.

1) Packets Received: These are the packets which are received by the basestation in response to a camouflage event. We calculate the hop count and the total time elapsed between the camouflage event and the time of reception of the packet. This is possible because we have the temporal information about the occurrence of the event. The basestation checks if the number of hops and the time of delivery is within the deviance limits for the particular node. If either of the two parameters lie beyond the threshold limits, we do a path analysis.

If the number of hops is within the limits (called Hop-Validity) and if the time elapsed is also within deviance (Time-Validity), we mark the nodes packet received parameter. If either of hop-validity or time-validity fails, we do a path analysis for the packet. The first step in path analysis is to do a traceback to the source. Under certain circumstances like a wormhole attack, we will not successfully traceback to the source. If the traceback is successful, we perform a per-hop analysis for the packet received. In a per-hop analysis, we consider each pair of consecutive intermediate nodes and see if they adhere to the upstream-downstream requirement.

Let P_i be the packet sent by node N_s to report an event E. m is the number of hops and t is the time taken by the packet to reach the basestation. h_e and t_e are normalized expected number of hops and time.

$$Hop\ Validity_i = \frac{m-h_e}{h_e}$$
$$Time\ Validity_i = \frac{t-t_e}{t_e}$$

If either of the two validity fails, we trace the packet back to the source based on the encoded relative addresses. If trace back to the source is successful, for each hop of the packet, let node x_t be the transmitter and node x_r be the receiver of the packet. Check if node x_r is a downstream neighbor of node x_t. If it is not we mark node x_t. If the traceback fails, mark the packet garbled parameter similar to the way the packet lost parameter is marked. This is presented in algorithm 5.1.

Algorithm 5.1 Node Marking

$N \rightarrow all\ nodes, R_i \rightarrow received\ packets$
$L_i \rightarrow lost\ packets, x_n \rightarrow nodes\ on\ path\ of\ packet\ P_n$
Require: $P_i \in R_i$
  if $((HopValidity_i)\ ||\ (TimeValidity_i))\ fails$ **then**
    Initiate Source Traceback
    while $(n_i \neq N_s)\ ||\ (TracebackAdd! = \phi)$ **do**
      $p = NumNeighbors(n_i)$
      $RelAddLen = sizeofbitlength(binary(p))$
      $RelAdd = RelAddLength\ LSB\ of\ TracebackAdd$
      $n_i = Lookup \rightarrow RelAdd(n_i)$
      $TracebackAdd = TracebackAdd - RelAdd$
    end while
    if Source Traceback == SUCCESS **then**
      for all $(Node\ n \in x_n)$ **do**
        if $n \in path\ of\ P_i$ **then**
          if $x_r\ is\ downstream\ neighbor\ of\ x_t$ **then**
            $x_t \rightarrow SuspectNode$
          end if
        end if
      end for
    else
      Intermediate Node Identification
      $(Node\ n \in x_n) \rightarrow Path\ of\ P_i$
      $Mark\ PacketGarbled(n)$
    end if
  end if
Require: $P_i \in L_i$
  Intermediate Node Identification
  for all $(Node\ n \in x_n) \rightarrow Path\ of\ P_i$ **do**
    $Mark\ PacketLost(n)$
  end for
  Marked Node Analysis
  for all $(Node\ n \in N)$ **do**
    if $PktLost(n) \geq 1$ **then**
      $u \in UpstreamNeighbor(n)$
      if $PktLost(n) > \sum PktLost(u)$ **then**
        $n \Rightarrow suspectnode$
      end if
    end if
    if $PktGarbled(n) \geq 1$ **then**
      $u \in UpstreamNeighbor(n)$
      if $PktGarbled(n) > \sum PktGarbled(u)$ **then**
        $n \Rightarrow suspectnode$
      end if
    end if
  end for

We populate the table over a period of time and perform a short term analysis and a long term analysis. Each of these analysis caters to a particular type of attack. The short term analysis is needed to identify wormhole, sybil and sinkhole attacks while a long term analysis is performed to identify a selective forwarding attack. The long term analysis has sufficient information that helps us to differentiate between the sinkhole and selective forwarding attacks. Also, it helps us differentiate between a dead node and a malicious node performing a selective forwarding attack. The reasoning behind this is the fact that to identify a selective forwarding attack, we need to gather information over longer durations as compared to other attack types.

2) Packets Lost: With the knowledge of the occurrence of the events, the basestation expects a set of packets from a group of nodes. If there is malicious activity, there may be loss of packets. For each packet lost the basestation is able to traceback the packet loss to within a subset of nodes. This is depicted in Algorithm 5.1. To pinpoint the node which was responsible for the packet loss from within the subset of nodes, we have each node maintaining a simple but efficient data structure which works on the principle of bloom filter and stores overheard packet transmissions.

Each node maintains multiple counting bloom filters. These

bloom filters are used to mark the overheard packet information during different intervals of time. The basestation can query the neighboring nodes to check if they overheard the packet transmission by the node over a particular interval of time. As for the response to the query, the architecture provides two options. One, the queried node responds back to the basestation with its bloom filter representation and the basestation processes it to decipher the information. In the second case, the node processes the bloom filter data and replies back with only the required information.

The two methods have their own advantages and disadvantages. In the first method where the sensor node returns the entire bloom filter as a response, the basestation might be saved from making repeated queries to the same node about its different neighbors. Another significant advantage of this method is the discreteness with regards to the node under observation. The drawback is the increased packet size. In the second method, the communication cost is less, but the basestation will need to query the node about a specific neighbor and cannot be discrete like in the first method. Hence there is loss of anonymity and it can be qualifying factor in certain military applications.

Bloom Filter implementation: Each node has multiple counting bloom filter. The nodes overhear the packet transmissions of their neighbors and mark information into the counting bloom filter using the relative address of the neighbor. The number of bloom filters maintained by each node will decide the granularity of the information stored. For example, information maintained in a single bloom filter over a period of time t will give us less accurate description of events compared to 10 bloom filters maintained by the node for each of $t/10$ time intervals.

D. Attack Fingerprinting

Different attack types possess some distinct characteristics and based on the detection model, these characteristics can be represented as a blueprint to identify the attack. In this section we model the attacks based on the these characteristics possessed by each attack type.

Sybil attack: The basis of sybil attack depends on two factors- The ease of acquiring different identities and the amount of damage a node can inflict by acquiring the identity. The second factor about the amount of damage is prevalent in systems with peer-managed reputation systems. In our system which is based on the nodes response to the real-world-like camouflage events, a node will not be able to tarnish the reputation of another node since the neighborhood of the nodes is set and the peers are not required to maintain reputation. Consider the scenario in which a group of nodes collaborate with malicious intent, they still will be identified by the outermost ring of malicious node's neighbors. The control of managing the reputation lies with the basestation.

Let it be feasible for the node to assume different identities. Since, the neighborhood for each node is set, it can only assume an identity of a node within the neighborhood. There will be two or more nodes with the same identity in the neighborhood. In this scenario, the node assuming the identity is the attacker and the node whose identity is assumed is the victim. When a packet is received, the attacker can choose to forward the packet or drop the packet with the victims identity. Dropping the packet does not do any damage. Suppose the packet is forwarded by the attacker node posing as the victim. The node has to encrypt the packet with the unique private key of the victim node while including the information about the relative address of the node from which it received the packet. The malicious node can choose to forward the packet after appending garbage or without modifying the packet and the intermediate nodes will not be able to deduce that the packet is corrupted. The basestation on receiving the packet initiates the traceback process and will not be able to trace the path back to the originating node since all the relevant information is not included in the packet. In our methodology, the sybil attack possesses characteristics similar to wormhole attack in which case, the non-verification of the traceback path to the originating node indicates the presence of malicious activity.

Sinkhole attack: This attack can be easily launched by a node by indicating a low cost path availability through it and then dropping the packets. In sinkhole attacks, the node advertises a low cost path to the basestation and may or may not evince itself to be within the neighborhood of a larger number of nodes than it actually is. The basestation knows the location and neighborhood information of all the nodes and can track the loss of packets to within a few hops of this malicious node. This attack type can be identified by analyzing the packets lost and then verified by querying the the bloom filters of neighborhood nodes in the identified region.

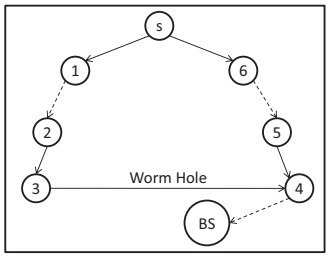


Fig. 4. Wormhole

Wormhole attack: We described earlier about how the intermediate node cannot thwart being identified in case of wormhole attacks. Successfully tracing back to the source under wormhole attack: Consider the scenario in Figure 4 wherein the source node s sends packet to the basestation. The nodes 3 and 4 collaborate to launch a wormhole attack. In this case, the node 4 will fake a relative address of a neighbor before forwarding the packet since it cannot specify that it received the packet from node 3. For the basestation to be able to traceback to the source, it needs to satisfy the following. There needs to exist a valid path from node 4 to source with $n + 1$ number of hops where n is the number of hops from node 3 to the source. Let there exist a path P satisfying this condition. Next, it needs to satisfy the per-hop condition such that relative addresses encoded in the packet from source to node 3 using the private keys of the intermediate nodes matches that of a new path from node 4 to the source increases exponentially with each hop. Also, the effectiveness of a wormhole attack depends on how farther away the traffic is re-routed in the network, thereby increasing the resource consumption in the network. If it needs to satisfy the first condition *i.e.* to maintain the hop count, then the collaborating attacking nodes within the network cannot

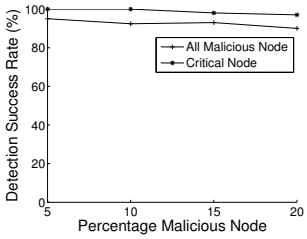


Fig. 5. Sinkhole Detection Rate

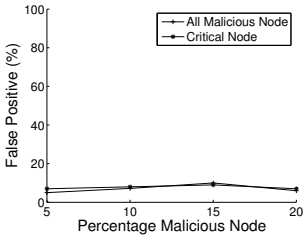


Fig. 6. Sinkhole False Positive

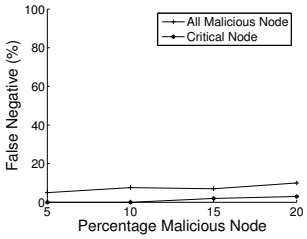


Fig. 7. Sinkhole False Negative

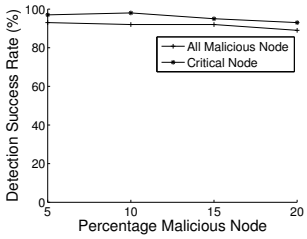


Fig. 8. Selective Forwarding Detection Rate

reroute the packet across large distances. This requirement defeats the purpose of launching a wormhole attack.

The final goal of the basestation is to identify the malicious node irrespective of the type of attack. Identifying the type of attack helps the basestation to garner further information about the colluding accomplices in the attack. In the case of a wormhole attack, there needs to be at least two collaborating nodes or else in the case of Sybil attack, there can be one or more compromised nodes.

VI. Simulation Results

The simulations were performed over an area of 500x500m, having 125 nodes. There were three varying factors in the attacks. The number of malicious (compromised) nodes was varied as 5% 10%, 15%, 20%. The compromised nodes were randomly selected but the critical nodes were given a higher priority to be under attack considering a worst-case scenario with a sophisticated attacker. Secondly, the attack type of the compromised nodes were again randomly assigned. The base station gathered data over multiple runs.

The first set of simulations consisted of all the malicious nodes launching the same type of attack. Figure 5 depicts the detection success, when all the malicious nodes are performing sinkhole attack. Even though when the success rate of identifying the malicious node was not 100%, the subset of malicious node which are critical were identified accurately. There are two reasons for this, the critical nodes have connectivity to the basestation and the camouflage events are generated to cover the critical nodes.

The false positive rate when only considering the critical nodes is slightly higher as compared to the false positive rate when the entire set of malicious nodes was considered. Again, as the number of malicious nodes increased we saw an increase in the false positive rate. This is because we aggressively mark a node as a suspect node to get lesser false negatives, with assurance that the verification discards the false positives as depicted in Figure 6. The false negative rate when considering the critical nodes was much lower compared to the false negative rates of the whole malicious nodes set (Figure 7).

The next set of results comprise the malicious nodes all performing selective forwarding attack. We present the results when the nodes randomly dropped 30% of the packets. It was seen that this drop percentage affected the number of rounds of the camouflage event generator needed to make, to get accurate results. Lower the drop percentage, higher was the number of rounds needed by the mobile-node, but again lower drop percentage means a less severe attack. The results are presented in Figures 8, 9, 10 and are very similar to sinkhole. Unlike in sinkhole, the analysis to detect selective forwarding attack is performed over 5 rounds by the mobile-node. In other words, it was slower to catch a malicious node performing selective forwarding attack.

Figures 11, 12, 13 represents the detection success, false positives and false negatives detected for the network under sybil attack. We assumed that at a point in time, a node can acquire the identity of another node, but does not have access to cryptographic information of the node.

In the next simulation, all randomly selected malicious nodes launch a wormhole attack. A wormhole attack is different from other attacks due to the fact that a single attack will have at least two colluding nodes. In the attack, randomly selected nodes collude with each other and launch the attack by directing the traffic to the other side of the network. The results for the wormhole attack is presented in Figures 14, 15, 16. Under the circumstance when just one of the node is directing traffic to another part of the network, the model had difficulty identifying the other colluding node since it was only directing traffic to the basestation and was not doing anything malicious per se. This is depicted in Figure 17.

The next set of simulations involved malicious nodes having equal distribution of all the four attack types. The results for the same are shown in Figures 18, 19, 20. In short term analysis, *i.e.* when analyzing each round of camouflage event packets, it was difficult to differentiate the sinkhole attack from selective forwarding. Analyzing the camouflage packets over multiple rounds produced sufficient information to demarcate the same. The results provided in Figures 18, 19, 20 are for five rounds of camouflage events.

The performance of CENDA is compared with the results from CHEMAS by Xiao *et al.* [11] and with intruder detection algorithm by Ngai *et al.* [9]. The simulation results indicate that the performance is at par with the two methods while having the advantage of being able to detect and differentiate multiple attack types at the same time. The verification of the packet transmissions from the neighbor node helps in drastically reducing false positives in the case sinkhole and selective forwarding attacks. Additionally, we can prevent region segregation malicious attacks. With regards to conserving energy, based on the requirement or attack type anticipated, the verification part of the system can be turned on/off. As a by product, the system is able to detect any dead nodes due to energy exhaustion or due to environmental conditions.

VII. Conclusions

CENDA is a proactive architecture to detect malicious nodes in the sensor network. It can be used as a compliment to an existing system and is controlled by the basestation. We use a mobile-node based camouflage event generator scheme

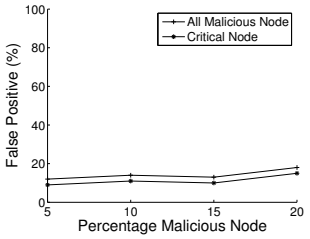


Fig. 9. Selective Forwarding False Positive

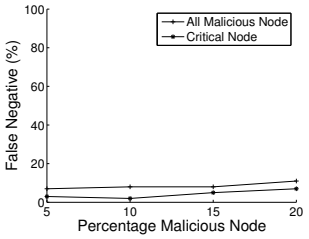


Fig. 10. Selective Forwarding False Negative

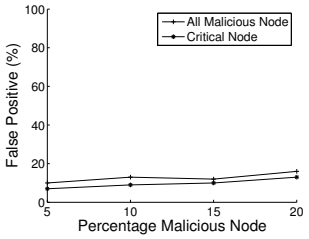


Fig. 11. Sybil False Positive

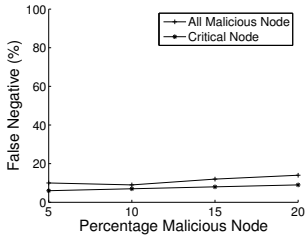


Fig. 12. Sybil False Negative

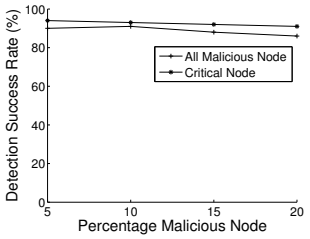


Fig. 13. Sybil Detection Rate

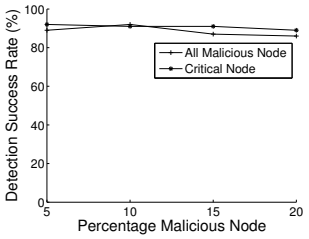


Fig. 14. Wormhole Detection Rate

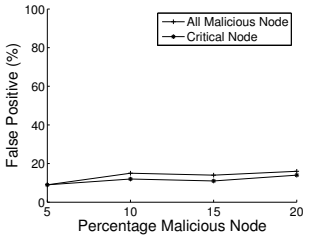


Fig. 15. Wormhole False Positive

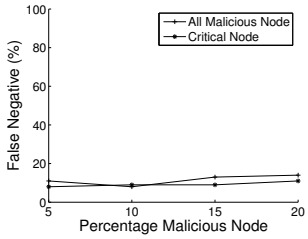


Fig. 16. Wormhole False Negative

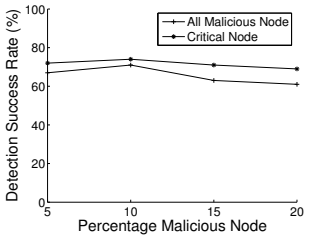


Fig. 17. Wormhole Detection 1 of 2 Nodes

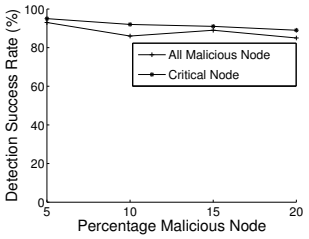


Fig. 18. All Attacks Detection Rate

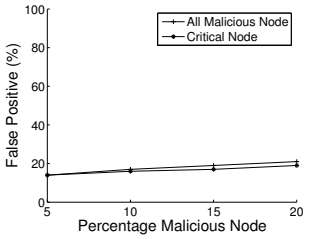


Fig. 19. All Attacks False Positive

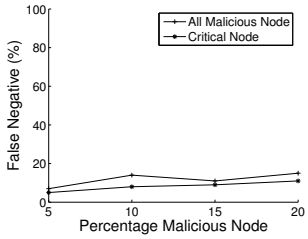


Fig. 20. All Attacks False Negative

to overcome the problem of region segregation by malicious nodes trying to prevent event reporting. This is important in case of sporadic events wherein the basestation cannot differentiate between non-occurrence and non-report of events. The camouflage-event based malicious node detection system uses a bloom-filter based verification procedure before labeling a node malicious. We provide a node-classification scheme based on the role/importance of the node in the network and present a light weight path marking system using relative-addresses to indicate the path traversed by the packet to reach the basestation. The effectiveness of the system is examined using simulations and the results demonstrate this. Compared to an IDS system, we can not only detect an intrusion but also identify the malicious nodes. Additionally, compared to existing systems, which identify attacks of single type, we can differentiate attack of four types and in future this can be further enhanced to identify other attack types as well.

ACKNOWLEDGEMENT

The authors would like to thank Guoliang Xing for his valuable comments to help shape the work.

REFERENCES

[1] C. Karlof, N. Sastry, and D. Wagner, "Tinysec: a link layer security architecture for wireless sensor networks," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*. New York, USA: ACM, 2004, pp. 162–175.

[2] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "Spins: security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521–534, 2002.

[3] R. Watro, D. Kong, S.-f. Cuti, C. Gardiner, C. Lynn, and P. Kruus, "Tinypk: securing sensor networks with public key technology," in *SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*. New York, USA: ACM, 2004, pp. 59–64.

[4] A.-S. K. Pathan and C. S. Hong, "Serp: secure energy-efficient routing protocol for densely deployed wireless sensor networks," *Annales des Télécommunications*, vol. 63, no. 9-10, pp. 529–541, 2008.

[5] A. Pirzada and C. Mcdonald, "Circumventing sinkholes and wormholes in ad-hoc wireless networks," in *International Workshop on Wireless Ad-hoc Networks (IWWAN 2005)*, London, England, 2005.

[6] R. Roman, M. C. Fernandez-Gago, and J. Lopez, "Featuring trust and reputation management systems for constrained hardware devices," in *Proceedings of the 1st international conference on Autonomic computing and communication systems*, ICST, Brussels, Belgium, 2007.

[7] H. Chen, H. Wu, X. Zhou, and C. Gao, "Reputation-based trust in wireless sensor networks," in *MUE '07: Proceedings of the 2007 International Conference on Multimedia and Ubiquitous Engineering*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 603–607.

[8] S. Ganeriwal and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," in *Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, NY, USA, 2004.

[9] E. C. H. Ngai, J. Liu, and M. R. Lyu, "An efficient intruder detection algorithm against sinkhole attacks in wireless sensor networks," *Computer Communication*, vol. 30, no. 11-12, pp. 2353–2364, 2007.

[10] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[11] B. Xiao, B. Yu, and C. Gao, "Chemas: Identify suspect nodes in selective forwarding attacks," *Journal of Parallel and Distributed Computing*, vol. 67, no. 11, pp. 1218 – 1230, 2007.