2015 IEEE 39th Annual International Computers, Software & Applications Conference

Design and Analysis of a Method for Synoptic Level Network Intrusion Detection

Deanna T. Hlavacek
Department of Electrical and Computer Engineering
Iowa State University
Ames, Iowa 50011-3060
e-mail: dhlavacek@comcast.net

J. Morris Chang
Department of Electrical and Computer Engineering
Iowa State University
Ames, Iowa 50011-3060
e-mail: morris@iastate.edu

***Abstract*—Current system administrators are missing intrusion alerts hidden by large numbers of false positives. We propose an intrusion detection tool that effectively uses select data to provide a picture of "network health". Our hypothesis is that by utilizing the data available at the node and network levels we can create a synoptic picture of the network providing indications of many intrusions or other network issues. Our major contribution is to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues. Our first contribution in this vein is to present a method based on utilizing the number of packets sent, number of packets received, node reliability, route reliability, and entropy to develop a synoptic picture of the network health in the presence of a sinkhole.**

I. Introduction

Wireless ad hoc networks are self-organizing and self-configuring infrastructure-less networks of nodes which are connected by wireless links such as 802.11/WiFi products, wireless sensor networks (WSNs), and the new cognitive radio network. With the growth in popularity of these technologies there is a growing demand for intrusion detection systems (IDS) that can operate with network node cooperation. Current research on intrusion detection systems for wireless ad hoc systems focus mainly on anomaly or signature detection. These methods are subject to high false positive rates. With limited time and resources, many true positives are lost in the overload of the combined true and false alerts. Perhaps the answer is not more data, but the better use of existing data.

Our motivation for research related to intrusion detection arises from the current lack of comprehensive research into methods of analysis of selective information in an effort to construct a big picture of network security and integrity, termed as "network health". Our major contribution is to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues. Our first contribution in this vein is to present a method based on utilizing packet delivery ratio (PDR), node reliability, route reliability, and entropy to develop a synoptic picture of the network health in the presence of a simple sinkhole. In order to provide a proof of concept we utilize a simple grid based stationary network similar to a wireless sensor network. With this simplified first example we intend to show that, although the concept of intrusion detection is not revolutionary, the method in which we analyze the data for clues about network intrusion and performance is innovative, and can be a valuable addition to the intrusion detection "toolbox".

In this paper we will first take a look at current research related to intrusion detection systems designed for wireless ad hoc networks. We then provide a description of our methodology in network analysis for sinkhole detection, and our results based on simulation data. Last we will conclude with a synopsis of the process and the impact of our experimental results. The sections of the paper are organized as follows: Section II-B describes current research into intrusion detection systems and sinkhole detection in wireless ad hoc networks; Section III presents the methodology of identifying a sinkhole based on the PDR, node reliability, and system entropy; Section IV presents our results with simulation and data analysis; Section V provides comparison of the simulation results to other sinkhole identification methods; and Section VI is the conclusion.

II. Related Work in Wireless Ad Hoc Networks

Wireless ad hoc networks have no router or access point providing infrastructure to the network. Each node provides routing services, via routing protocols, by forwarding packets to their neighbors. All nodes in an ad hoc network have equal status in the network, and can associate with any network device within range. There are three main routing protocols for ad hoc networks: Optimized Link State Routing (OLSR), Dynamic Source Routing (DSR), and Ad Hoc On Demand Routing (AODV). OLSR is a proactive, or table-driven, protocol. DSR and AODV are both on-demand, reactive protocols in which the nodes maintain routing tables. AODV's routing tables are refreshed according to a timer. We have chosen to use the AODV routing scheme for our first demonstration.

A. Intrusion Detection Systems

Conventional intrusion detection systems (IDS) are based on misuse detection, anomaly detection, or deviation from specifications. Misbehavior/misuse detection refers to identifying an attack by comparing collected information against a predefined list of "signatures" of known attacks. Anomaly detection is a close opposite to misuse detection. With anomaly detection, rather than storing a list of the signatures of known attacks, the system stores patterns of "normal" behavior for comparison to the current behavior. The third technique, specification comparison, also compares the current behavior to a stored behavior profile. However, the comparison is against manually defined specifications, rather than machine learning and training techniques.

Recently, most of the intrusion detection system research for wireless ad hoc systems has focused upon the detection of anomalous behavior patterns. In the paper [13] a new detection scheme called AODVSTAT is presented. The method is similar to other watchdog schemes in that the nodes watch the packet events and the metadata in the packets for anomalies in the protocol. Deviations from the protocol are considered state changes, and trigger an alarming event. In an effort to lower false positive rates using packet features as the basis for anomaly detection, the authors of [11] conduct careful feature selection from the available set of packet features for their method. Similarly, [9] uses the entropy of packet features to detect deviations, and provides a ranking of alerts in an attempt to lower the false positive rate. The authors of [12] base their system solely upon the packet sequence number mismatch with the expected packet sequence number. A confidence level for each node is calculated based on the number of interactions with its neighbors. Rather than packet features, the authors of [14] base their detection on anomalies

0730-3157/15 $31.00 © 2015 IEEE
DOI 10.1109/COMPSAC.2015.17

in traffic patterns, comparing the current traffic pattern to a learned traffic pattern. Each of these techniques requires either knowledge stored in memory of the "normal" pattern of behavior, or some type of training before deployment. Attacks that do not register against the learned normal profile can not be detected by these systems. These methods are also prone to false positives, since hiccups in the network can cause the systems to react with identification of pattern anomaly.

The authors of the papers [1], [6] both present hybrid detection schemes. The paper [1] combines anomaly and misuse detection schemes in order to lower the false positive rate generally seen with anomaly detection, and raise the low detection rate ascribed to misuse detection. The authors of [6] chose to combine the anomaly and specification based schemes, and uses a reputation based system in an attempt to lower the false positive rate. However, this method adds much packet overhead as the nodes in the neighborhoods vote.

In comparison to these methods of intrusion detection, our intrusion detection method is based not on stored patterns, signatures, or rules, but the effect upon the node, route, and network function. Our system requires no training or pre-placed data concerning the expected network protocol or traffic pattern. Therefore, new or craftily tweaked older intrusion methods, such as a stealthy sinkhole that follows network protocols while selectively dropping packets, will still be identified, as long as the effect of the disruption surpasses the established threshold. We have no watchdogs observing the network, and so we are not plagued by the high false positive rates endemic to these methods. Similarly, we are not doing signature comparison, so we are not plagued by the inability (and associated low accuracy rate) to recognize new attack signatures. Our system instead is monitoring the state of the network, and reacts when the state of the whole, or portions of, the network move out of alignment. As in holistic medicine, we do not only pay attention to the symptom; we analyze the symptoms to explore the network function and find the root cause of the malfunction.

B. Sinkhole Detection

We consider sinkholes as Byzantines in the network. Byzantine behavior is displayed by any action of a member node that negatively affects the routing service in the network. Many such attacks, such as eavesdropping or packet modification, can be prevented by traditional authentication, integrity, and encryption mechanisms. The malicious actions of a Byzantine sinkhole may be more complex, such as modifying the hop count, sequence number, or list of nodes in a path, in order to make itself more attractive as an entry to an ideal route. According to the paper [2], attacks using these tactics can also be prevented with more sophisticated authentication and integrity techniques. We therefore consider the stealthy sinkhole that drops data packets, entirely or selectively, while participating in the routing protocol.

The authors of [2] present a method of identifying a sinkhole with link weights and probes. However, this method is part of a newly proposed routing protocol that includes double flooding during route discovery and the sending of probes to all network nodes for attack discovery. These steps create additional network overhead. Additionally, in this routing protocol, the sinkhole will only be discovered if it is acting maliciously during the probing phase. Finally, this work provides no insight as to how to identify a sinkhole in the accepted ad hoc routing protocols Ad Hoc On Demand Distance Vector Routing (AODV), Dynamic Source Routing (DSR), or Optimized Link State Routing (OLSR).

In the paper [10] the authors present a packet drop attack detection method in which the neighbors adjacent to a communications route monitor the actions of the en route nodes. If a particular node does not forward a specific number of packets in a certain time period, an alert proclaiming a malicious node is started. The authors make a distinction between greyholes, which drop only a portion of the packets, and blackholes, which drop all received packets. Analysis by the authors indicates that only blackholes, and gray holes in the same vicinity, can be identified. If no blackhole exists, the greyholes will not be identified. Similar to this method is the watchdog method presented by the authors of [7]. Again, collaborative nodes observe the actions of the nodes en route and use a protocol to determine when an alarm should be sounded. Unfortunately the reliability and effectiveness of this method is difficult to determine since the authors have not yet determined the false positive/false negative rates for the protocol.

Several papers rely upon the sinkhole bucking the routing protocol by changing the sequence numbers. The sinkhole identification schemes described in the papers [3], [5], [8] are therefore not effective in identifying a stealthy sinkhole that does not change the packet sequence number. The authors of [3] additionally use the previous image ratio to identify the sinkhole. In the previous image ratio method the received routing packets are compared to other stored routing packet images. However, this method relies upon the sinkhole having forged the route records in their route request packets. Therefore, if a stealthy sinkhole has not forged the route records, the sinkhole will not be identified by this method.

A third indicator of a sinkhole identified and used by the authors of [3], [8] is the route add ratio. The route add ratio is the number of routes that traverse a particular node divided by the total number of routes added to the node's routing table. Unfortunately, [8] only mentions the idea of the route add ratio, but does not explain how the ratio is used. In [3] a network node is specified to keep a counter for each node in the network, and increment the counter when a route passing through the node is added to the cache. This presents the issue of one node storing data for all of the nodes, and additional messages created and sent to the assigned node when any network node adds a route to its cache. The data related to this study did not provide a method to determine the message overhead related to this technique.

In comparison to these studies, our method does not rely solely upon the sinkhole cheating on the routing protocol. Therefore, a stealthy node will still be identified by its effect on the network, rather than missed due to it not sending signals through the routing protocol. Also, even though there are additional messages included with our sinkhole identification scheme, no additional messages are required before the possibility of a sinkhole is discovered. The overhead of messages m related to sinkhole identification is related to the number of hops from the detecting node to the sinkhole and is very small. We analyze the number of messages required in Section V-C.

III. Methodology

The methodology of the sinkhole identification scheme consists of two parts. First, the detection phase, in which one or more nodes are alerted that there is a possible sinkhole in the network. Individual nodes calculate their neighbors' reliability values, and are alerted when a reliability crosses a threshold. The second phase is the sinkhole identification phase and involves querying specific network nodes for data they have about *their* neighbors. We make the assumption that each node is aware of its immediate neighbors. However, the nodes may not immediately be aware of the position of its neighbors relative to itself or each other.

A. Detection Process

In the neighbor reliability method of determining the health of the network, each node counts the number of packets sent and received

Algorithm 1 Calculations of Neighbor Reliability

```
//Done for each neighbor node (Y) around the starting node A
//routeRel: route reliability
//neighRel: neighbor reliability
1 : for each neighNode(Y)
//Calculate route reliability for routes X through neighNode Y
2 :    for each route(X) (Z nodes along route)
3 :       for each node(Z) on route X
4 :          routeRel(X) = routeRel(X) * pdr(Z);
5 :          saveRouteRel(XZ) = routeRel(X);
6 :          get next node pdr(Z) on route(X);
7 :       get next route(X);
//Calculate neighbor reliability using all routeRel(X)
8 :    for each routeRel(X)
9 :       neighRel(Y) = neighRel(Y) +
                        (routeRel(X) * [log2(1/routeRel(X))]);
10:       saveNeighRel(X) = neighRel(Y);
11:       get next routeRel(X);
12: get next neighNode(Y);
```

along each route. The nodes calculate the packet delivery ratio for the stored routes by relating the number of packets received along the route to the number of packets sent along the route. This PDR is also referred to as the "route reliability", and represents the cumulative reliability for each node along a route. From this data, the nodes each determine their "neighbor reliability" by calculating the entropy for all known routes through the neighbor node. Neighbor reliability refers to a single node's perception of the probability of a packet following any route through a single neighbor to successfully reach the intended destination. The Shannon entropy equation, used to estimate the diversity of the system, is applied. The formula follows, where p(x) is the route reliability:

$$H(x) = -\sum p(x) log_2 p(x) = \sum p(x) log_2 (1/p(x)) \qquad (1)$$

Algorithm 1 describes the process to obtain the neighbor reliability values. Note that in simulation we use the probability that a packet will reach its destination when routed through an individual node for the PDR of node Z. This was substituted for the true PDR of a route that would be known in a live network.

Using this method, assuming a stationary MANET grid of nodes with one node per grid space, each node in the grid will have up to eight different neighbor reliability values, each value from the perspective of one of its (up to eight) neighbors. Additionally, each node will have up to eight neighbors for which it has determined neighbor reliability values, comprising the neighbor reliability set. Using its neighbor reliability set, each node calculates the standard deviation of the set. A lower boundary is calculated by subtracting the standard deviation from the mean of the set; the boundary acts as a threshold. The use of the standard deviation was determined experimentally; one standard deviation provided a proper boundary to determine if the neighbor reliability was low enough to indicate the possibility of a sinkhole when compared to the neighbor set. The lower boundary in a live network will need to be determined based upon the particular network. This process provides the node the ability to observe the current state of the network surrounding it, and helps identify anomalies based on the current network state. If the neighbor reliability of any neighbor crosses the threshold, the node is alerted that there may be an attacker in the network. Note that until this point, all calculations and decisions are made upon data collected by the node without additional messages or queries to the network or neighbors. Therefore, unless an alert is signaled, there is no impact upon the network throughput for this method.

The example in Figure 1 shows several nodes with reliability values

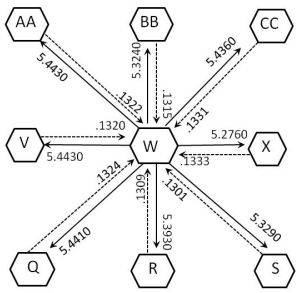

Figure 1. Neighbor Reliability Plot (values * 10^{-6})

applied to each node *by its neighbors*. The values in the figure are charted according to compass coordinates from the perspective of an individual node. For example, Node W applies the values of 5.443 to node AA, 5.324 to node BB, 5.436 to node CC, 5.276 to node X, 5.443 to node V, 5.393 to node R, 5.441 to node Q, and 5.329 to node S. These values comprise node W's neighbor reliability set. Conversely, node W is applied the values of 0.1322 as perceived by node AA, 0.1315 by node BB, 0.1331 by node C, 0.1333 by node X, et cetera.

B. A Proof

We consider a simple grid network of nodes with a sinkhole in the center (Figure 2). Let a be the percentage of successful packet delivery for regular nodes; it is assumed regular nodes drop few packets. The sinkhole drops a large number of received packets; let the percentage of successful packet delivery for the sinkhole be b. The network uses AODV routing, utilizing the shortest route. We make the following assumptions:

1) Each node is aware of all of its eight neighbors.
2) No route will pass through more than one of the source node's neighbors, and no more than two neighbors of any node along the route.
3) All routes are of three hops.
4) Percentage $a >> b$.

A source node (src1) located next to the sinkhole will have seven neighbors with routes that do not traverse the sinkhole, and therefore experience a packet delivery percentage of a. The last neighbor will have all packets routed through the sinkhole, experiencing a packet delivery percentage of b. The average percentage of packets experienced by this node will therefore be:

$$PDR_{src1} = (7a+b)/8. \qquad (2)$$

A source node (src2) two hops from the sinkhole will have more routing choices that do not include the sinkhole. In the described network, five of the neighbors will have no routes through the sinkhole. Of the three neighbors left, two will each have four of their nineteen routes traversing the sinkhole (see Figure 2). Let this be represented by $PDR_{31/33}$ since the description applies to both of these neighbors in the example. One neighbor will have three of its thirteen routes passing through the sinkhole (Figure 3). Let this be represented by PDR_{32}. The PDR experienced by src2 is therefore represented by the following:

$$PDR_{src2} = 5a + PDR_{31/33} + PDR_{32} \qquad (3)$$

where

$$PDR_{31/33} = [2(15/19)a + 2(4/19)b]/8 \qquad (4)$$

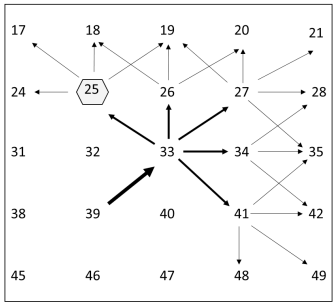

Figure 2. Route Example from SRC2 (Node 33)

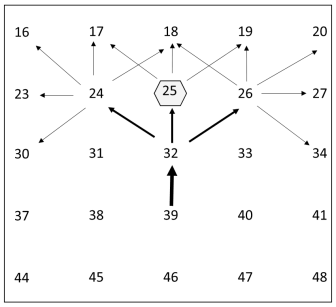

Figure 3. Route Example from SRC2 (Node 32)

and

$$PDR_{32} = [(10/13)a + (3/13)b]/8 \quad (5)$$

or

$$PDR_{src2} = [7.53a + .65b]/8. \quad (6)$$

Likewise, a source node (src3) three hops from the sinkhole will have more routing choices that do not include the sinkhole. In this scenario seven of the neighbors do not have any routes through the sinkhole. One neighbor has three of its thirteen routes traversing the sinkhole. We get the following equation:

$$PDR_{src3} = [7a + (10/13)a + (3/13)b]/8 \quad (7)$$

or

$$PDR_{src3} = [7.77a + .23b]/8. \quad (8)$$

Since $a >> b$, we remove the terms with b from the equations, resulting in the relationship $PDR_{src1} < PDR_{src2} < PDR_{src3}$ since

$$7a < 7.35a < 7.77a \quad (9)$$

C. Sinkhole Identification Process

The identification process works similar to water in a reservoir. When the water reaches a higher point in the reservoir, it stops moving forward and splashes back. In our identification process, the query messages will move towards the lowest reliability point and stop at the higher value nodes on the other side of the lowest point. Once the higher nodes are reached, the "splash" will be a broadcast message flooded through the network reporting the identity of the sinkhole.

In the identification process, we assume that each node is aware of its immediate neighbors, although not necessarily their relative positions. We call this set of neighbors "ring one" as related to the alerted node. When a node determines that a threshold has been crossed, it broadcasts a query to its immediate neighbors (ring one) for their neighbor lists. The query includes the alerted node's assignment of L_1, which is the node with the lowest neighbor reliability from the perspective of the alerted node. Only the neighbors with neighbor lists that include L_1 broadcast a reply to the query. Node L_1 was able to receive the information sent by its own immediate neighbors in response to the query made by the alerted node. Node L_1 assigns L_2 to its neighbor with the lowest reliability value (L_2 will most likely, but not necessarily, be in ring two), and sends this assignment along with a broadcast query for neighbor lists. Only the neighbors with L_2 in their neighbor lists that have not already broadcast their neighbor lists during this identification process round reply to the query. (Algorithm 2)

This process continues until we have reached the area around the sinkhole, ring n. The nodes queried in ring n will include the sinkhole and some of its immediate neighbors. At this point the sinkhole may or may not participate in the process. If it does participate, the sinkhole will continue the process by identifying L_{n-1}. L_{n-1} will then identify the sinkhole as its neighbor with the lowest reliability value. The sinkhole may not acknowledge, nor realize, it is the sinkhole. Therefore, the sinkhole (call the sinkhole S) assigns L_{LOW} (call this node X) to one of its neighbors. Node X compares its neighbor reliability values and re-identifies the sinkhole (S) as its neighbor with the lowest neighbor reliability. To attain confirmation, X broadcasts a query to its neighbors for a "vote" as to which of the two nodes they identify as the node with the lowest value. Only nodes with both nodes S and X as neighbors reply. The node requesting the vote will flood the network with the identification message naming the sinkhole. If the sinkhole does not participate, the node (L_{n-1}) that sent the message to the sinkhole and its neighbors will note that there have been no messages from L_n in time T. At this point, L_{n-1} will resend a message to the sinkhole and neighbors common to the requesting node and the non-responding (possible) sinkhole, requesting the reliability values for all of their neighbors. If the neighbor lists received do not include at least three neighbors in common, additional queries will be sent by the neighbors of L_n to their neighbors. The additional information received will be provided to L_{n-1}. Analyzed together, this information will confirm that the node with the lowest reliability in ring n is indeed the sinkhole, and L_{n-1} will broadcast the message identifying the sinkhole to the network.

The action taken by the network upon identification of the attacking node is dependent upon the system protocol, and is beyond the scope of this paper. However, there are basically two types of actions: isolation of the node, or incentivation for proper network behavior. Isolation means the suspect node may be ignored by the network when it advertises access to routes to any destination, and so ostracized from the network. Incentivation means allowing the suspect node to send its own data only at the rate that it is providing to the network, and this allowed rate increases as the misbehaving node decreases the number of dropped packets and increases its packet delivery ratio. It must be recalled that although we have referred to the suspect node with low reliability an attacker, it may be a selfish node, or just a malfunctioning node. This method of identifying the unreliable node does not determine intent.

IV. Simulation

To obtain simulated network data we chose OPNET Modeler 17.5, a commercially available tool set used by the communications industry for modeling, simulation, and analysis of communications networks and applications. The initial network topology is a six by five grid mobile ad-hoc network (MANET). The size of our network was chosen based on a study of the effects of insider attacks by the

Algorithm 2 Identification of Sinkhole

1 : initialize row, col, x to 0; AlertedNode is L_0 at [0,0]
3 : L_x identifies L_{x+1} as neighbor with the lowest neighbor value
4 : L_x broadcasts query for neighbor lists along with L_{x+1} identification
5 : x=x+1
// if L_x participates in the discovery process it will automatically query neighbors
6 : **while** sinkhole not found
7 : L_x identifies L_{x+1} as neighbor with the lowest neighbor value
8 : **if** L_{x+1} == L_{x-1} OR L_{x+1} == L_{x-2} AND if other shared neighbors that have not yet been polled exist
9 : immediate neighbors (not L_{x-2}, L_{x+1}, or L_{x-1}) vote to determine which of the compared nodes has the lowest value
10: node that is not determined lowest value alerts network of identity of sinkhole
11: **else if** y > 2 AND L_{x+1} == L_{x-y}
12: L_{x-1} removes L_x neighbor value from neighbor list
13: x = x-1 //this is a loop with no positive sinkhole determination
14: **else if** L_x queries neighbors for THEIR neighbor list
15: L_{x+1} receives neighbor lists from immediate neighbors and determines relative placement of immediate neighbors
16: **else if** L_{x-1} does not hear L_x query neighbors in time T
17: L_{x-1} queries immediate neighbors shared with L_x (L_{shared_1} and L_{shared_2}) for their neighbor lists and neighbor values
18: L_{x-1} determines relative locations of neighbors shared between L_{shared_1} and L_{shared_2}
19: **if** neighbors shared include only L_x and L_{x-1}
20: L_{shared_1} and L_{shared_2} send queries for neighbor lists and reliabilities
21: neighbors with lists that include L_x respond to query
22: L_{shared_1} and L_{shared_2} provide query information to L_{x-1}
23: L_{x-1} uses neighbor locations and neighbor values to determine if L_x has lowest local neighbor value
24: **if** L_x has lowest neighbor value as reported by surrounding immediate neighbors
25: L_{x-1} alerts network L_x is a sinkhole
26: **else**
27: L_{x-1} removes L_x neighbor value from neighbor list
28: x = x-1
29: x=x+1
30: **end while**

Node Parameters	Value
Number of Nodes	30
Node Placement	6 by 5 grid
Simulation Duration	1500 seconds
Routing Protocol	AODV
Type of Stations	MANET
Node Speed	0 kts
Transmit Power	.0001 w
Packet Reception Power Threshold	-95 dBm
Buffer Size (Member Nodes)	256000
Buffer Size (sinkhole)	415
Route Length	5 nodes, 4 hops

Table I
NETWORK PARAMETERS

authors of [4]. According to the study, thirty nodes is the ideal size of network for a sinkhole to operate effectively.

For the initial study the nodes are all stationary and the parameters for the member nodes are uniform. Traffic supplied by the OPNET MANET model is used. Each node is a traffic generator. No specialized traffic is added, and no additional noise is added to the network. Ad hoc on demand (AODV) routing is chosen as the routing protocol. Route length for the identification method is limited to five nodes, with four hops. This limitation was imposed to provide a simple initial standard for comparing route reliability results. The routes used for the calculations are not all inclusive. Therefore, not all neighbors may play a part in determining the calculated neighbor reliability value. Parameters for the member nodes are shown in Table I.

The buffer size for the member nodes was left at the OPNET MANET default of 256000 packets. The buffer size for the attacker was lowered to 415 packets to simulate a sinkhole. This buffer size was chosen because it allowed packets to be dropped without stopping all traffic routing through the attacking node. The number of dropped packets at this setting impacted significantly the amount of data traffic sent by the attacking node. Normal nodes sent an average of 903.5 packets over the 1500 second simulation period, as opposed to the 17.3 packets sent by the sinkhole. Note that the 1500 second simulation period and the 15 second intervals were arbitrary. Since the sinkhole effect is recorded almost immediately and the effect is nearly constant over the entire period using the simulation parameters described, the interval parameters for recording the data can be optimized. Subsequent tests showed that the sinkhole could be detected in the first fifteen seconds of the commencement of network traffic. Data collected per node included:

1) At each node, the number of packets received per second in fifteen second intervals.
2) At each node, the number of packets sent per second in fifteen second intervals.

A. Discovery and Identification Processes Using Network Simulation Results

In this context, the packet sent ratio (PSR) is the ratio of the number of data packets sent to the number of data packets received at a particular node. To find the neighbor reliability in our simulation, we first calculate the route reliability by multiplying the PSRs for each node along a route, starting at the first hop (as opposed to the sending node). Next we calculate the entropy of all known routes through the neighbor node. Plotting the neighbor reliability values on a plot of the network shows that the lower values tend to be centered around the sinkhole.

For an example, we assume node G is the first node to identify a possible attacker. From the reliability calculations (Table II), node G identifies neighbors H and M as having entropy values below the threshold value for node G. Node G currently has a view of the network that includes its immediate neighbors, although node G may

Starting Node: G	Cumulative Reliability: 0.000258769			Average Reliability: 4.31282E-05		
Reliability Neighbor List	A	B	C	H	L	M
Reliability Value List	3.05418E-06	2.95563E-06	3.05779E-06	2.21184E-06	2.62348e-06	2.2152E-06
Entropy Per Neighbor List	5.59549E-05	5.42868E-05	5.60144E-05	4.07334E-05	4.82379e-05	4.07903E-05

Table II
NODE G'S NEIGHBOR RELIABILITY

not know the relative locations of the neighbor nodes at this time. Node G identifies node H as the neighbor with the lowest reliability value. Node G broadcasts this assignment and queries its neighbors for their neighbor lists.

Only G's neighbors that share H as a neighbor broadcast their neighbor list; therefore, nodes L, M, C, and B broadcast their lists. Upon receipt of the data, node H places the nodes on the grid relative to itself (4). Node H reports it's neighbor with the lowest neighbor reliability value to node M, and queries its own neighbors for their neighbor lists. Again, only the neighbors that share node M as a neighbor broadcast their neighbor lists. From this information, node M is able to place itself and its immediate neighbors on the grid. Node R is assigned as having the lowest neighbor reliability value by node M. Neighbors that share node R as a neighbor make their neighbor list reports. Node R identifies node W as the neighbor with the lowest reliability value, continues the process, and node W identifies node X. Node X then identifies node W as the neighbor with the lowest reliability. Since nodes W and X identified each other, shared neighbor nodes "vote" for the neighbor with the lowest value. Since W has participated in the process, and it has been identified as the neighbor with the lowest reliability in the local area, node W (or, if needed, node X) announces to the network W is the sinkhole.

Node H Neighbors	L	N	M
Reliability Values	4.73898	5.32743	3.51034
Node M Neighbors	Q	R	S
Reliability Values	5.56492	1.90244	5.39725
Node R Neighbors	V	W	X
Reliability Values	5.29084	.130932	5.31013
Node W Neighbors	X	BB	CC
Reliability Values	5.27565	5.32431	5.43592
Node X Neighbors	W	BB	DD
Reliability Values	.133259	5.23300	5.29058

Table III
PARTIAL TABLE OF NEIGHBOR RELIABILITY VALUES ($*10^{-5}$)

We borrow a technique from weather forecasting to analyze the plot by drawing isopleths for the reader to indicate the levels of entropy in the network. This provides a human-readable synoptic view of the network at the current time. As can be noted in Figure 4, the sinkhole has been identified as the lowest area of reliability in the network. Even though the immediate neighbors to the sinkhole node are not dropping packets, their proximity to the sinkhole affects their reliability value because of the number of routes they have stored for use in their routing table that traverse the sinkhole. Nodes farther away from the sinkhole have their reliability values less affected by the attacker because they tend to have fewer routes traversing the sinkhole.

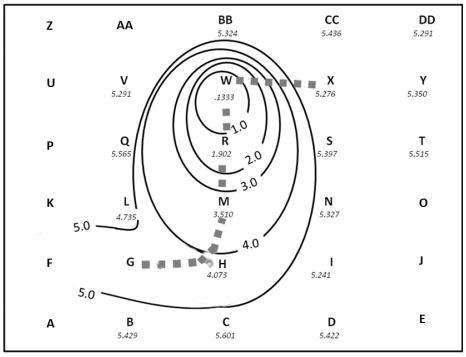

Figure 4. Synoptic analysis of neighbor reliability

B. Data Analysis - One Sinkhole

Analysis of the data set shows that every node except node E resulted in an alert due to at least one of the neighbor nodes having entropy less than the one sigma lower boundary threshold. It must be noted that node E did not have any routes traversing node W (the sinkhole) in its routing table. Every other node had at least one route traversing node W. Also, starting the identification process from every node (except E) found W to be the sinkhole.

There are cases in which the identification process may not be initiated and the sinkhole not found. However, as the network matures and nodes store routes with the sinkhole traversed, the identification process will be initiated. The special cases are listed below:

1) If a node exists with no routes stored that traverse the sinkhole, the sinkhole will not be detected by this particular node.
2) If a node exists with all routes traversing only one neighbor there will be no detection alert by this node. This is due to the method of calculating the threshold.
3) If a node exists with with all routes traversing only two neighbors the detection process will not be started. This is due to the method of calculating the threshold.

C. Additional Cases Investigated

In the case analyzed above we showed that one attacker acting as a sinkhole can be identified in the stationary grid network of thirty nodes. Since the sinkhole was alerted upon by all but the sinkhole itself and one additional node, the chance of sinkhole discovery was 93% (based on the number of nodes that are alerted to a sinkhole in the network by the detection process divided by the number of nodes in the network - in this case, 28/30 = 93%). This is because the effect of the sinkhole extends through the network for routes at least five nodes long. The following cases investigate whether more than one sinkhole can be identified in the network of thirty nodes.

1) No Sinkholes in the Network: In the case of no sinkholes in the network, we would like to have no alerts, and to find no suspects. However, since no network is homogenous (including the OPNET MANET network set up as described), it is likely that there will be at least one node in a neighborhood of nearly homogenous

transmitting neighbors that will be falsely identified as a sinkhole. Using the algorithm described based solely on one standard deviation as the threshold, in a network with no intentional sinkholes placed, there were twenty-six nodes that were alerted to the possibility of a sinkhole. Four false positives were identified. This result portends the possible result that after identified true positives (sinkholes) in a network, the process may then find one or more false positives.

2) Two Adjacent Sinkholes in the Network: In this case, we have included two adjacent sinkholes operating in the same area of the network (nodes V, W). Experimental results show that all but one node was alerted to the possibility of a sinkhole in the network, and nine nodes have two neighbor reliability values that indicate the presence of one or more sinkholes. Since the sinkholes are adjacent, the identification process first finds the sinkhole with the lowest value. After the identified sinkhole is removed from the network, as well as all routes in route lists incorporating the sinkhole, the remaining nodes automatically re-calculate the neighbor reliability values. The second sinkhole is identified by the identification process since it now has the lowest neighbor reliability value in the network.

In this instance, node DD again was not alerted to the possibility of a sinkhole. This time node DD had one neighbor reliability value that was high because it had no routes with sinkholes included. The other two neighbor reliability values were nearly equal, lower, but fairly close to the higher value. Therefore, the method of identification and alerting based on incorporating the standard deviation did not indicate an outlier based on our thresholding scheme.

3) Two Randomly Distributed Sinkholes in the Network: There are two sinkholes in the network located near opposite corners of the grid (nodes I, V). Experimental results show that all but two of the other grid nodes were alerted to the presence of at least one sinkhole in the network by their neighbor reliability values. After applying the sinkhole identification process as described in this paper, each sinkhole was correctly identified by the alerted nodes.

Two of the nodes each have two neighbor reliability values that indicate the existence of one or more sinkholes. The first node, S, has two neighbors whose reliability values cross the threshold; both of the neighbors' reliability scores are affected by their routes through the same sinkhole. The single sinkhole was correctly identified.

The second node (L) also has two neighbors (H, K) with reliability values that indicate the presence of a sinkhole, as denoted by the asterisks in the Table IV. However, each neighbor's values are affected by routes through a different sinkhole. Therefore, if we follow the identification process in which we first place and query the neighbor with the lowest reliability value, one sinkhole is identified by node L. After the first identified sinkhole and all associated routes are removed from the network route lists, and the neighbor reliability values are re-calculated, the second sinkhole is identified by node L. However, after the associated routes are removed and the process followed a third time there were two false positives identified in the network.

Starting Node	Neighbors	Number Routes with Sinkholes vs. Number Routes	Neighbor Reliability Value
Node K Neighbors	F	1/3	3.999E-05
	G	2/6	3.882E-05
	L	3/9	3.958E-05
	P	3/8	3.885E-05
	Q	3/9	4.073E-05
Node L Neighbors	F	0/2	5.924E-05
	G	2/4	2.961E-05
	H*	3/4	1.592E-05
	K*	2/3	2.258E-05
	M	1/11	5.435E-05
	P	0/1	5.886E-05
	Q	3/8	3.888E-05
	R	1/8	5.245E-05

Table IV
NEIGHBOR LISTS FOR NODES K AND L

The two nodes that gave null results (did not indicate the presence of a sinkhole) were K and DD. We would expect nodes that have no routes through the sinkholes to be unable to alert upon the possibility of a sinkhole. However, both of these nodes have routes traversing each sinkhole. The neighbor reliability values did not alert these nodes because none of the values crossed the calculated threshold for the node. Investigation shows that in both cases, the lists of routes through every immediate neighbor included nearly half of the routes traversing a sinkhole, and the other half of the routes not traversing a sinkhole. Therefore, the mean of the route set for each neighbor of a node falls between the values reported for routes with no sinkhole, and the values for routes with a sinkhole (Section 5.2.1 number 4). (It should also be noted that node K is at the edge of the network, and node DD is in a corner of the network. Although this placement does not guarantee failure in identifying a sinkhole, it does lower the possibility due to the smaller number of neighbor nodes for comparison. However, the success or failure also depends upon the routes contained in the routing table of the node.)

The result is no values fall outside of the threshold, and the node does not receive an alert. In essence, by the perspective of these two nodes, routes including sinkholes are as common in the network as routes with no sinkholes, and therefore will not be identified as outliers. This argument foretells the results we will see in future cases with increased numbers of sinkholes.

The Table IV shows the neighbors of nodes K and L with their neighbor reliability values and the ratio of the number of routes that traverse the neighbor and a sinkhole to the routes that do not go through a sinkhole. In this case, node K has no neighbors for which the number of routes through a sinkhole is greater than the number of routes with no sinkhole nodes. L has two such neighbors - H and K (note that here we are not looking at node K, but as node K *as a neighbor of* node L). Nodes H and K both met the conditions we set for alerting node L of a possible sinkhole in the network. Node K did not receive any such alert.

4) Multiple Randomly Placed Sinkholes in the Network: Multiple randomly placed sinkhole nodes were placed in the network. Table V shows the number of sinkholes, alerts, sinkholes identified, sinkholes missed, and false sinkholes identified in the thirty node network. It is important to note that this table represents the number of node members that would alert upon a sinkhole and start the discovery process. However, with a protocol in place to determine which alerted node would take action, the other nodes would not initiate the identification process. As a consequence, the number of falsely identified sinkholes appears quite high. This is because several nodes would alert upon the same false node.

Table V shows that as the number of sinkholes increased, the number of nodes receiving alerts decreased. This was expected, because as sinkholes become more numerous, using the algorithm based on the standard deviation, localized neighborhoods of the network will see the results of the sinkholes as a norm, rather than an anomaly.

In this network we were able to identify all of the sinkholes

Sinkholes In Network	Alerts	Sinkholes Identified	Sinkholes Missed	False Sinkholes
0	26	0	0	4
2	28	2	0	2
4	25	4	0	$\geq$1
6	20	6	0	$\geq$1
8	10	7	1	$\geq$1
10	6	6	4	0
15	2	6	9	0
18	0	0	18	0

Table V
ONE STANDARD DEVIATION AS THRESHOLD

Sinkholes In Network	Alerts	Sinkholes Identified	Sinkholes Missed	False Sinkholes
0	0	0	0	0
2	13	2	0	0
4	14	4	0	0
6	12	5	1	0
8	8	5	3	0
10	3	4	6	0
15	1	0	15	0
18	0	0	18	0

Table VI
EXPERIMENTAL RESULTS - ADDITIONAL THRESHOLD CRITERIA

in networks with six or less sinkholes. In the cases of more than six sinkholes, at least one sinkhole was left unidentified. After the sinkholes were identified, the sinkholes and associated routes were removed from the routing tables and the neighbor reliabilities re-calculated. The identification process was repeated until there were no alerts in the network, or until any alerts were deemed non-productive because of loops in the process (i.e. node A points to node B as the neighbor with the lowest neighbor reliability value, who points to node C, to node D, to node E, to node A). In all of the networks with 8 or fewer actual sinkholes randomly placed in the network at least one false positive was identified.

5) New Criteria for Threshold: The original threshold criteria allowed too many false sinkholes to be identified. Such identification could result in friendly nodes being excluded from the network. After reviewing the data for every case, an additional threshold was added to the algorithm. This threshold excluded nodes with neighbor reliabilities that were not at least one order of magnitude less than the average of the local neighbor reliabilities from causing alerts. The results seen in Table VI show that there were no false positives reported when using the new threshold. For the networks with four sinkholes or less, all of the sinkholes were properly identified. For networks with six, eight, or ten sinkholes, up to two additional sinkholes were unidentified when compared to the results in the Table V. Since it is expected that it is highly unlikely there will be more than one or two misbehaving nodes in a network of thirty nodes, the tradeoff when using the enhanced threshold criteria is for less disruption in the network due to the removal of innocent nodes.

V. ANALYSIS OF NETWORK OVERHEAD

The method presented does not depend upon stored patterns, signatures, or rules. It does, however, require limited storage (s) of additional collected data and a small amount of energy (e) for calculations. Messages (m) are only required for identification when a sinkhole is detected. Therefore, the overhead o can be described by the equation:

$$o = s + e + m. \tag{10}$$

A. Storage

The AODV routing tables already house information about the "next hop" to known destinations. The "next hop" is an immediate neighbor. Therefore, the data we require to be stored in the routing table can be associated with the "next hop", or neighbor. We assign s_s as the storage space used to hold the number of packets sent and s_r as the storage space for the number of packets received. After the calculation for the neighbor reliability is completed, it is stored in s_{nr}. The additional storage space s required by a network of i nodes, each with b neighbors, can therefore be represented by the equation

$$s = ib(s_s + s_r + s_{nr}). \tag{11}$$

B. Energy

The detection and identification processes rely upon the calculation of the packet delivery ratio and neighbor reliability values assigned to the b neighbors of the i nodes in the network. The energy required for the packet delivery ratio calculation is represented by e_{pd}. Similarly, the energy required by the neighbor reliability calculation is represented by e_{nr}. Additional energy is required by the creation (e_{mc}) and distribution (e_{md}) of the identification messages m. The total additional network energy consumption e required by the detection and identification processes can therefore be described by the equation

$$e = m(e_{mc} + e_{md}) + ib(e_{pd} + e_{nr}). \tag{12}$$

C. Messages

Messages related to the identification process are generated and transmitted in the network only when a sinkhole is detected. As mentioned before, there are no additional network messages required for the detection process. The message overhead is dependent upon how far, or the number of hops h, the detecting node is from the sinkhole. Therefore, m_{Low} refers to the number of messages required for the assignments of L_{Low} along the path to the sinkhole. Also partially dependent upon the number of hops to the sinkhole are the number of neighbor list messages (m_{nl}) which are sent by nodes that have both the assigning node and L_{Low} in common. Whether the sinkhole participates in the process by assigning a L_{Low} greatly affects the number of messages generated. We therefore apply binomials j and k to reflect participation where j = 1 for no participation, and k = 1 if there is participation. The additional messages generated near the sinkhole when the sinkhole fails to participate is m_f, with m_p reflecting the number of messages generated during the "splash back" beyond the sinkhole due to its participation. Finally, we include the messages generated by the average number of "voters", m_v . The number of messages required by the identification method can be approximated by the following equation:

$$m = m_{Low} + m_{nl} + km_p + jm_f + m_v. \tag{13}$$

This equation can be simplified by including the parameters for the number of hops and the average number of neighbors common to L_{n-1} and L_n in each hop to:

$$m = h + 2h + 5k + 3j + 3 \tag{14}$$

or

$$m = 3h + 8k + 6j. \quad (15)$$

Figure 5 provides a graph depicting the actual messages sent by nodes on each ring compared to the estimated number of messages for the ring. A protocol for sharing the burden of starting the sinkhole identification process would need to be developed in a real network, and is not in the scope of this paper. With the proper protocol, only one node would start the identification process, limiting the number of messages as overhead.

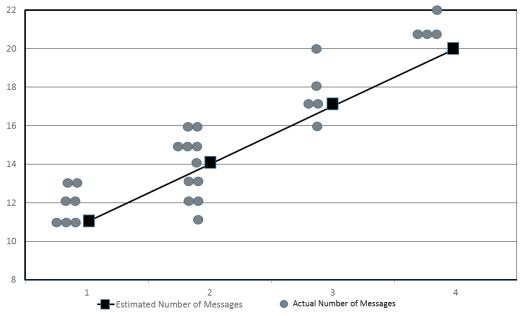

Figure 5. Number of Messages Per Ring

VI. Conclusion

In this paper we presented a hypothesis that, by adapting a methodology borrowed from the science of meteorology, we can utilize the data available at both the node and cooperative network levels to create a synoptic picture of the network health, providing indications of any intrusions or other network issues. Our major contribution is to provide a revolutionary way to analyze node and network data for patterns, dependence, and effects that indicate network issues at a distance. This method did not rely upon the conventional methods of stored patterns for comparison, but only proper analysis of a subset of the real time parameters of the network.

Simulations using the network described in Table I showed that the original scheme found false sinkholes in networks with eight or fewer sinkhole nodes. However, the addition of a second threshold resulted in the elimination of the false positives, with the trade off of increased false negatives in networks of six or more sinkholes. This tradeoff is justified because we can realistically expect a network to have fewer than six (sinkhole) attackers. Therefore, using the two thresholds provides proper intrusion detection for this type of attack.

Our results showed the number of nodes in a thirty node network that would start the identification process and identify the sinkhole(s) based upon the threshold criteria. For the sinkhole identification process to be implemented in a real network, a protocol for sharing the responsibility of identifying the sinkhole would need to be developed. The objective of the protocol would be to allow identification of the sinkhole while limiting the number of nodes starting the identification process, and the number of messages flooding the network. Possible strategies are a round robin system based upon time, or assignment of responsibility to cluster heads.

The synoptic analysis technique presented in this paper is founded upon comparing the counts of events in or effects on the wireless network. Other attacks that are based upon causing or affecting countable events that trigger changes in network characteristics are candidates for synoptic analysis. Attacks at the physical, network, and data link layers such as jamming, HELLO flood, and wormhole assaults are likely contenders. Challenges to the use of the technique described include the development of protocols to identify the initiating node(s) and the development of a proper network reaction. An additional challenge is a method to provide the network nodes with more distributed network data without increasing the controlling network traffic.

References

[1] Abror Abduvaliyev, Sungyoung Lee, and Young-Koo Lee. Energy efficient hybrid intrusion detection system for wireless sensor networks. In *Electronics and Information Engineering (ICEIE), 2010 International Conference On*, volume 2, pages V2–25. IEEE, 2010.

[2] Baruch Awerbuch, Reza Curtmola, David Holmer, Cristina Nita-Rotaru, and Herbert Rubens. Odsbr: An on-demand secure byzantine resilient routing protocol for wireless ad hoc networks. *ACM Transactions on Information and System Security (TISSEC)*, 10(4):6, 2008.

[3] Benjamin Jack Culpepper and H Chris Tseng. Sinkhole intrusion indicators in dsr manets. In *Broadband Networks, 2004. BroadNets 2004. Proceedings. First International Conference on*, pages 681–688. IEEE, 2004.

[4] Humaira Ehsan and Farrukh Aslam Khan. Malicious aodv: Implementation and analysis of routing attacks in manets. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 1181–1187. IEEE, 2012.

[5] Nisarg Gandhewar and Rahila Patel. Detection and prevention of sinkhole attack on aodv protocol in mobile adhoc network. In *Computational Intelligence and Communication Networks (CICN), 2012 Fourth International Conference on*, pages 714–718. IEEE, 2012.

[6] Keldor Gerrigagoitia, Roberto Uribeetxeberria, Urko Zurutuza, and Ignacio Arenaza. Reputation-based intrusion detection system for wireless sensor networks. In *Complexity in Engineering (COMPENG), 2012*, pages 1–5. IEEE, 2012.

[7] Enrique Hernandez-Orallo, Manuel D Serrat, J Cano, Carlos T Calafate, and Pietro Manzoni. Improving selfish node detection in manets using a collaborative watchdog. *Communications Letters, IEEE*, 16(5):642–645, 2012.

[8] Jeba Veera Singh Jebadurai, AAR Melvin, and IJR Jebadurai. Sinkhole detection in mobile ad-hoc networks using mutual understanding among nodes. In *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, volume 3, pages 321–324. IEEE, 2011.

[9] Shiv Kumar and RC Joshi. Design and implementation of ids using snort, entropy and alert ranking system. In *Signal Processing, Communication, Computing and Networking Technologies (ICSCCN), 2011 International Conference on*, pages 264–268. IEEE, 2011.

[10] Xu Li, Rongxing Lu, Xiaohui Liang, and Xuemin Shen. Side channel monitoring: packet drop attack detection in wireless ad hoc networks. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–5. IEEE, 2011.

[11] Yang Li, Bin-Xing Fang, You Chen, and Li Guo. A lightweight intrusion detection model based on feature selection and maximum entropy model. In *Communication Technology, 2006. ICCT'06. International Conference on*, pages 1–4. IEEE, 2006.

[12] S Umang, BVR Reddy, and MN Hoda. Enhanced intrusion detection system for malicious node detection in ad hoc routing protocols using minimal energy consumption. *IET communications*, 4(17):2084–2094, 2010.

[13] Giovanni Vigna, Sumit Gwalani, Kavitha Srinivasan, Elizabeth M Belding-Royer, and Richard A Kemmerer. An intrusion detection tool for aodv-based ad hoc wireless networks. In *Computer Security Applications Conference, 2004. 20th Annual*, pages 16–27. IEEE, 2004.

[14] Han Zhijie and Wang Ruchuang. Intrusion detection for wireless sensor network based on traffic prediction model. *Physics Procedia*, 25:2072–2080, 2012.