Temporal Analysis of Routing Activity for Anomaly Detection in Ad hoc Networks

Haitao Liu Rajiv Gupta
Department of Computer Science
The University of Arizona
Tucson, Arizona 85721
{haitaol,gupta}@cs.arizona.edu

***Abstract*— Mobile ad hoc network (MANET) faces serious security threat due to lack of consideration of security in design and inherent weaknesses. Anomaly detection techniques as well as preventive measures are in urgent need to protect ad hoc network. We propose an innovative anomaly detection technique based on temporal analysis of routing activities. Our approach uses data mining techniques to discover the underlying patterns of routing behavior and detect anomaly by comparing routing activities with the patterns. We tested the approach with two common attacks targeted at routing protocols and the experiment results show it is effective to detect routing disruption.**

Keywords- Ad hoc network, security, anomaly detection, temporal analysis, routing

I. Introduction

Ad hoc network features a unique peer-to-peer structure where there is no ready-to-use authority and members of the network have to depend on each other to communicate. Ad hoc networks are more vulnerable to attacks than other networks. Current on-demand routing protocols did not take security into consideration. With on-demand routing protocols, the communicating counterparts have little control on which nodes to choose to serve on a path. A malicious node can mount many kinds of attacks to bring the communication under its control. Several attacks that exploit the weaknesses of AODV protocols are described in [1]. By attacking routing protocols, malicious users can cause longer delay and packet loss. They can even control visible data in the network by employing subtle attacking strategies.

General approaches are desired to detect both known and as yet unknown attacks. In this paper, we propose a *lightweight anomaly detection* approach to detect *routing disruption* attacks. To interfere with the normal function of an ad hoc network, attackers affect the routing process in one way or another to launch attacks. Because ad hoc routes are alive for a relatively short time, attackers must repeat the attack procedure periodically. Recognizing that the essential task of protecting an ad hoc network is to protect the routing process, our approach tries to detect routing disruption attacks by watching routing activities closely. Observed routing activity is compared with patterns extracted through data mining from normal routing activity to identify anomalous behavior. By monitoring routing activities only, we lower the cost of anomaly detection. Without using information specific to certain known attacks, our approach is made general enough to deal with multiple kinds of attacks. We devise a novel temporal analysis algorithm that uses per route information to accurately capture the temporal relationships between routing events. Our experiments demonstrate that this algorithm is able to clearly distinguish between normal activity and abnormal activity during blackhole and wormhole attacks.

The rest of the paper is organized as follows. In section 2, we present the relevant background information on ad hoc routing mechanism and intrusion detection techniques. In section 3, we present our anomaly detection algorithm in detail. Results of experiments are presented in section 4. Related work is discussed in section 5 and conclusions are given in section 6.

II. Background

The dominant routing protocols used in ad hoc networks are on-demand routing protocols (e.g., AODV and DSR). In an on-demand routing protocol, the source node floods the whole network with routing discovery messages for the destination node. The request message also carries with it the up-to-date state information for the source node. Node receiving the request message establishes a reverse route entry to the source. If a node has the routing information about the destination or it is the destination itself, it sends back routing reply message. When the routing reply message travels across the network, it establishes route entries towards the destination on nodes along the path.

Blackhole attack tries to create a blackhole in the network to trap all the traffic. Attacker can generate and disseminate fake routing information to make other nodes believe that a fresher or better path can be used through attacker.

Wormhole attack is a more sophisticated attack. It is carried out by a pair of nodes. Attackers create a tunnel between them. When one attacker receives a packet, it wraps it and sends it through the tunnel. The other attacker unwraps the packet and replays it in the network. When replayed packet arrives at the destination, it appears to travel shorter path because many hops between the two attackers are not counted. Through wormhole attack, attackers can control the

data appearing in the network.

Rush attack [3] exploits duplicate suppression at each node. An attacker disseminates route request messages quickly throughout the network, suppressing any later legitimate route request messages when nodes drop them due to the duplicate suppression. [3] proposes random dropping of route request messages to alleviate the damage of rush attack.

III. Temporal Analysis of Routing Activity

We recognize the disturbance of attack on routing activity and propose to analyze and detect routing disruption from *temporal* prospective. Our approach detects routing disruption by watching the *lifetime of each route*, from its establishment to removal – either because of routing error or timeout, and identifying suspicious routing activities using patterns extracted from normal routing activities during training.

1) Data Collection: Our approach focuses on the basic property – time. By considering only basic time information, our approach is universal despite the difference between ad hoc routing protocols. We define the history of a route entry as the ordered sequence of timestamps when the routing events happened to this route entry. The history of a route is denoted by $H : \{t_i | i^{th}\ routing\ event\ happens\ at\ time\ t_i\}$.

We call the history recorded above as *raw* because it contains the actual timestamps of routing events. The next step is to transform the raw history data to ease its later analysis. First we convert the absolute timestamps to relative timestamps with respect to the time when the route entry was established. The first routing event, which is always the establishment of the route entry, has the timestamp value of zero. The values for the remaining events are the timespans from the moments when the events take place to the moment when the route entry is established. That is, the history of the route entry $H : \{t_1, t_2, ..., t_n\}$ is converted to $H' : \{0, (t_2 - t_0), ..., (t_n - t_0)\}$.

Next, to enable comparison of different route entries in terms of their life time, our approach assumes that the lifetime of route entry consists of a certain number of stages of equal durations. Using the relative occurrence times of the events, we compute how many events take place in each stage. Finally, the numbers of events in each stage are normalized based on the total number of routing events of the route entry. Through this transformation we normalize the different route entries which may differ in the duration of their lifetimes and the number of events that occur during the lifetimes. The final result of this transformation is that the history of a route entry is converted to the frequency distribution of its routing events during its lifetime. We refer to this as the *event frequency vector* of that route entry. The above transformation process is summarized in Figure 1, where N is the number of stages and D denotes the duration of each stage.

2) Pattern Extraction: Our approach uses data mining techniques to discover the underlying patterns of routing activities under normal operations. The patterns are used later at runtime to judge if routing disruption happens. We use two data-mining methods, *k-means* and *k-harmonics means* [6], to analyze and extract patterns from frequency distribution of routing events calculated above.

Data: history of routing entry $H = \{t_i | 1 \leq i \leq n\}$;
Result: frequency distribution of routing events;
begin
  $H' = \{t_i^{'} | t_0 = 0, t_i^{'} = t_i - t_0$ for $2 \leq i \leq n\}$;
  $D = (t_n - t_1)/(n-1)$;
  $Freq = \{f_i | 1 \leq i \leq N, f_i = |\{t_p^{'}, ..., t_q^{'}\}|/|H|$ where $(i-1) * D \leq t_p^{'} \leq t_q^{'} < i * D\}$
  return $Freq$;
end

Fig. 1. Routing History Transformation Algorithm.

We observed that the activity of single route entry contains limited information because not many routing events happen for one route entry under normal operation. So we combine the event frequency vectors of several route entries together. That is, we sum the frequency value at each stage and then average the result by the number of used route entries to get a *merged event frequency vector* of several route entries. We then run data mining algorithms on the merged vectors. The merged event frequency data provides us with more insight into the routing behavior in general.

In practice, we only watch the activity of route entries that have been referred to at least once to forward a packet. We make this decision because lots of route entries are set up but never used. Such route entries provide little information for routing activity analysis and actually decrease the efficiency of detection because there are too many of them and they distract our attention from more meaningful routing activities. So we decide to disregard the unused route entries.

3) Distributed Detection of Routing Disruption: Using the pattern extraction algorithm described above, we can get a set of patterns. We denote the pattern set by $P : \{g_1, g_2, ..., g_m\}$, in which each g_i is a pattern group like $g_i : \{p_{i1}, p_{i2}, ..., p_{in}\}$. Each pattern group is the result of one run of k-means or k-harmonic means algorithm. Using these patterns, we propose a cooperative distributed architecture for detecting routing disruption.

Our approach uses patterns in a different way other than measuring distance between observation and patterns. We consider the distribution of event frequency vectors among clusters corresponding to patterns. From the training data, we can get the number of event frequency vectors in each cluster. Further, we calculate the percentage of all vectors that belong to each cluster. We let each pattern bear a value which indicates what percentage of event frequency vectors among all vectors should belong to its cluster in normal situation. Thus, the representation of pattern group described before is changed from $g_i : \{p_{i1}, p_{i2}, ..., p_{in}\}$ to $g_i : \{(p_{i1}, per_{i1}), (p_{i2}, per_{i2}), ..., (p_{in}, per_{in})\}$, where

per_{in} denotes the standard percentage for a pattern.

In practice, we collect a certain number of event frequency vectors before examining the routing behavior in the recent past. For each vector, we calculate its distance to each pattern in one pattern group. The vector is affiliated with the pattern that is closest to it. After all vectors are affiliated with some pattern, we can calculate the percentage of collected vectors in each pattern and compare it with the standard percentage of the pattern. We sum the difference in percentage of each pattern as the indication of alert level for routing activity. To increase the accuracy of detection, we use multiple pattern groups together and add the difference in each group. The final result is compared with a threshold to decide if routing process is under attack. The detection algorithm is summarized in Figure 2. This algorithm is run after we have collected m event frequency vectors and we have n pattern groups to compare with.

Input: event frequency vector set
$V = \{F_i | 1 \leq i \leq m\}$, and pattern set
$P = \{g_i | 1 \leq i \leq n\}$;
Result: alert level;
begin
$alert = 0$;
foreach $g_i = \{(p_{i1}, per_{i1}), ..., (p_{ik}, per_{ik})\} \in P$;
begin
$C = \{c_i | 1 \leq i \leq k, c_i = \phi\}$;
foreach $F_i \in V$;
begin
find p such that $|F_i - p_{ip}| \leq |F_i - p_{iq}|$ for any $q \neq p$;
$c_p = c_p \cup \{F_i\}$;
end
foreach $c_j \in C$;
begin
$alert = alert + |\frac{|c_j|}{|V|} - per_{ij}|$;
end
end
return $alert$;
end

Fig. 2. Detection algorithm.

Overall, our anomaly detection system runs in the following way. Each node runs the detection algorithm independently. When a route entry is established, the host adds a record corresponding to the new route entry and records the setting-up time. When later a routing event happens for this route entry, the time is recorded too. Finally, when the route entry is removed, the history of the route entry is transformed to the event frequency vector using the algorithm in previous section. After collecting enough number of past event frequency vectors, the algorithm in Figure 2 is run on the data to decide if the current situation of routing activities is normal.

IV. Experimental Evaluation

A. Experimental Setup

We have implemented and evaluated our anomaly detection algorithm in the NS-2 network simulator. We implemented two kinds of attacks, blackhole and wormhole attacks. We choose a test area of 1000m×1000m. On the test area, fifty nodes are placed randomly and move following random way point model. We choose AODV as the testbed of our approach, but our algorithm is applicable to other routing protocols. As we show in later sections, there is distinct difference in alert level between normal routing activities and activities under attack. By choosing an appropriate threshold for alert level, nodes can detect abnormal routing behavior.

B. Training

Using the above settings, we generated twenty scenarios as the training set. From the training process, we obtained 124264 records of normal routing activity information. We used the *k-means* and *k-harmonics means* algorithms on collected routing activities to compute the underlying patterns. We trained with pattern numbers from three to six. After applying the above, we obtained *sixteen groups of patterns* and their corresponding *percentage of routing activities*. The training is costly, but it only needs to be done once.

C. Results

Distance from Pattern Groups. We first show that the routing activity under normal operation differs from that under attack significantly. We generated ten scenarios and collected routing activities under *normal operation*, *blackhole attack*, and *wormhole attack*. Figure 3 shows the results for each of the sixteen pattern groups. The figure indicates that routing behavior under attack is very different from that under normal operations. For all pattern groups, routing activities under normal operations are very similar to the pattern groups and the distance between them are almost negligible. Blackhole attack is very aggressive and causes the most deviation of routing behavior. Wormhole attack causes smaller deviation but is still significantly large.

Anomaly Detection. Next we traced the change of alert level in a single testing scenario. We ran the same scenario three times under normal operations, blackhole attack, and wormhole attack respectively. In the standard test, we created the test scenarios using the same parameters used to create training scenarios. Figure 4 shows the average alert level of one typical scenario. As expected, the alert level is the highest under blackhole attack, which agrees with the maximal deviation from standard pattern caused by blackhole attack noticed in Figure 3. The difference in alert level between normal and wormhole cases is also apparent. For any scenario, the alert level in normal operations is well below the alert level of blackhole and wormhole attacks. A threshold is chosen so that it can distinguish most normal operations from operations

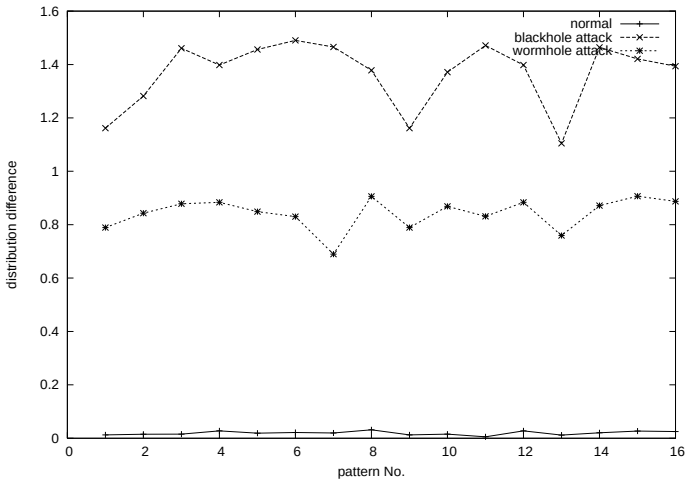

Fig. 3. Distance from pattern groups under normal operation, blackhole attack and wormhole attack.

affected by attacks. This threshold can be used to decide if the routing process is under attack.

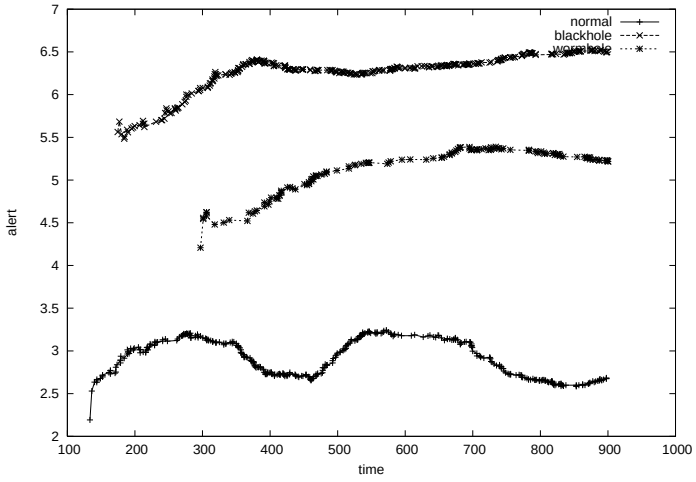

Fig. 4. Alert level over time in standard tests.

V. Related Work

Watchdog [7] proposed first to detect misbehaving nodes. With Watchdog, when nodes forward packets, they watch if the next hop relays the packet properly. If a node is found to fail to relay packets certain times, its rating is decreased and the later routing discovery avoids selecting nodes with low rating. The same idea is developed in [8], which punishes the lazy nodes by declining there service request later. Anomaly detection system dealing with active attackers are studied in several other works. [10] proposes an intrusion detection framework using agent technology. [5] suggests a distributed architecture. They exploit the relationship between node movement and route table changes to detect suspicious activities. [4] uses data-mining to detect anomaly. They monitor dozens of features and develop classifiers for the relationship of each individual feature and the remaining feature set. In detection, every classifier is applied to one event and the results are accumulated. If the accumulation is above certain threshold, the event is considered abnormal. [9] does anomaly detection with state machine. It monitors the routing discovery procedure step by step. If the procedure enters a wrong state, it indicates some fake information is trying to interfere with the normal process. [2] fights wormhole attacks by authenticating either a precise timestamp or location information combined with a loose timestamp.

VI. Conclusions and Future Work

In this paper we proposed a new approach to detect abnormal routing behavior. Our approach focuses on identifying suspicious operations by watching routing activity itself. By limiting monitoring and analysis to routing activity, we reduce the cost of anomaly detection significantly. A key contribution of work is the identification of an algorithm that transforms routing activity information into a form that is suitable for detecting anomalies. Our experiments demonstrate that our approach is effective in detect routing disruption caused by blackhole and wormhole attacks. In this work, we address the problem of detecting anomaly routing activity. Currently we are working on the problem of fighting back routing attacks based on routing activity observation.

References

[1] P. Ning and K. Sun, "How to Misuse AODV: A Case Study of Insider Attacks Against Mobile Ad-hoc Routing Protocols," In *Proceedings of the 4th Annual IEEE Information Assurance Workshop*, pages 60-67, West Point, June 2003.

[2] Y. C. Hu, A. Perrig, and D. B. Johnson, "Packet Leashes: A Defense Against Wormhole Attacks in Wireless Ad Hoc Networks," In *Proceedings IEEE INFOCOMM, The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol. 3, pages 1976-1986, IEEE, San Francisco, CA, April 2003.

[3] Y. C. Hu, A. Perrig, and D. B. Johnson, "Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols," In *Proceedings of the ACM Workshop on Wireless Security* (WiSe), pages 30-40, ACM, San Diego, CA, September 2003.

[4] Y. A. Huang, W. Fan, W. K. Lee and P. S. Yu, "Cross-Feature Analysis for Detecting Ad-hoc Routing Anomalies," In *Proceedings of the 23rd International Conference on Distributed Computing Systems* (ICDCS), May 2003.

[5] Y. G. Zhang, W. K. Lee and Y. A. Huang, "Intrusion Detection Techniques for Mobile Wireless Networks," *ACM/Kluwer Wireless Networks Journal* (ACM WINET), Vol. 9, No. 5, September 2003.

[6] B. Zhang, "Generalized k-harmonic Means - Boosting in Unsupervised Learning," Technical Report HPL-2000-137, Hewlett-Packard Labs, 2000.

[7] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," In *Proceedings of the Annual International Conference on Mobile Computing and Networking* (MobiCom), Boston, August 2000.

[8] S. Buchegger and J. Y. Le Boudec, "Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad-hoc Networks," *10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*, Canary Islands, Spain, pages 403-410, January 2002.

[9] C. Y. Tseng, P. Balasubramanyam, C. Ko, R. Limprasittiporn, J. Rowe, and K. Levitt, "A Specification-based Intrusion Detection System for AODV," In *Proceedings of the First ACM Workshop on Security of Ad Hoc and Sensor Networks* (SASN), pages 125–134, Fairfax, Virginia, 2003.

[10] O. Kachirski and R. Guha, "Effective Intrusion Detection Using Multiple Sensors in Wireless Ad-Hoc Networks," In *Proceedings of the 36th Hawaii International Conference on System Sciences* (HICSS), 2003.