Feature Selection in Intrusion Detection System over Mobile Ad-hoc Network

Xia Wang, Tu-liang Lin, Johnny Wong

Computer Science Department
Iowa State University
Ames, Iowa 50010
{jxiawang, tlin, wong}@cs.iastate.edu

Abstract

As Mobile ad-hoc network (MANET) has become a very important technology the security problem, especially, intrusion detection technique research has attracted many people's effort. MANET is more vulnerable than wired network and suffers intrusion like wired network. This paper investigated some intrusion detection techniques using machine learning and proposed a profile based neighbor monitoring intrusion detection method. Further analysis shows that the features collected by each node are too many for wireless devices with limited capacity. We apply Markov Blanket algorithm [1] to the feature selection of the intrusion detection method. Experimental studies have shown that Markov Blanket algorithm can decrease the number of features dramatically with very similar detection rate.

1 Introduction

In recent years, many wireless devices are available and tend to make life more convenient. e.g., mobile laptop computers, PDAs, and wireless phones. A mobile ad-hoc network (MANET) is composed by a group of mobile wireless nodes without a fixed network infrastructure. It shows many characteristics that are not shared by wired networks, such as shared open medium, dynamically changed network topology, and limited capacity. All those facts tell that new research methods should be investigated for the security problem. There are many important applications for mobile wireless ad-hoc network (MANET), for example, military operations, emergency rescue and home and community networking. System security is significant for the application of those systems.

Wired network may use intrusion prevention to secure the system. Firewalls can be installed at the routers or switchers to monitor and filter network traffic. However, MANET doesn't have this kind of facilities. There is no centered monitoring point for a MANET. Therefore, intrusion detection may be more suitable for wireless networks. An intrusion detection system analyzes network or system activities captured in audit data and uses patterns of well known attacks or normal profile to detect potential attacks. There are two different analyzing methods: misuse detection and anomaly detection. Misuse detection uses the "signature" of well known attacks to match activity as an attack instance. This method is not effective against new attacks. Anomaly detection uses established profile to filter out those system behaviors that deviate from the profile. Anomaly detection can be effective because it doesn't assume the knowledge of attack patterns.

Many intrusion detection systems in MANET have each mobile node monitoring their own traffic and report the measurements or statistics to other nodes when they are asked [Huang cooperative]. Self-monitoring is natural way to implement IDS and is easy to be implemented. But it also has the disadvantage to faking measurements. As each node has full control of its own IDS and the traffic going through a node is only monitored by itself, it is very easy for a malicious node to fake traffic measurements for itself or for the traffic going through it. We propose *neighbor-monitoring scheme* in our IDS, in which each mobile node monitors its neighbors' traffic. As each node monitors its neighbors, this node is also monitored by all its neighboring nodes it is hard for a neighboring node to fabricate traffic information for the monitored node unless all the neighboring nodes have been compromised.

The main task of the intrusion detection system is to discover the intrusion from the network packet data or system audit data. One of the major problems that the intrusion detection system might face is that the packet data or system audit data could be overwhelming. For example, one of the data set provided by MIT Lincoln Lab, which collect the network traffic from Air Force's Research Laboratory, contains around 400 MB in per day base (http://www.ll.mit.edu/IST/ideval/). For wireless network, due to the limited capacity of wireless devices, choosing those features that can best characterize the behavior of

network is very important. In this paper, we propose a profile based neighbor monitoring anomaly detection with feature selection. In wireless network, the way a node communicates with other nodes is by exchanging messages. Therefore, a node's behavior can be obtained by monitoring the network traffic. In our intrusion detection system each node monitors its neighboring nodes' network traffic and built a profile during offline training. Then the profile is used as a threshold to detect abnormal behavior in the network. We select all possible features as the object of the monitoring. Totally there are 25. That is suitable for a small wireless network which has only a few nodes. But it requires a big amount of capacity for very large network. Then a Markov blanket discovery algorithm is applied to obtain the Markov blanket set of on those 25 features. Our experiments studies show that this algorithm displays high accuracy in feature selection.

The rest of paper is organized as the follows. We briefly introduce some of the related work in MANET intrusion detection and discuss the routing protocol we considered. Then in section 3, we describe our intrusion detection approach and Markov Blanket discovery. In section 4, some experiments results are presented and analyzed. Section 5 is the conclusion and future work.

2 Related works

2.1 Other intrusion detection in MANET

There are many research works that have been done in this area. The group led by Wenke lee at Georgia Institute of Technology has conducted different method in this area, such as machine learning algorithm and data mining technologies [4]. One work is related to ours is their cooperative intrusion detection system in [5]. They developed an intrusion detection system based on [4] which is not able to detection intrusions, but also can detect the type of attack and the source of attack. This is achieved by exchanging monitored data between neighboring nodes. Then they used a cluster-based method to select a cluster in each group to do the monitoring such that the efficiency of the system can be improved due to the limited power supply of wireless nodes. The traffic related features considered in their monitoring are listed in Table 1. The total number of features they claim is 132.

Table 1: Traffic Related Features

Dimension	Values
Packet type	Data, route (all), ROUTE REQUEST, ROUTE REPLY, ROUTE ERROR and HELLO message.
Flow direction	Received, sent, forwarded and dropped
Sampling periods	5 seconds, 60 seconds and 900 seconds
Statistics measures	Count the average and standard deviation of number of packet or size of data packets,

2.2 AODV Routing Protocol

AODV, or Ad hoc On-demand Distance Vector, is an on-demand touting protocol. The route discovery is not started until it is required (on-demand). The protocol operates in two mechanisms: route discovery and route maintenance. Route discovery is used when the packet sender has no route available in its routing table. It broadcasts a ROUTE REQUEST packet into the network. A node receives a fresh ROUTE REQUEST will check its route table to see whether it has a route to the requested destination. It replies if there is one otherwise, the ROUTE REQUEST is forwarded. Before forwarding, it keeps a reverse path to the source node in its route table. The route table records the route information of the next hop, the distance and the current highest sequence number it has seen. Route maintenance starts when changes in the network topology invalidate a cached route. It is used to notify the source node or to trigger a new route discovery.

3 Markov Blanket Feature Selection Techniques

3.1 Profile based neighbor monitoring intrusion detection technique

We propose a profile-based intrusion detection system for wireless ad hoc network. In the intrusion detection system, each node builds a profile for each of its neighbors. The profile includes all features listed in Table 1. For the packet type, we also consider the data packet size. They are all traffic related features (we will extend to all network features in our future work). Holding the profile, a node can use it to monitor its neighbor nodes' behavior. Once the traffic feature exceeds a certain threshold, for example, exceed the normal range [mean – 3*STD, mean + 3*STD], an alert should be reported.

Our method is very simple. Each node only needs to save a profile for all other nodes in the network. But further analysis, we'll see that the features each node is monitoring are too many. Consider a network with 10 nodes, each statistics measurement accounts for 4 bytes. Then each node requires around 5KB to save the profiles for all other nodes, and another 1.5KB for data at each sampling period. But if a node need to records a period of data, for example, a day, it requires 500MB capacity which is big. This is only for a wireless network with 10 nodes. What if there are more than 100 nodes in the network? Therefore, decreasing the number of features that need to be monitored, and also guarantee the accuracy is very important. We consider those feature values that need to be collected in the time period. They are listed in Table 2. Packets can be divided into 6 types (We consider both data packets and the size of data packets, the total packet type is 7 in fact) and each packet type can have 4 flow directions and 3 kinds of sampling rates. We use the combination of packet types, flow directions and sampling periods to produce the features. Totally, there are (7*4-3)*3*2=150 features, but only (7*4-3) * 3= 75 features will be considered for feature selection (In our simulation, 25 features are considered, as for each sampling period, we can get the results similarly).

Table 2: Monitored Traffic Related Features

Type	Features
Packet type	Data Size, Data Num, route (all), ROUTE REQUEST, ROUTE REPLY, ROUTE ERROR and HELLO message.
Flow direction	Received, sent, forwarded and dropped
Sampling periods	5 seconds, 60 seconds, 900 seconds.

In our feature selection, we infer the related features from the Markov blanket and try to decrease the number of features without affecting the detection rate. For simplicity, we assume no hidden variables and no missing values in our problem although it is not quite realistic.

We use the score-based approach to learn the Bayesian network structure from the simulation data. Since 25 features can produce huge amounts of Bayesian network structure, it is quite unrealistic to evaluate all possible network structures. The hill-climbing has been used to search the best quality network structure. After obtaining the best network structure, we infer the Markov Blanket from the network structure.

After finishing the feature selection, we use some learning algorithms provided by Weka [6], a machine learning software, to evaluate the selected features by comparing the detection rate using only the selected features with the detection rate using all the original 25 features.

3.2 Markov Blanket Discovery

Markov blanket discovery is one of the feature selection approach that could provide all the relevant features when a predict feature or class attribute is given. The selection of Markov blanket is based on the d-separation rule of the Bayesian network. When given a specific attribute, which is node in the Bayesian network, Markov blanket for the attribute is the set of nodes composed of the attribute's parents, its children, and its children's parents.

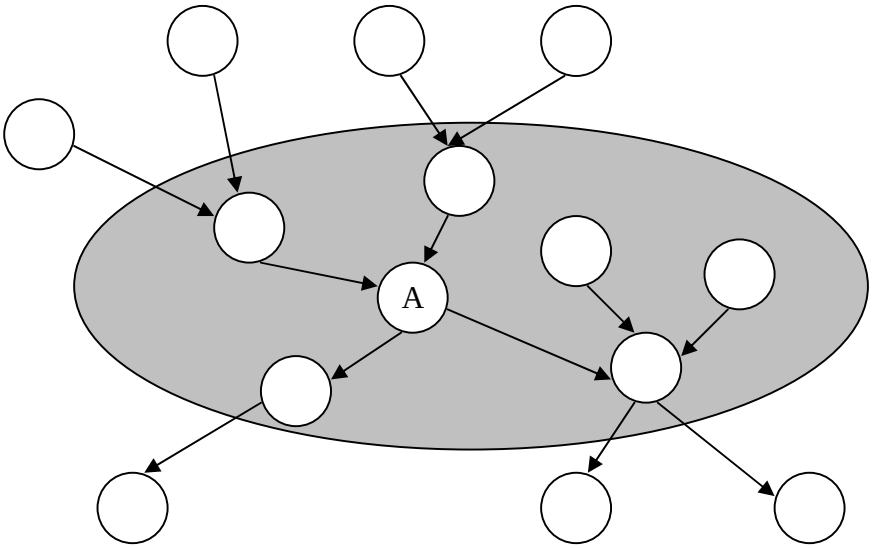

Figure 1. the Markov Blanket of Node A

Theoretically, given a faithful Bayesian network structure of the training data set, those features that are identified by the Markov blanket indeed block all the influence of the other features. However, the most difficult part of the problem is that how could we get a faithful Bayesian network structure from the training data set.
There are two kinds of approach to infer the Bayesian network structure from the training data. The first one is score-based method and the other is constraint-based method. The constraint-based method employs statistical tests on the data set for deciding the existence of edges in the Bayesian network. The accuracy of the constraint-based method depends heavily on the size of the data set. A huge data set can provide a more accurate statistical independence test. On the other hand, the score-based approach might suffer the local minima. In this project, we try to use a small data set to infer the Markov blanket, so we adopt the score-based approach here.
Generally, there are two kinds of score which have been used to evaluate the Bayesian network structure. One is Bayesian measure and the other is MDL (Minimum description length) measure. The MDL have been suggested by the research of Remco R. Bouckaert[1], which indicates that the performance of learning Bayesian network structure using MDL score is slightly better.

The main difference between the Bayesian measure and MDL measure is that MDL measure punishes the complex edge structure. Therefore, we might obtain a much simpler network structure than Bayesian measure.
Table 2 contains the notations that will be used to calculate the MDL score.

Table 3: the Notations for MDL Score Equation

Notation	**Meaning**
U	The set of attributes {X1,X2,….Xn}, n>=1
n	The total number of attributes
Xi	An attribute which take values from {Xi1,Xi2….}
Ri	The total number of values of Xi
D	The data set over U
N	Number of instances in D
Bs	A Bayesian network structure over U
Pi	The set of parents of Xi in Bs
Wij	The jth instantiation of Pi.
Qi	The total number of value combinations of Pi in Bs
Nijk	The number of cases in D in which Xi=Xik and Pi=Wij

Nij	$\sum_{k=1}^{Ri} Nijk$

Using the above notations, the MDL score can be calculated as the following formula.

L(Bs,D) = logP(Bs)-N*H(Bs,D)-1/2*K*logN [1]

where

K= $\sum_{i=1}^{n} Qi * (Ri - 1)$ and H(Bs,D)= $\sum_{i=1}^{n}\sum_{j=1}^{Qi}\sum_{k=1}^{Ri} -(Nijk / N) * \log(Nijk / Nij)$

Now we have the score that can be used to measure the quality of the Bayesian network. What we need now is a search algorithm that can help us to find the best score Bayesian network structure. Due to the time limitation of this project, we adopt the simplest search algorithm, hill-climbing. Although we might face many local minima, we can try to run this algorithm several times and choose the best one. The following is the hill-climbing algorithm that we use.

```
Function Hill-Climbing() return BayesNetStructure finalBNS
Inputs: BayesNetStructure initBNS
Local variables: BayesNetStructure current
        BayesNetStructure neighbour
current←initBNS
neighbour←a highest MDL score  successor of current
while ( current.getMDLscore < neighbour.getMDLscore)
{
   current←neighbour
   neighbour← a highest MDL score  successor of current
}
finalBNS←current
```

Figure 2 the hill-climbing algorithm

The MDL score can be seem as the heuristic function of the search algorithm. The initial Bayesian network structure is generated from a random attribute list. After obtaining the final Bayesian network structure, we infer the Markov Blanket from the final Bayesian network structure.

3.2 Implementation

The learning program takes two input arguments, a name files and data file. The name file contains the attribute names and the definition of the attributes. The data file only contains the attribute value of each instance and the values are separated by ",". The figure 3 shows the definition of the attribute types that our program can take. Some attributes types, such as date and time, are not a good attribute for Bayesian network. The program will ignore this kind of attributes. The name file is stored in a list structure and the data file is stored in an array.

continuous
The attribute takes numeric values.
date
The attribute's values are dates in the form YYYY/MM/DD or YYYY-MM-DD.
time
The attribute's values are times in the form HH:MM:SS.
a comma-separated list of names
The attribute takes discrete values, and these are the allowable values.
discrete N for some integer *N*
The attribute has discrete, unordered values, but the values are assembled

	from the data itself; *N* is the maximum number of such values.
ignore	The values of the attribute should be ignored.
label	This attribute contains an identifying label for each case, such as an account number or an order code.

Figure 3 the Attribute Types of the Name File

The adjacent matrix is employed to store the network structure. An n*n Boolean matrix is used for each network structure where n is the number of attributes. If the value of the Boolean adjacent matrix at row i and column j equals to true, it means that an edge from node i to node j exists. The following is an example of the adjacent matrix and its mapping graph.

	1	2	3	4
1	True	True	False	False
2	False	True	True	False
3	True	False	True	False
4	False	False	True	True

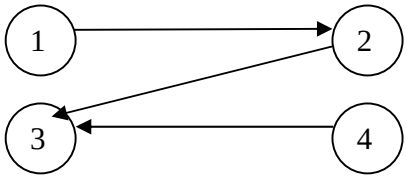

Figure 4 an Example of an Adjacent Matrix and its Mapping Graph

The first Bayesian network structure is generated from a random list containing all the variables. For each variable, we connect it with the next five variable retrieved from the random list. We use the following algorithm to form the random list.

```
Function produceRandomList() return List finalList
Inputs: List orderedList
Local variables: integer index, count;
count←the size of the orderedList
while (orderedList is not empty)
{
   index←random()*count
   finaList.add(orderedList[index])
   orderedList.remove(index)
   count←count-1
}
```

Figure 5 the Random List Generating Algorithm

The neighbors of a Bayesian network structure are generated by adding an edge or deleting an edge. Suppose there are n nodes and m edges in the current network structure. The total neighbors from deleting an edge are m and the total neighbors from adding an edge are $\binom{n}{2}$ - m. We will calculate the MDL score for each neighbor. The hill-climbing algorithm will compare the current MDL score and the neighbor with the highest MDL score. If the neighbor with the highest MDL score has higher MDL score then the current structure, then assign the neighbor with the highest MDL score to the current and iterate. If the neighbor with the highest MDL score does not higher than the current structure, it means that the current structure is the best structure we can find using the hill-climbing algorithm, so we just discover the Markov Blanket from the current structure.

4 Experimental studies

4.1 Experiment setup

The implementation of our algorithm has been described in section 3.2. The wireless simulation has been conducted on ns2.27 network simulator [9]. The verification of feature selection is done using Weka [8]. We simulate a mobile wireless network with 10 nodes. The simulation time is 1000 seconds.

4.2 Attack implementation

We implement the Black hole attack in AODV as described in [4]. A malicious node *Nm* broadcasts ROUTE_REQUEST message with a selected source node and destination and a fake maximum sequence number. In the ROUTE_REQUEST packet, the malicious node *Nm* claims a one hop distance to the source node. The fake ROUTE_REQUEST is then flooded in the network as it has the highest sequence number. All nodes that receive the ROUTE_REQUEST packet will update their route table with a reverse path to the victim through *Nm*. If Nm sends several ROUTE_REQUEST messages with different source node, eventually it attracts most of traffic in the network. Just like a black hole in the universe.

4.3 Experimental Results

We run ns2 network simulator to get classified data set. Each data set contains all 25 features collected for a neighbor of a selected node and the classification. For instance, node 9 has 9 datasets, each of which corresponds to a node that is monitored by node 9. We can use dataset 9-* to denote the corresponding dataset. Since each node monitors all other neighbors, for a 10 nodes network, there are totally 81 data sets. We only select the data set for the malicious node. For instance, in the simulation we set node 0 as the node that sends out all bogus ROUTE_REQUEST message. We randomly pick a dataset for node 0 for our testing. Since the hill-climbing algorithm can only find the local maximum, we run this algorithm several times and chose the feature set that has the highest MDL score. Table 3 lists the feature selection results for dataset 1-0 with different runs.

Table 4: Feature selection results for different runs (using dataset 1-0)

Features	MDL Score
DataFrwd, NBRREQRecv, NBDataSend, NBRREQDrop	-1614
DataFrwd, NBRERRFrwd	-1932
NBDataFrwd, NBRREQDrop	-2066
DataFrwd, NBRREPDrop	-2063
NBDataFrwd, DataRecv, NBHELLOSend	-1890

(*notation: in above table we used some abbreviation representation: Data represents the size of data packet; NBData represents the number of data packets; RREQ, RREP, RERR, and HELLO all represent different packet types. The flow directions are represented as Send, Frwd, Recv and Drop. Each feature is represented by a packet type and a flow direction. For instance, DataFrwd denotes the feature of data packet of forwarded data packets)*

From Table 3, we can see that the first feature set has the highest MDL score. According to the algorithm, we choose it to be our final result. Now we analyze the results: All five testing results include a DATA packet feature, which makes sense because data packet is relatively independent from routing packets. Only the fourth run has ROUTE_REPLY feature, other runs have no features related ROUTE_REPLY. This is because Route reply packets are all triggered by ROUTE_REQUEST packets.

We use Weka [8] to verify the selected feature set. Two learning algorithms, the decision tree and Bayesian network, are used here. For each learning algorithm, two classifiers are trained. One uses the data set with all attributes and the other uses the data set with only the four selected attributes. Table 4 lists the result for the verification. Each line corresponding to a different data set for node 0. For example, line 3 represents the results for data set 3-0.

Table 5: Verification of Feature Selection

The node that the simulation data were collected	Accuracy of the classifier using decision tree with all features	Accuracy of the classifier using decision tree with only four selected features	Accuracy of the classifier using Bayes Net with all features	Accuracy of the classifier using Bayes Net with only four selected features
1	96.3235 %	94.8529 %	99.2647 %	98.5294%
2	97.3684 %	96.3158 %	92.6316 %	96.3158 %
3	98.5714 %	98.5714 %	98.5714 %	98.5714 %

4	100 %	100 %	100%	100 %
5	97.3684 %	94.2105 %	99.4737 %	94.2105 %
6	97.861 %	89.8396 %	91.9786 %	94.6524 %
7	100 %	96.6851 %	98.3425 %	96.6851 %
8	99.4764 %	99.4764 %	99.4764 %	99.4764 %
9	99.4048 %	99.4048 %	99.4048 %	99.4048 %

Generally, the performance of the selected features does not degrade a lot. Sometimes the Bayes net classifier using the selected features even outperforms the Bayes net classifier using all features, such as the node 2 and 6. It seems that the selected features are not quite suitable for the decision tree classifier in some case, such as the node 6.

5 Conclusions and Future Work

In this paper, we described a profile based neighbor monitoring intrusion detection approach in MANET. This approach is simple but it requires monitoring many features. We apply Markov Blanket Discovery in our feature selection and found that the algorithm is accurate by verifying it in Weka.

The project left many future works. First, it takes around one hour in a PC equipped with a Pentium4 2.8 GHz CPU and 1 GB memory to obtain the best Bayesian network structure from the simulation data which contains only 25 attributes and around 200 instances. The bottleneck of the performance is the calculation of MDL score. Before the MDL score can be calculated, the conditional probability table should be constructed first. The code of the conditional probability table construction probably can be further optimized to improve the performance.

Currently, we use hill-climbing algorithm to discover the best Bayesian network structure. We run the hill-climbing algorithm several times and select the network structure that has the maximum MDL score among all the output local maxima. Simulated annealing with a temperature schedule can be implemented in the future to help us to find the global maxima.

The Markov Blanket also can be discovered using the constraint-based approach, such as Grow-Shrink Markov blanket algorithm [6][7]. The constraint-based approaches probably are more efficient than the score-based approach, because the score-based approaches spend too much time on the construction of conditional probability table. Suppose the current node has 10 parents and each parent takes 5 discrete values. The conditional probability table for this node is difficult to compute. Although, the MDL score might punish the complex structure, the programmer still needs to be very careful to avoid such computations. On the contrary, the constraint-based approach can be applied to a large scale in practice [7]. However, due to the time limitation of this project and what we have learned from the class so far, the score-based approach seems much easier for us to implement. In the future, perhaps we can incorporate the constraint-based approach into the scheme.

Another extension is to do feature selection on all network features which include both static features and traffic related features for intrusion detection. I have not seen any research work done in this area yet.

References

[1] R. R. Bouckaert. Probabilistic network construction using the minimum descriptions length principle. *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, 1993
[2] Yongguang Zhang, Wenke Lee, and Yian Huang. *Intrusion Detection Techniques for Mobile Wireless Network*, ACM/Kluwer Wireless Networks Journal (ACM WINET), Vol. 9, No. 5 (September 2003)
[3] H Yang, H Y. Luo, F Ye, S W. Lu, and L Zhang, *Security in mobile ad hoc networks: Challenges and solutions*, IEEE Wireless Communications. 11 (1), pp. 38-47. 2004
[4] Yi-an Huang, Wei Fan, Wenke Lee, and Philip S. Yu, *Cross-Feature Analysis for Detecting Ad-Hoc Routing Anomalies*, In Proceedings of The 23rd International Conference on Distributed Computing Systems (ICDCS)**,** Providence, RI, May 2003.
[5] Yian Huang and Wenke Lee, *A Cooperative Intrusion Detection System for Ad Hoc Networks*, In Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03)**,** Fairfax VA, October 2003.
[6] D. Margaritis. Learning Bayesian network model structure from data. PHD thesis, Carnegie Mellon University, 2003
[7]S. Yaramakala. Fast Markov blanket discovery, Msc thesis, IOWA State University, 2004
[8] Weka http://www.cs.waikato.ac.nz/ml/weka/.

[9] ns2 network simulation tool http://www.isi.edu/nsnam/ns/