A Framework of Machine Learning Based Intrusion Detection for Wireless Sensor Networks

Zhenwei Yu

World Evolved Services, LLC, New York

Jeffrey J.P. Tsai

Department of Computer Science, University of Illinois at Chicago

tsai@cs.uic.edu

Abstract

Some security protocols or mechanisms have been designed for wireless sensor networks (WSNs). However, an intrusion detection system (IDS) should always be deployed on security critical applications to defense in depth. Due to the resource constraints, the intrusion detection system for traditional network cannot be used directly in WSNs. Several schemes have been proposed to detect intrusions in wireless sensor networks. But most of them aim on some specific attacks (e.g. selective forwarding) or attacks on particular layers, such as media access layer or routing layer. In this paper, we present a framework of machine learning based intrusion detection system for wireless sensor networks. Our system will not be limited on particular attacks, while machine learning algorithm helps to build detection model from training data automatically, which will save human labor from writing signature of attacks or specifying the normal behavior of a sensor node.

1. Introduction

A wireless sensor network (WSN) consists of a large set of tiny sensor nodes. Sensor nodes can perform sensing, data processing and communicating but with limited power, computational capacities, small memory size and low bandwidth. The senor nodes in WSNs are usually static after deployment, and communicate mainly through broadcast instead of point-to-point communication. Sensor networks have been used in a variety of domains, such as military sensing in battlefield, perimeter defense on critical area such as airport, intrusion detection for traditional communication network, disasters monitoring, home healthcare and so on. Obviously, some applications are security critical (e.g. military sensing in battlefield), which attract many researchers' attention to secure a sensor network. Some security protocols or mechanisms have been designed for sensor networks. For example, SPINS (Sensor Protocol for Information via Negotiation), a set of protocols, provides secure data confidentiality, two-party data authentication, and data freshness and authenticated broadcast for sensor network [1]. LEAP (Localized Encryption and Authentication Protocol), is designed to support in-network processing bases on the different security requirements for different types of messages exchange [2]. INSENS is an intrusion tolerant routing protocol for wireless sensor networks [3]. A lightweight security protocol relying solely on broadcasts of end-to-end encrypted packets was reported in [4]. However, a sensor network, as a complicated system, there are always some vulnerabilities to be attacked. Moreover, WSNs may be deployed in hostile environments such as battlefields, where sensor nodes are susceptible to physical capture. Security sensitive information (e.g. shared key) might be exposed by compromised nodes. A subvert node might be rejoined the sensor network to perform further attacks. So a particular intrusion detection system for the sensor network is desirable for those security critical applications.

Several schemes have been proposed to detect intrusions in wireless sensor networks [5]-[13]. However, most of them aim on some specific attacks (e.g. selective forwarding [6],[7]) or attacks on particular layers, such as routing layer [9] or media access layer [10]. In this paper, we present a framework of machine learning based intrusion detection system for wireless sensor networks. Our system will not be limited to particular attacks, while machine learning algorithm helps to build detection model from training data automatically, which will save human labor from creating signature of attacks or specifying the normal behavior of a sensor node.

The second section will give the overview of the challenges on intrusion detection in WSNs. Then our framework of IDS will be discussed in section 3. The related works are reviewed on section 4. The paper ends with conclusion and future work in section 5.

2. Challenges on Intrusion Detection in WSNs

To understand the challenges of intrusion detection in WSNs better, we will give the overview

978-0-7695-3158-8/08 $25.00 © 2008 IEEE
DOI 10.1109/SUTC.2008.39

of a sensor node first. A well-known sensor node is the MICA2/MICAz series by Crossbow. The processor in these nodes is an 8 MHz 8-bit Atmel ATMEGA128L CPU. It has only 128 kB of instruction memory, 4 kB of RAM for data, and 512 kB of flash memory [14]. The sensor node is usually powered by 2 AA batteries. MICA2 series sensor nodes feature a multi-channel radio delivering up to 38.4 kbps data rate with a range of up to 1000 feet [15]. MICAz series sensor nodes offer 250Kpbs data rate with a range of up to 100 meters [16]. Security properties or the challenges of the wireless sensor networks have been reported in varied literatures [17]-[20]. In the following, we summarize the challenges on designing IDS in WSN from the limited resources of the sensor nodes, the wireless communication, the dynamic topology of the network and the hostile working environment.

Sensor network nodes usually have severely constraints in computational power, memory size, and energy as we see on MICA2/MICAz series nodes. With those limited resources, some effective security defense techniques for traditional LAN/WAN/Internet are no longer suitable for wireless sensor network. For example, asymmetric cryptography is often too expensive for many WSN applications. Intrusion detection, as another layer of security, plays a more important role to secure wireless sensor networks. However, the low computational power and the insufficient available memory pose big challenges to the design of an intrusion detection system for WSNs: the intrusion detection components should optimize resource consumption, and it might sacrifice its performance to fit the resource constraints. Another challenge is only limited log/audit data could be used for intrusion detection due to low available storage.

Sensor nodes use wireless communication in WSNs. Any information over the radio can be intercepted and the private or sensitive information could be captured by a passive attacker. An aggressive attacker can easily inject malicious messages into the wireless network to perform varied attacks. Unlike wireless local area networks (LANs), whose available bandwidths could be 54Mbps, the data rate for WSN is likely far less than 1Mbps. The low bandwidth prevents some analysis on suspicious data being executed promptly in the powerful remote base station. In other hand, communication is a very energy-hungry task in sensor node, transmitting with maximal power could consume about 3~4 times power as processor does in active mode. Most of communication ability should be reserved for target sensed information. Only limited amount of security related data could be sent to powerful base station for further comprehensive analysis to detect intrusions.

Knowledge of the network is very useful information to detect intrusions. In a wireless sensor network, the topology of the network is usually not a priori. Even after the deployment, the network is always evolving due to frequent failure of sensor nodes, new added sensor nodes. It could be a big challenge to build a base profile in such a dynamic network for an intrusion detection system.

WSNs may be deployed in hostile environments such as battlefields, where sensor nodes are susceptible to physical capture. Security information (e.g. shared key) might be exposed by compromised nodes. The development of tamper-proof nodes is one possible approach to security in hostile environment, but the complicated hardware and high cost keep it away from WSN applications. An intrusion detection system for WSNs has to be aware of physical attacks and can not trust any node.

3. Our Framework of Machine Learning based ID for WSNs

3.1. Architecture

For the traditional wired network, four architecture of intrusion detection system were studied. Centralized network intrusion detection systems are characterized by distributed audit collection and centralized analysis. A hierarchical NIDS has some intermediate components between the collection and analysis components to form a tree structure hierarchy. The intermediate components aggregate, abstract and reduce the collected data and output the results to analysis. A netted architecture permits information flow from any node to any other node. The collection, aggregation and analysis components are combined into a single component that is residing on every monitored system. In a mobile agent based intrusion detection system, all of collection, aggregation and analysis components are wrapped by mobile agent. The code can be migrated to a destination instead of passing massive audit data to reduce the network traffic. Although a centralized detection algorithm was proposed to detect *sinkhole/selective forwarding* attack in wireless sensor networks in [6]. However, the centralized architecture is not suitable for an intrusion detection system to detect as many types of attacks as possible, because the low data rate of wireless communication and limited energy of the sensor nodes couldn't afford to pass the massive audit data to a base station to be analyzed. In other hand, the codes in the sensor nodes are usually written in its ROM before the sensor network is deployed. There is no feasible solution to support transmitting and executing code dynamically which is required by the mobile agent based architecture. A hierarchical architecture of IDS was suggested in [5], where the local agent monitors the node local activity to detect intrusions and the global agent monitors the packets sent by its all neighbors to detect attacks. The local agents run on every sensor nodes while the global agents run on selected nodes, such as the cluster head in applications deploying hierarchical routing protocols, or some watchdog nodes. A decentralized high-level rule-based IDS model was proposed in [11]. Like in the netted architecture, all IDS functions, from data acquiescing to analyzing, are implemented in monitor nodes. However, unlike in the netted architecture, only selected sensor nodes act as monitor nodes and only intrusion alerts are sent to base station.

In our intrusion detection system, the architecture is similar with the netted architecture, where every sensor node will be equipped an intrusion detection agent (IDA). But no cooperation exists between two IDAs since no node can be trusted. Like the attacks against the traditional wired networks, the attacks in wireless sensor network could be inside attacks or outside attacks. The outside attacks could come from more powerful adversary nodes like laptop, while the inside attacks might be launched by compromised sensor nodes that have the legitimate access to the sensor network. Sensor networks are application-oriented, the codes in the sensor nodes are written in its ROM before the sensor network is deployed. An adversary can physically capture a sensor node from a sensor network and reprogram it with extracted security sensitive data (such as id, key) and malicious codes. The subverted node could join the senor network to attack the sensor network further as a compromised node. Unlike the traditional wired network, where HIDS (Host based Intrusion Detection System) can analyze the host features to detect whether the host is compromised or misused. We can not expect to design a similar intrusion detection component to report that its host node is compromised or misused because all original codes (including intrusion detection codes) in its ROM could be erased or modified in such a compromised node. However, we could design a similar Local Intrusion Detection Component (LIDC) to analyze local features to detect whether its host node is suffering attacks from other malicious nodes.

One of the goals of intrusion detection is to stop any ongoing attacks if it is possible. Wireless sensor networks mainly rely on wireless broadcast communication with certain effective range, thus it is possible to locate the inside intruder (subverted node) and isolate it from the sensor network. To locate the subverted node, the intrusion detection system must monitor some suspect nodes and identify subverted nodes by monitoring communication activities of neighbor nodes, which is the task of Packet based Intrusion Detection Component (PIDC) in our IDS. However, the density of nodes in wireless sensor networks is usually high. Many WSNs related research work were based on the network where each node had eight or more neighbor nodes. If a PIDC has to monitor communication activities of its all neighbor nodes, it will cost too much its precious energy. Based on the analysis of the LIDC, the IPDC could monitor only one or couple of its neighbor nodes which could be particular suspect nodes. The PIDCs in its watchdog nodes could cooperate together to identify the real subverted node.

The intrusion alerts are sent to the base station, where the user may be able to verify some possible intrusion. For some false alerts, the base station could do some tuning of the intrusion detection model to reduce further false alerts, and pass the tuning result to sensor nodes. The system block diagram of our proposed intrusion detection is shown Figure 1.

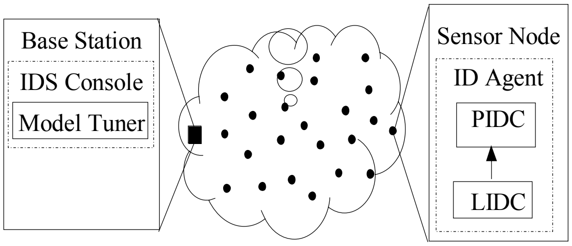

Figure 1. System block diagram

3.2. Audit data for LIDC

In the domain of intrusion detection for traditional wired networks, comprehensive research works have been done on the audit data for intrusion detection. Many system features were identified to be useful for intrusion detection. For example, 41 features were conducted on the network connection in the KDDCup'99 Intrusion Detection dataset [21]. However, due to the resource constraints of a sensor node, there are only few features that could be used to detection intrusion. Moreover, a sensor network will have its own application requirement and employ only the necessary protocols. Thus not all of features identified below could be available for one particular sensor network application.

The sensing component, processor, radio and energy provider are the core parts of a sensor node. Beside the CPU usage, memory usage and changes on storage which have been identified to detect attacks in HIDS for traditional wired networks, more features can be identified related to communication, energy and sensing to detect intrusion in sensor nodes.

3.2.1. **Packet collision ratio.** Packet collision occurs when two or more neighbor sensor nodes try to send packet at the same time through the shared communication channel. Collided packets have to been discarded and retransmitted and waste the constrained energy. Collisions are handled by MAC (Media Access Control) protocol. Scheduled protocols (such as TDMA-based LEACH [22]) are collision free protocols and all transmissions are scheduled on different time/frequency slots. However, adversary nodes could break the schedule intended to attack the sensor network. Contention based protocols (such as CSMA based protocol [23]) allocate the shared channel on-demand and employ some mechanism to avoid the collision but accept some level collisions. A good MAC protocol should archive relative low collision rate when the sensor network works normally, thus abnormal high collision ratio indicates the existence of adversary.

3.2.2. **Packet delivery waiting time.** In contention based MAC protocols, a packet will be buffered to wait for the shared channel. The fairness of accessing shared channel of the MAC protocol will ensure the waiting time of a packet in a reasonable level. The statistic of waiting time could be used to detect some attacks against the fairness of MAC protocol.

3.2.3. **RTS packets rate.** To avoid packet collision, contention based MAC protocols adopt RTS/CTS mechanism. When the channel is idle, the sender is required to send RTS (Request-to-Send) packet to the receiver, and the receiver acknowledges the CTS (Clear-to-Send) packet. The sender starts to send its data after receives the CTS packet from the receiver. This RTS/CTS mechanism could be attacked by sending lots of RTS packets to gain unfair channel or to exhaust the receiver's energy.

3.2.4. **Neighbor count.** Sensor nodes have limited radio transmission range, while the sensor network could be very large. A large sensor network usually employs a multi-hop routing protocol to communication. Each sensor node maintains a neighbor table to record its neighbor information (e.g. nodes id, link cost etc) to build its routing. Unlike the mobile ad-hoc network (MANET), most of nodes in sensor network are supposed to be stationary. The neighbor table should be stable in relative short period, although new nodes could be added and existed node could be removed since fault or energy exhaust over a long time. The change of its neighbor count could be used to detect some attacks. For example, The *Sybil attack* [24] is where a malicious node illegitimately claims multiple identities and works as if it were many nodes.

3.2.5. **Routing cost.** In wireless sensor networks, a multi-hop routing protocol maintains a route table in every node to route its packets. The route table mainly records the next node of the path from one node to base station and its cost, such as hop count or latency. Attacks against routing protocol (such as sinkhole/wormhole) usually broadcast fake routing information to attract more packets to route to its node. Monitoring the routing cost and analyzing its change could be used to detect those attacks.

3.2.6. **Power consumption rate.** Sensor nodes have constrained power. The components of the sensor node, including processing unit, sensing unit and radio, are designed to be powered off to save the energy if it is possible. The node spends most of time in sleep mode to extend the node life. Some proposed energy-aware routing protocols (e.g. SPIN) have access to the current energy level of the node and adapt the protocol it is running based on how much energy is remaining. Some DOS (Denial of Service) attacks aim at the limit power of the sensor nodes. For example, an intruder interferes the transmission to increase the collision ratio or send RTS packets flood to exhaust the victim's energy. The power consumption ratio could be monitored to detect such attacks. Usually sensor node (e.g. MICA2/MICAz nodes) has its own resource manager which keeps track of resource consumption including the power consumption.

3.2.7. **Sensing reading report rate.** Sensing is one of the main functions of sensor nodes. Different applications have different sensing reading report requirement. Some applications require each sensor node report its reading periodically. In these applications, if a sensor node couldn't report its sensing reading following the desired interval, the sensing component could be under attacks. Some other applications require each sensor node reports its reading as the answer of the query from the base station. Subverted node could query the sensing reading more frequently to exhaust the energy of victim nodes.

3.3. Packet features for PIDC

Packet based Intrusion Detection Component (PIDC) analyzes the packets from a suspect node to know whether the suspect node is attacking the host node. The following identified features are calculated on the packets from the same sender (a suspect node).

3.3.1. **Distribution of packet type.** There are several packets to be transmitted over the air in the wireless sensor network, such as sensing data, route update, query/command from the base station, HELLO packets. But the main purpose of a sensor network is to sense certain interesting information, thus the main part of the packets should be sensing data.

3.3.2. **Packet received signal strength**. In wireless transmission, the sender radiates electromagnetic energy into the air through its antenna and the receiver picks up the electromagnetic wave from its surrounding air through its antenna. The received signal strength (RSS) measures the energy of the electromagnetic wave. To receive a packet correctly, the received signal strength must be greater than a threshold known as receiver sensitivity. The received signal strength gradually decreases as the distance between the sender and receiver increases. The distance between the sender and receiver can be estimated according to the RSS and propagation model. The estimated distance could be used to detect the attacker with much powerful radio (such as laptop) compared to the radio of the sensor node. In the other hand, the received signal strength should decrease as the system runs since the energy of the sensor node will be consumed. If the received signal strength increases, it is possible that the node identification was stolen by a powerful malicious node.

3.3.3. **Sensing data arrival rate.** There are two types of sensor network applications according to how the sensor nodes are driven to sense data. In the first applications, the sensor nodes are driven by some particular events. In this type of applications, the sensing data will arrive without any pattern. However, in the second type of applications, the sensor nodes sense the data every preset interval, i.e., driven by the time. In those applications, either missing an expected sensing data or receiving unexpected sensing data identifies some abnormality of the target node.

3.3.4. **Sensing reading value changing ratio.** Sensor networks are mainly used to monitor some environment parameter, such as temperate, sound, wind speed and so on. Some parameters will change within a certain range in a short time. For those applications, if the sensing reading value changes beyond the normal range, there may be some abnormality.

3.3.5. **RTS packets rate.** This feature is calculated on the packets sent from the particular sender, the suspicious node.

3.3.6. **Packet drop ratio.** We have stated that a large sensor network usually employs a multi-hop routing protocol to communication since sensor nodes have very limited radio transmission range. Most sensor nodes also work as a route to forward its received packets. A subverted node could attack this forwarding function by dropping packets or selectively forwarding some packets. To calculate this drop ratio, the host node must know the received packets and the forwarded packets of the suspicious node.

3.3.7. **Packet retransmission rate.** A packet could be retransmitted when the previous transmission is failed due to conflict. However, such retransmission mechanism could be attacked. A subverted node could retransmit a packet multiple times to exhaust the energy of the receiver or try to alter the aggregation value. Abnormal retransmission rate can be used to detect intrusion.

3.4. Detection model and optimization

We would like to apply a machine learning algorithm called SLIPPER [25] to build the detection model. The model will consist of multiple binary classifiers, which includes a set of rules. SLIPPER is a confidence-rated boosting algorithm, and each rule learned from its training dataset might not have very high prediction accuracy on new data. However, the predictions based on the entire set of rules are expected to be highly true. A rule *R* in binary classifier is forced to abstain on all data records not covered by *R*, and predicts with the same confidence C_R on every data record *x* covered by *R*. The confidence C_R was calculated when the rule was built in the training phase. A default rule which covers all data has negative confidence, while all other rules have positive confidence. The binary prediction engine is same as the final hypothesis in SLIPPER [25], which is:

$$H(x) = sign(\sum_{R_t : x \in R_t} C_{R_t}) \quad (1)$$

In other words, the final hypothesis sums up the confidence values of all rules that cover the data and the sign of the sum represents the predicted class label. However, our binary prediction engines will output a signed sum of the confidence values of all rules that cover the data (not just the sign). We refer this signed sum to prediction confidence (*PC*). The magnitude of *PC* represents the confidence of the prediction.

Since the detection model consists of multiple binary classifiers, a final arbiter is needed to pick one of the prediction results from those binary classifiers as its final prediction. The prediction confidence ratio (*PCR*) based arbitral strategy [26] could be used in the final arbiter in our intrusion detection system for wireless sensor network, because the computation required by this arbitral strategy is very light and meet the constrained computational power of sensor nodes. The *PCR* is defined by:

$$PCR = PC \,/\, MAX\ \{PC^1, PC^2, \ldots, PC^m\} \quad (2)$$

Where *PC* stands for prediction confidence on a data record in test dataset while PC^1 stands for the prediction confidence on the i^{th} data record in the training dataset with total *m* records. The prediction confidence ratio based final arbitral strategy can be expressed as follow:

$$i = \{ j \mid PCR_j = MAX\{PCR_1, PCR_2, \ldots, PCR_n\}\} \quad (3)$$

Where PCR_j is prediction confidence ratio and computed by Equation (2), and the *i* is the index of the binary classifier whose prediction result is selected to be the final prediction result. We had built a detection model for traditional network and evaluated it on KDDCup'99 intrusion detection dataset, which was constructed from the raw TCP data for a wired local-area network (LAN) simulating a typical U.S. Air Force LAN. The performance of the detection model on the test dataset was better than the winner of the KDDCup'99 classifier contest [26].

However, the relationships among rules are not explored and rules in the model are disjunctive in default. Therefore, at least one condition in every rule which has multiple conditions has to be evaluated on every data to make the final prediction. In wireless sensor networks, the CPU has limited computational power and the sensor node has constrained energy, so it is desired to optimize the rule evaluation procedure to reduce unnecessary computation. Rules are in IF-THEN form. Most of rules have one or more conjunctive conditions (see rule examples in Figure 2). Each condition consists of a feature name, an operator and a reference value. For example, "Service = telnet", "SourceBytes <= 147". Some conditions in different rules could have same features. To optimize the detection model, we will explore the relationship among those conditions with same features while ignore any possible relationship among different features since we assume the features are independent. Among those conditions with the same features, we realize that two kinds of relationships could be used to optimize the rule evaluation procedure. The first relationship is mutually exclusive relationship among conditions such as "Service = login" and "Service = ftpdata" where these two conditions couldn't be true at the same time. The second relationship is implicit relationship between conditions such as "Duration >= 134" and "Duration >= 67" where the former implies the latter. For conditions with mutually exclusive relationship, at most only one condition could be true, while all other conditions must be false. So when these conditions are evaluated one by one, as long as the true condition is evaluated, the evaluations on remained conditions with mutually exclusive relationship could be skipped. These conditions with mutually exclusive relationship could be ordered further by its possibility to be true if such information is available. For example, the feature “Service” is expected more likely to be “ftpdata” than to be “login”, so the condition “Service = ftpdata” should be evaluated earlier than condition “Service = login”. For conditions with implicit relationship, the implied condition (“Duration >= 67”) should be evaluated only if the implying condition (“Duration >= 134”) is evaluated to be false.

R1: IF Service = login, Duration >= 67.

R2: IF Service = ftpdata, DstBytes <= 5.

R3: IF NumFiles >= 1.

R4: IF Duration >= 134, DstHostErrRate <= 0.

Figure 2. A rule set with four rules

To utilize these relationships to optimize the condition evaluation, we organize conditions in all rules into a tree structure. Each node consists of a condition to be evaluated and three child trees. The left child tree (true child) will be evaluated when its condition is true, while the right child (false child) tree will be evaluated when its condition is false. Of course, there are some rules such that none of its conditions has mutually exclusive or implicit relationship with any condition in other rules. The middle tree (unconditional child) is built on all conditions from those rules, which will be evaluated before its condition is evaluated. For example, we can organize the four rules listed in Figure 2 into a tree structure shown in Figure 3.

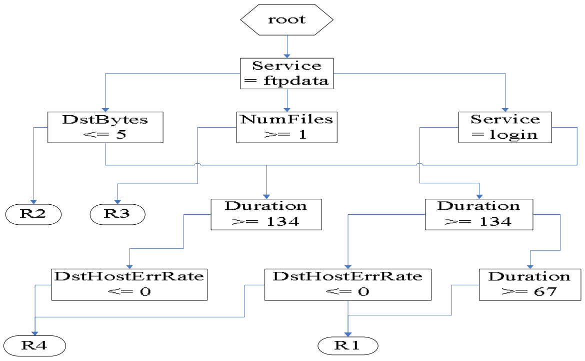

Figure 3. Rule evaluation tree

Each node in the tree structure can be described in text mode by a quad (unconditional child, node condition, true child, false child). For example, the node “NumFiles >= 1” is saved at (,NumFiles>=1, R3,). The structure shown in Figure 3 can be stored in text mode as:

((, NumFiles >= 1 , R3,), Service = ftpdata, ((, Duration >= 134 , (, DstHostErrRate <= 0, R4,),), DstBytes<=5, R2,), (, Service = login , (, Duration >= 134, (R1, DstHostErrRate<=0, R4,), (, Duration >= 67, R1,)), (, Duration >= 134 , (, DstHostErrRate <= 0, R4,),))).

3.5. Model tuning

An intrusion detection system will alert possible intrusions, however the alert could be false. The false alerts are also anticipatable in our system. We have developed a model tuning algorithm [27], which can tune the model (rule’s associate confidence) automatically to improve its performance in the future data. The tuning algorithm utilizes a property of the binary classifier that only rules that cover a data record contribute to the final prediction on this data record. Our tuning algorithm changes the associated confidence values to adjust the contribution of each rule to the binary prediction. Consequentially, tuning ensures that if a data record is covered by a rule in the original model, then it will be covered by this rule also in the tuned model and vice versa.

When a binary classifier is used to predict a new data record, two different types of false predictions may be generated according to the sum of confidence values of all rules that cover this data record. When the sum is positive, the binary classifier predicts the data record to be in the positive class. If this prediction is false, it is treated as a false positive prediction. When the sum is negative, the binary classifier predicts the data record to be in the negative class. If this prediction is false, it is considered a false negative prediction. Obviously, when the classifier makes a false positive (FP) prediction, the confidence values of those positive rules involved should be decreased to avoid the false positive prediction made by these rules on subsequent data. When the classifier makes a false negative (FN) prediction, the confidence values of the positive rules involved should be increased to avoid false negative predictions made by these rules on successive data. Formally,

$$C'_R = \begin{cases} p \cdot C_R & \text{if rule R} \xrightarrow{\text{contributes to}} FP \\ q \cdot C_R & \text{if rule R} \xrightarrow{\text{contributes to}} FN \end{cases} \qquad (4)$$

Where constrains $p < 1$ and $q > 1$ ensure that a positive rule always has a positive confidence. Because $p < 1$, $q > 1$, and the confidence value of the default rule is unchanged, trivially there exists a number n, such that after updating the confidence values n times, the sign of the sum of the confidence values of all rules (both positive rules and the default rule) will be changed. That means the tuned classifier could make a true prediction on the data where the original classifier made a false prediction. Our experiments showed that the system could achieve about 20% improvement with quick tunings while only 1.3% of the false predictions were used to tune the model [27].

Due to the limited computation power of the sensor node, the model tuning function is separated from the detection agent logically and physically. The model tuner with the system console is resident in the base station. To tune the model of a particular sensor node, the system must keep a copy of detection model for every intrusion detection agent. Only the tuned results will be delivery from the base station to save communication cost.

4. Related Work

The general guidelines for IDS in sensor networks were discussed in [5]. A hierarchical architecture of IDS was suggested, where the local agent monitors the node local activity to detect intrusions and the global agent monitors the packets sent by its all neighbors to detect attacks.

A centralized detection algorithm against *sinkhole/selective forwarding* attack was proposed in [6]. The base station identifies a list of suspicious nodes by detecting data inconsistency use a statistical method. Then the base station can estimate the *attack area* where the sinkhole node locates. A requested network data flow message will be sent by the base station to the nodes in *attack area* with the suspicious node IDs. All suspicious nodes will reply this request with its network flow information including its ID, next-hop ID and cost. The network flow information can be represented by a directional edge from source ID to its next-hop ID in a base station. The base station will realize routing pattern by constructing a tree using these direction edges. An area invaded by a sinkhole attack processes special routing pattern where all network traffic flows toward the same destination, the root in the tree of network flow, which is compromised by the intruder.

A specification-based network intrusion detection system principally against the *sinkhole/selective forward* attack was presented in [7]. A rule specifies that a normal node should forward the packets at a rate over a threshold. Otherwise, the node could be abnormal. For a link A->B (Node A sends packets to Node B), Node A and the watchdog nodes of link A->B monitor the behavior of node B and make the decision cooperatively through a majority-vote policy.

To identify an intruder impersonating a legitimate neighbor, a low-complexity anomaly detection algorithm was proposed in [8]. A sensor node recorded the arrival time and received power of each incoming packet for last N packets from each neighbor. A simple dynamic statistical model (the min and max of received power, the packet arrival rate on last N packets and on last N2 packets) was built. The simple statistical model was used to detect any abnormality by monitoring received packet power level and packet arrival rates from a neighbor node.

An unsupervised anomaly detection technique was proposed to detect routing attacks in wireless sensor network in [9]. Total 9 traffic related features based on AODV (Ad hoc On-demand Distance Vector [28]) routing protocol were identified to describe the conditions of the traffic flow through the node. Three non-traffic related features were selected to monitor changes of the path to the base station. The proposed system adopted a fixed-width clustering algorithm, which had been applied for anomaly detection in IP network.

Attacks against on MAC protocol in wireless sensor networks were studied and classified into collision attack, unfairness attack and exhaustion attack in [10]. Three statistics collision ratio, packet waiting time and RTS packet ratio were identified as intrusion indicators respectively. The probability of particular attacks was calculated by a soft decision function along with an overall probability of attacks related to packet successful delivery ratio.

A decentralized high-level rule-based IDS model was proposed in [11]. All IDS functions, from data acquiescing to analyzing, are implemented in monitor nodes. Only intrusion alerts are sent to the base station. Seven high level rules (*interval rule*, *retransmission rule*, *integrity rule*, *delay rule*, *repetition rule*, *radio transmission range rule* and *jamming rule* [11]) were defined to detect intrusions. This IDS performs analysis on data message listened to by the monitor node that is not addressed to it and message collision when the monitor node tries to send a message. After messages are collected in promiscuous mode and the important information is filtered and stored, a sequent rule-matching procedure is executed on every message. The order of rules depends on the message type. When a rule fires on a message, the rule-matching procedure will stop and the message will be discarded to save the storage space. Instead of reporting an alarm on attack, a failure counter is incremented when a rule fires on a message. An attack is alerted only if the counting failure number is greater than an expected value by the monitor node during the analysis of messages transmitted on its neighborhood in a round. This expected number is calculated dynamically by the monitor node according to the failure history for each node in its neighborhood.

The intrusion detection problem in WSN was formulated as a non-cooperative two-player nonzero-sum game between the intrusion detection system and the attacker in [12], [13]. The basis is that in non-cooperative games there exist sets of optimal strategies (so-called Nash equilibrium) used by the players in a game such that no player can benefit by unilaterally changing his or her strategy if the strategies of the other players remain unchanged. The relationship between an attacker and the IDS is non-cooperative in nature because no outside authority could assure any agreement between an attacker and the IDS. The proposed IDS is able to monitor all sensor nodes, but due to system limitations it can only protect one sensor node at each time slot, and based on a game theoretic framework it will choose such a sensor node (called cluster head) for protection.

5. Conclusion and Future Work

In this paper, we presented a framework of a machine learning based intrusion detection for wireless sensor network. In our system, each sensor node will equip a detection agent. The detection agent will analyze the local data and packet data from suspicious node to identify an intruder. When the user found a false alert, the system can automatically tune the model to improve its performance in the future data. In the future, we plan to set up an experiment environment to test our framework.

6. References

[1] A. Perrig, et al., “SPINS: Security Protocols for Sensor Networks”, *Wireless Networks*, 8(5):521- 534, Sep. 2002.

[2] S. Zhu, S. Setia, and S. Jajodia, “LEAP: Efficient Security Mechanisms for Large-scale Distributed Sensor Networks”, *Proc. of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, Oct. 2003.

[3] J. Deng, R. Han, and S. Mishra, “A Performance Evaluation of Intrusion-tolerant Routing in Wireless Sensor Networks”, *Proc. of the 2nd Int. IEEE Workshop on Information Processing in Sensor Networks (IPSN’03)*, Apr. 2003.

[4] J. Undercoffer, et al., “Security for Sensor Networks”, *CADIP Research Symposium*, 2002.

[5] R. Roman, J. Zhou and J. Lopez, “Applying Intrusion Detection Systems to Wireless Sensor Networks”, *Proc. of the 3rd IEEE Consumer Communications and Networking Conference (CCNC 2006)*, Jan. 2006 Vol. 1, pp. 640- 644.

[6] E. Ngai, J. Liu, and M. Lyu. **“**On the Intruder Detection for Sinkhole Attack in Wireless Sensor Networks”, *IEEE International Conference on Communications (ICC'06)*, Istanbul, Turkey, June 11-15, 2006.

[7] K. Ioannis, T. Dimitriou and F. Freiling, “Towards Intrusion Detection in Wireless Sensor Networks”, in *13th European Wireless Conference*, Paris, France, April 2007.

[8] I.Onat, A. Miri, “An Intrusion Detection System for Wireless Sensor Networks”, *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, 2005. (WiMob'2005),* V. 3, Aug. 2005, pp. 253 – 259.

[9] C. E. Loo, M. Y. Ng, C. Leckie and M. Palaniswami, “Intrusion Detection for Routing Attacks in Sensor Networks”, *International Journal of Distributed Sensor Networks*, Vol. 2, No. 4, October-December 2006, pp. 313-332.

[10] Q. Ren; Q. Liang, “Secure Media Access Control (MAC) in wireless sensor networks: intrusion Detections and Countermeasures”, *15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2004)*, Sept. 2004 Volume 4, pp. 3025 – 3029.

[11] A.P. Silva, et al., “Decentralized Intrusion Detection in Wireless Sensor Networks”, *Proc. of the 1st ACM Int. Workshop on Quality of Service and Security in Wireless and Mobile Networks*, 16-23, Oct. 2005.

[12] A. Agah, S.K. Das, S.K. and K. Basu, “A Non-cooperative Game Approach for Intrusion Detection in Sensor Networks”, *IEEE Vehicular Technology Conference (VTC)*, fall 2004.

[13] A. Agah, et al., “Intrusion Detection in Sensor Networks: a non-cooperative game approach”, *the 3rd IEEE Int. Symp. on Network Computing and Applications, (NCA'04)*, 343-346, 2004.

[14] Online ATMEGA128L datasheet, *http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf*, Jun, 2007.

[15] Online MICA2 datasheet, *http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2 _Datasheet.pdf*, Jun, 2007.

[16] Online MICAz datasheet, http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz _Datasheet.pdf, Jun, 2007.

[17] C.Y. Chong and S.P. Kumar, “Sensor Networks: Evolution, Opportunities and Challenges”, *Proc. of the IEEE*, 91(8):1247-1256, Aug. 2003.

[18] E. Shi and A. Perrig, “Designing Secure Sensor Networks”, *IEEE Wireless Communications*, 11(6):38-43, Dec. 2004.

[19] I. F. Akyildiz, et al., “A Survey on Sensor Networks”, *IEEE Communications Magazine*, 40(8):102-114, Aug. 2002.

[20] M. Tubaishat and S. Madria, “Sensor Networks: an overview”, *IEEE Potentials*, 22(2):20-23, Apr. 2003.

[21] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[22] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-efficient Communication Protocols for Wireless Microsensor Networks,” *Proc. of the Hawaii International Conference on Systems Sciences*, Jan. 2000.

[23] A. Woo and D. Culler, “A Transmission Control Scheme for Media Access in Sensor Networks,” *Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking*, Rome, Italy, July 2001, pp. 221–235, ACM.

[24] J.R. Douceur, “The Sybil Attack”, *Proc. of the 1st Int. Workshop on Peer-to-peer Systems (IPTPS ’02)*, Mar. 2002.

[25] W. Cohen and Y. Singer, “A Simple, Fast, and Effective Rule Learner”, *Proc. of 16th national Conference on Artificial Intelligence and 11th Conference on Innovative Applications of Artificial Intelligence*, Orlando, Florida, pp.335-342, July 1999.

[26] Z. Yu and J. Tsai, “An Efficient Intrusion Detection System using Boosting Based Learning Algorithm”, *Int'l Journal of Computer Applications in Technology (IJCAT)*, Vol. 27, No. 4, pp.223–231. 2006.

[27] Z. Yu, J. Tsai and T. Weigert, “An Automatically Tuning Intrusion Detection System”, *IEEE Transactions on Systems, Man, Cybernetics, Part B*, Vol. 37, No. 2, pp.373-384, April 2007.

[28] C. Perkins and E. Royer, “Ad-hoc On-Demand Distance Vector Routing”, *Proc. of the 2nd Workshop on Mobile Computing Systems and Applications (WMCSA'99)*, February 1999, pp. 90-100.