Distributed and Control Theoretic Approach to Intrusion Detection

Rahul Khanna
Intel Corporation, 2111 NE 25th Ave., Hillsboro, OR 97124, USA
rahul.khanna@intel.com

Huaping Liu
Oregon State University, School of EECS, Corvallis, OR 97331 USA
hliu@eecs.oregonstate.edu

ABSTRACT

Ad hoc wireless networks are more vulnerable to malicious attacks than traditional wired networks due to the silent nature of these attacks and the inability of the conventional intrusion detection systems (IDS) to detect them. These attacks operate under the threshold boundaries during an intrusion attempt and can only be identified by profiling the complete system activity in relation to a normal behavior. In this paper we discuss a control-theoretic Hidden Markov Model (HMM) strategy for intrusion detection using distributed observations across multiple nodes. This model consists of a distributed HMM engine that executes in a randomly selected monitor node and functions as a part of the feedback control engine. This drives the defensive response based on hysteresis to reduce the frequency of false positives, thereby avoiding inappropriate ad hoc responses.

Categories and Subject Descriptors: G.3 [Probability and statistics]: Multivariate statistics, Stochastic processes

General Terms: Security, Algorithms.

Keywords: IDS, Intrusion Detection, Hidden Markov Models, Wireless Ad-Hoc Networks.

1. INTRODUCTION

Intrusion detection system (IDS) protects data integrity and manages the system availability during intrusion. In a mobile ad hoc network (MANET) with self-regulating properties [1] it deals with challenges related to resource-constrained, fully-mobile, self configuring, multi-hop wireless networks with varying resources and limited bandwidth. Distributed and cooperative nature of ad hoc network nodes enables a malicious node to exploit the weakest node by hijacking or launching an attack through it. This inherent vulnerability can disable the whole network cluster and further compromise the security by impersonating, message contamination, hijacking, passive listening, or acting as a malicious router. Some of the common attacks that exploit these limitations are route messages and route information tampering, selective forwarding, sybil attack, sinkhole attack, wormhole attack, spoofing, packet flooding, packet-dropping, location exposure, sleep deprivation (battery exhaustion), and radio jamming (MAC layer attack). Adding to the problem, constantly changing topologies and volatile physical environments make it difficult to discriminate between an intrusion and a normal operation. Various routing techniques have been researched in this area that tries to resist attacks [2]. Intrusion is a pattern of an observed sequence. Its detection is similar to an immune system that identifies and eliminates anomalies by measuring deviations from the normal processes using distributed identifiers over the system with identifiable and adaptable relationship. This can be supported using a model where each state has probabilistic distribution of producing identifiable observations and transition matrix to other states.

Hidden Markov model (HMM) [3] is one such model that correlates observations (parameters changes, fault frequency, etc.) [4] to predict hidden states that factor in the system design. Observation points are optimized using an acceptable set of system-wide intrusion checkpoints (IC) while hidden states are created using explicit knowledge of probabilistic relationships with these observations. For modeling a large number of temporal sequences, HMM acts as an excellent alternative, as it has been widely used for pattern matching in speech recognition and image identification. Some of the previous work on IDS using HMM includes an HMM-based predictive model capable of discriminating between normal and abnormal behaviors of network traffic [5], a framework for handling multiple sensors implemented by representing each of the sensors monitoring a host with an HMM [6], HMM-based detection of complex Internet attacks consisting of several steps that occur over an extended period of time [7], HMM based anomaly decisions at system call level using sequences of system calls trace as observable [8], and HMM-based algorithms for building behavior classifiers capable of detecting intrusion attempts on computer systems [9]. Other work in this area includes a statistical approach [10] which monitors the system call trace of a program's execution for compliance to the precomputed model, and alert-based policy mechanism [11] that associates an alert with multiple events frequently occurring together.

This paper investigates the problem of intrusion detection while reducing the number of false positives in a power friendly manner. It extends the traditional HMM-based IDS approach by using a *control-theoretic, distributed HMM* to make it suitable for ad hoc networks with limited power and processing capabilities. The control theoretic component is the *proportional integral differential (PID) control engine* that drives the defensive response based on feature hysteresis to reduce the frequency of false positives. These controllers do not require advanced mathematics to characterize the model underlying the checkpoint measurements and can be easily be implemented as silicon hooks coupled to the monitored intrusion checkpoints with an adjustable response. Distributed HMM processing distributes the computational load of training and state estimation by choosing a Monitor Nodes among member nodes of the ad hoc node cluster using a voting mechanism (Sec. 4.3).

Section 2 presents the Intrusion Checkpoint Control Stage (ICCS) that is the observability stage with an objective to produce stable emissions using continuous estimations. It adapts the checkpoint

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
IWCMC'07, August 12–16, 2007, Honolulu, Hawaii, USA.
Copyright 2007 ACM 978-1-59593-695-0/07/0008...$5.00.

trigger based on the weighted sum of proportional, average and derivative sensor measurements over derivative and integral time window. This stage is also responsible for detecting temporary changes due to legal activity and concept drift signifying changing long-term user behaviors to avoid falsely predicting an attack situation. Observation can be rejected as a noise, or classified to a valid state based on the trending, similarity between un-classified states tending toward certain classification, and feedback from state machine based on other independent observations. Section 3 discusses the Intrusion State Detection Stage (ISDS) that receives the observability data from multiple checkpoints and predicts the transition to one of the hidden states (normal, intrusion) based on trained statistical model (Section 3.1). Estimated intrusion decision is fed back to ICCS which helps re-estimating the usage trends while avoiding any false positive preemptive responses. Section 4 discusses the *Monitor Node Selection* that optimizes the computational load of HMM processing between nodes.

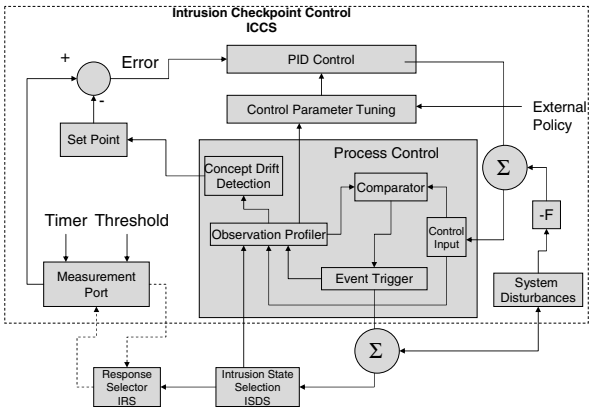

Figure 1: ICCS is responsible for providing the stable observability data to the intrusion state detection stage. This data is profiled for variances due to changing user behavior and temporary changes in system environment (also referred to as disturbances).

2. INTRUSION CHECKPOINT CONTROL STAGE (ICCS)

In this section we define checkpoint control components of the IDS that cooperate with each other to enact an observation. In ad hoc networks, an IDS is deployed at the nodes to detect the signs of intrusion locally and independent of other nodes, instead of routers, gateways or firewalls. The IDS architecture consists of multiple stages with information feedback mechanism between stages. ICCS represents the feedback control component for an individual intrusion checkpoint. It consists of measurement port, PID controller, observation profiler, concept drift detector (CDD), and feedback path to the process input.

(1) **Measurement Port** consists of fast-acting software and silicon hooks that are capable of identifying, counting, thresholding, time-stamping, eventing, and clearing an activity. Examples of such hooks are performance counters, flip counters (or transaction counters), header sniffers, fault alerts (page faults etc.), bandwidth usage monitors, session activity, system call usage between various processes and applications, file-system usage, and swap-in/swap-out usage. Measured data is analyzed as it is collected or afterwards to provide real-time alert notification for suspected intrusive behaviors. These fast-acting hooks are clustered to enact an observation. Measurements can be sampled at regular intervals or cause an alert based on a user-settable threshold and can be classified as:

(a) *Resource activity trend* that is the measure of a resource activity monitored over a larger sampling period and has characteristics that repeat over that sampling period.

(b) *Event interval* that is a measure of an interval between two successive activities.

(c) *Event trend* that is the measure of events monitored over a large sampling period with an objective to calculate the event behavior with a built-in repeatability.

(2) **Observation Profiler** monitors various inputs for maintaining & re-estimating activity profile that ascertains rough (partially perfect) boundary between normal and abnormal activity. It is characterized in terms of a statistical metric and model, where a metric represents a quantitative measure accumulated over a period. Measurements obtained from the audit records in this statistical model analyzes any deviation from a standard profile. Observation profiler receives multiple feedback from PID control output, event trigger and ISDS (Sec. 3), and performs recursive estimations to generate successive probabilistic profile data estimates with closed-form solution. Activity profile data consists of probability distribution function (pdf) parameters represented by $\lambda_j = (\sigma_j, \mu_j, \eta_j)$, where σ_j, μ_j, and η_j represents variance, mean, and activity drift factor, respectively. Successive observations are evaluated against this profile which results in its new profiles and drift detection. An observation (emission) can also be a set of co-related measurements but represented by a single probability distribution function. Each of these measurements carries different weights as in multivariate probability distribution. Such relationship is incorporated into the profile for the completeness of the observation and reduces the dimensionality for effective runtime handling.

(3) **Concept Drift Detector** detects and analyzes the concept drifting [12] in the profile where training data set alone is not sufficient, and the model (profile) needs to be updated continually. For Example, An instantaneous deviation from a normal profile can be construed as an intrusion due to a momentary change in the system environment. Such deviations may be legal as also seen during installation of new patches in an operating systems. When there is a time-evolving concept drift, using old data unselectively helps if the new concept and old concept still have consistencies and the amount of old data chosen arbitrarily just happen to be right [13]. This requires an efficient approach to data mining that helps select a combination of new and old (historical) data to make an accurate re-profiling and further classification. The mechanism used is the measurement of *Kullback-Leibler (KL) divergence* [14], or relative entropy measures the kernel distance between two probability distributions of generative models. KL divergence is also the gain in Shannon information involved in going from the *a priori* to the *posteriori* expressed as:

$$\alpha_t = KL(b(v|\theta'_t), b(v|\theta_t)) \tag{1}$$

where α_t is KL divergence measure, θ'_t is new Gaussian component, and θ_t is old Gaussian component at time t.

We can evaluate divergence by a Monte Carlo simulation using the law of large numbers [15] that draws an observation v_i from the estimated Gaussian component θ'_t, computes the log-ratio and averages this over M samples as

$$\alpha_t \approx \frac{1}{M} \sum_{i=1}^{M} \log \left(\frac{b(v_i|\theta'_t)}{b(v_i|\theta_t)} \right). \tag{2}$$

KL divergence data calculated in the temporal domain are used to evaluate the speed of the drift, also called drift factor $0 \leq \eta \leq 1$. These data are then used to assign weights to the historical parameters that are then used for re-profiling.

(4) **Feedback Path** is responsible for feeding back the current state information to the profile estimator. The current state information is calculated by running the ISDS module using the current model parameters. This information is then used by the profiler to filter out any noise and re-estimate the activity profile data. If a trigger activity is not followed by a state transition, then a corrective action is performed to minimize the false positives in the future.

(5) **PID Controller** generates an output that initiates a corrective response applied to a process in order to drive a measurable process variable toward a reference value (set point). It is assumed that any intrusion activity will cause variations in the checkpoint activity, thereby causing a large error. Errors occur when a disturbance (intrusion) or a load on the process (changes in environment) changes the process variable. The controller's mission is to eliminate the error automatically. Discrete form of PID control is represented as:

$$u(nT) = K_p e(nT) + K_i T \sum_{i=(nT-w)}^{nT} e(i) + K_d \frac{e(nT) - e(nT-1)}{T} + u_0 \quad (3)$$

where $e(t)$ is the error represented by difference between measured value and set-point, w is the integral sampling window, nT is the n-th sampling period, and K_p, K_i, and K_d are the proportional, integral, and derivative gains, respectively.

Stability is ensured using the proportional term, the integral term permits the rejection of a step disturbance and the derivative term is used to provide damping or shaping of the response. The desired closed-loop dynamics are obtained by adjusting these parameters iteratively by *tuning* and without specific knowledge of an intrusion detection model. Control parameters are continuously tuned to ensure the stability of the control loop in a control-theoretic sense, over a wide range of variations in the checkpoint measurements. While control parameters are evaluated frequently, they are updated only when improvement in stability is anticipated. These updates can be periodic over a large period of time.

2.1 Intrusion Checkpoint Control

Control-theoretic architecture (Fig. 2) drives the defensive response based on hysteresis to reduce the frequency of false positives, thereby avoiding inappropriate ad hoc responses. Excessive responses related to adjusting component functionality (e.g., throttling), alert generation (to predict intrusion state) and analyzing concept drift can slow down the system and negatively impact the effectiveness of the IDS. Alternately, an appropriate response can predict the attack pattern and trigger the selective response using a PID controller that takes a measured value from an intrusion checkpoint and compares it with a reference value. The difference is then used to trigger alert (abnormal activity) to the process in order to bring the process' measured value back to its desired set-point. PID controller can adjust the process outputs based on the history and rate of change of the error signal, which gives more accurate and stable control. This avoids inaccurate representation of intrusion activity due to false alarms or miss detections that can result in either disproportionate and costly defensive measures or complete security failure. It is therefore essential to build weighted integral and differential response to the trigger mechanism instead of reacting to an instantaneous measurement. While integral response measures the amount of time the error has continued uncorrected, differential response anticipates the future errors from the rate of change of error over a period of time. The reference (set-points) values are dynamic in nature and set as a part of coarse grain settings that are estimated over long periods of time. These re-estimates are required to account for the changing user behavior, also referred as *concept drift*.

Checkpoint control loop forms the first stage of multi-stage intrusion detection system of a sequential IDS where process output provides the observability of an individual intrusion checkpoint to aid in the state estimation. Collective observations from multiple checkpoints are fed into the statistical model (in this case HMM) responsible for predicting the state transition. Response measures are delayed to account for delay involved in estimation of intrusion state based on observations from other checkpoints. System policy is driven by long-term hysteresis based on the system's behavior and the well-known relationship with various checkpoints. While the set-point (reference) is constant over a long period of time, it can change due to user behavior of system policy driven by a temporary change in the operating environment.

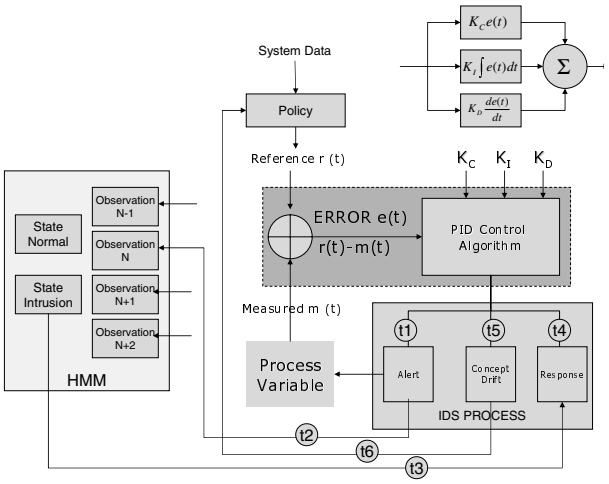

Figure 2: PID control loop for intrusion checkpoint. The Process output (alert) constitutes the observation (emission) in an HMM. A true-positive response is fed back to the process response unit of the PID control to aid runtime re-training. Concept drift analysis aids in re-setting the reference point.

3. INTRUSION STATE DETECTION STAGE (ISDS)

ISDS defines the statistical model responsible for predicting the current intrusion state based on observability data received from ICCS modules. We choose HMM (Section 3.1) model where states are hidden and indirectly evaluated based on model parameters.

3.1 Hidden Markov Model

Since anomaly can be treated as a classification problem, stochastic approach like HMM can very well be used in intrusion detection. An HMM is a stochastic model of discrete events and a variation of the Markov chain consisting of a set of discrete states and a matrix $\boldsymbol{A} = \{a_{ij}\}$ of *state transition probabilities*. The model consist of *observed (intrusion checkpoints) states*, *hidden (intrusion) states*, and *HMM (activity) profiles*. HMM training using initial data and continuous re-estimation creates profile that consists of transition probabilities and observation symbol probabilities. HMM modeling involves:

(1) Measuring *observed states* that are test-points spread all over the system representing competing risks derived analytically or logically using intrusion checkpoint indicators based on correlations among two or more metrics.

(2) Estimating *instantaneous observation* probability matrix that indicates the probability of an observation, given a hidden state $p(S_i|O_i)$. This density function can be estimated using explicit

parametric model (multivariate Gaussian) or implicitly from data via non-parametric methods.
(3) Estimating *hidden states* by clustering the homogeneous behavior of single or multiple components together that are indicative of various intrusion activities that need to be identified. Hidden states $\boldsymbol{S} = \{\boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_{N-1}, \boldsymbol{S}_N\}$ are the set of states that are not visible but each state randomly generates a mixture of the M observations (or visible states O). The IDS has the following states:

1. *Normal* (N) state indicates the profile compliance.

2. *Intrusion in progress* (IP) indicates an intrusion activity that is setting itself up. This includes attempt to gain privileged resources, acceleration in resource usage, etc.

3. *Intrusion successful* (IS) indicates a successful intrusion. A successful intrusion will be accompanied with unusual resource usage (CPU, memory, IO activity, etc.).

(4) Estimating *Hidden (intrusion) state transition* probability matrix using prior knowledge or random data and long-term temporal characteristics.

3.2 ISDS Architecture

ICCS triggered output acts as an emission to a specific HMM model and allocates a weight according to their confidence. Observation probability is expressed as a mixture of individual observation probabilities from multiple checkpoints, measured as fractions of a total, to improve the performance of IDS. The weights are given to each model based on trivial knowledge and continuous training. The mixture model can be represented as:

$$p(x) = \sum_{k=1}^{K} a_k h(x|\lambda_k) \tag{4}$$

where $p(x)$ is the modeled probability distribution function, K is the number of components in the mixture model, and a_k is mixture proportion of component k.

This allows to model the intrusion states at varying degree of granularity while retaining the advantages of each model. Based on data characteristics (amount of data, frequency), models are adapted by modifying weights such that complex models are favored for complex inputs and vice versa.

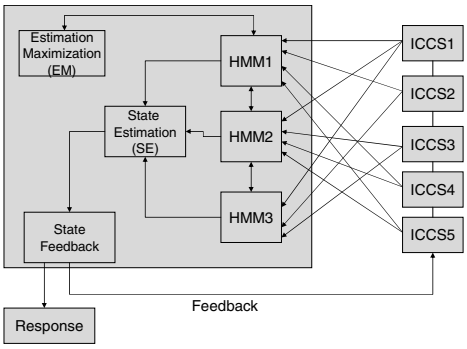

Figure 3: Intrusion state detection stage.

HMMx sub-block (Fig. 3) receives the *abnormal activity* alert and processes the interrupt to service the hidden-state (intrusion) estimation. It maintains the *HMM data* and interacts with the expectation maximization (EM) block and the state-estimation (SE) block for re-training and state-prediction flows. This block also implements reduced-dimensionality by combining multiple inputs into a single observation with its own probability distribution function. This observation is then fed into the EM and SE blocks for state estimation.

In the HMM mixture modeling, intrusion checkpoint events under consideration have membership in one of the distributions we are using to model the data. This requires an estimation to devise appropriate parameters for the model functions we choose, with the connection to the data points being represented as their membership in the individual model distributions. The EM algorithm sub-block [16] provides the mechanism to the problem of maximum likelihood (ML) parameter estimation process that yields a parameter set used to assign observations points to new states. It estimates the ML estimates of parameters in the HMM model as well as mixture densities (or model weights) and relies on the intermediate variables (also called latent data) represented by state sequence. EM alternates between performing an E-step, which computes an expectation of the likelihood, and an M-step, which computes the ML estimates of the parameters by maximizing the expected likelihood found on the E-step.

SE sub-block models the underlying state and observation sequence of HMM mixture to predict state sequences for new intrusion states using the Viterbi algorithm. The Viterbi algorithm is a dynamic algorithm requiring time $O(TS^2)$ (T is time steps count and S is the number of states) where at each time step it computes the most probable path for each state given that the most probable path for all previous time steps has been computed. Trained mixture appears to be a single HMM for all purposes and applied as a standard HMM algorithm to extract the most probable state sequence, given a set of observations. Estimates for the transition and emission probabilities are based on multiple HMM models and are transparent to the standard HMM models. The state feedback sub-block feeds-back the estimated state to the *observation profiler* in ICCS (Fig. 1), and uses it for profile re-calibration.

As a part of the proactive approach in an active IDS, the response unit encapsulates various actions that are undertaken upon a suspected intrusion. It modifies the state of the attacked system to thwart or mitigate the effects of the attack. Such control can take the form of terminating network connections, increasing the security logging, killing errant processes, APR poisoning, using decoys (false IP address), etc. This action is also important because after raising the abnormal activity alert, profiler (ICCS) constantly monitors the abnormal activity (PID control output) and expects it to reduce based on some external actions. This action is equivalent to the process control function that influences the process variable in the feedback control system with an objective to reduce the abnormal activity. This requires a complete understanding of active intrusion responses which is still an open problem. An over-reactive response can turn into a denial of service (DoS) attack.

4. INTRUSION DETECTION NODES

In this section we will discuss cooperative IDS that involves participation of the member nodes in the global decision process. This involves distributed processing among local nodes and randomly elected monitoring nodes. While ICCS is implemented locally using silicon and software hooks, ISDS operations execute on the monitoring nodes (Sec. 4.2). *Monitoring nodes* are at a single-hop distance and elected randomly at periodic intervals using a fairness and risk cost evaluation. Various factors such as the number of refusals, membership period, and voting patterns are considered for making such evaluation. While local nodes contribute the trigger data locally and externally, monitor nodes consume this data to estimate the intrusion state through the contribution of observations from all member nodes. Whenever a suspected activity is detected, it initiates an intrusion detection event that is propagated to the monitor nodes. Monitor nodes in turn request for the sharable observation data from individual nodes. Based on multi-

ple observations with node-level dimensionality, an HMM mixture algorithm is executed to predict the possible intrusion state.

4.1 IDS Node

Intrusion detection in mobile local hosts is limited to profiling it's local activity using floating ICCS modules. The intent is to reduce the system complexity and the possibility of software reuse. These hooks are presented to accelerate the combined measurements of the clustered components with an ability to send alerts based on a systems-level policy. It contains the hardware and software that act as a glue between transducers and a control program that is capable of measuring the event interval and event trend with an ability to generate alerts on deviation from normal behaviors (represented by system policy). In this specific case, the feedback control loop is implemented partially in the silicon (ICCS block) with configurable control parameters (see Fig. 1). To further enhance the auto-discoverability, modularity, and re-usability, configuration and status registers are mapped into the capability pointer of the PCI express configuration space. Similar mechanisms exist today in the very basic form as performance counters (PerfMon), leaky-bucket counters, etc. These counters need to be coupled with ICCS modules that contain PID Controller, profilers, threshold detectors, drift detectors, and coarse-grain tuners. ICCS modules are implemented in isolation from the measured components such that a single ICCS component can multiplex between multiple measurement modules. While some of the checkpoints are used for local consumption, others are shared with the monitor nodes to aid in cooperative state estimation. Examples of such checkpoints are *packet drop rate*, *route request rate*, and *route reply rate*. These checkpoints share the trigger data with the monitors nodes and contribute as the node's contribution to the mixture of HMMs.

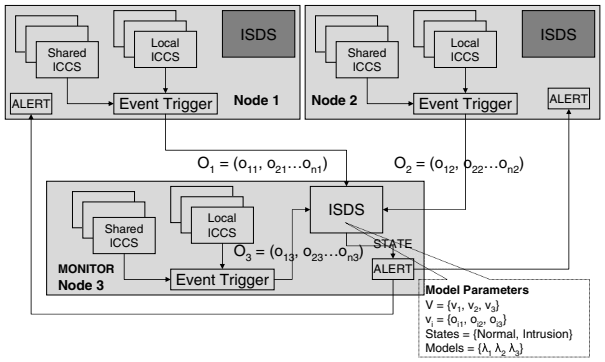

Figure 4: Distributed state estimation using HMM. Similar observations from each node act as combined observation at the monitor node according to its weight. Each node contributes multiple observations to the monitor node. Monitor node then executes HMM mixture algorithm and returns the estimated state back to the host.

4.2 IDS Monitor

IDS monitor is required to offload the member nodes from performing the redundant computation by distributing the computational load and selecting a new monitor periodically. Monitor function is performed by sharing the individual node's observations by either exchanging the data or overhearing the node traffic on all the members of the cluster (see Fig. 4). Some static information related to route and location still needs to be transmitted to the monitors. There can be more then one monitors in the cluster executing independent of each other and yields this role to another node upon cost evaluation. This allows accurate evaluation using multiple samples. Node monitors implement HMM mixture model using the multiple observations from all the member nodes as defined by ISDS (see Fig. 3). Monitor node is responsible for (a) estimating the current state of the cluster (state estimation), (b) re-training the model based on the cluster dynamics (expectation maximization), (c) initiating a trigger response to allow all member nodes to update any sharable information (location, routes, trigger data, etc.), (d) listening to member nodes who may be experiencing abnormal activity, (e) yielding monitor role to another monitor using a hand-off mechanism, and (f) alerting the nodes of a change in intrusion state.

Hence, node monitor completes the feedback loop by initiating the response action in case the state transitioned to an intrusion state. It is expected that the response action will help scale back the abnormal activity to normal activity and therefore reduce the control feedback error. In case of multiple monitors, each monitor votes for the estimated state and the majority vote prevails. Cooperative IDS provides with us not only with a lower battery consumption, but also with a hierarchical approach where local abnormalities are substantiated using shared processing among the member nodes of the ad hoc cluster. This evolves into an IDS tree where host nodes act as a leaf structures and the monitor nodes act as the node structure with a cooperative decision process. These decisions can be accepted/rejected according to host node's local policy.

4.3 Monitor Selection Policy

Monitor nodes are selected periodically and randomly using a cost function that favors the long-term relationships, average battery conditions, estimation trends, and fair loading as a result of voting by IDS nodes. They use a self-defined *Local Fitness Score (LF_i^j)* which represents the fitness score if an IDS Node i acted as an IDS monitor on the behalf of IDS Node j. Local Fitness score (Eq. (5)) of remote IDS nodes is calculated based on observed data of interest to the local IDS node and uses following matrix (a) RSS_i^j - received signal strength of IDS node i to IDS node j. A low signal strength will help determine if a node is misbehaving or has simply moved out of range, b) PD_i - Average period an IDS node i acted as Monitor Node, (c) VP_i^j represents the voting pattern of i as seen by node j.

$$LF_i^j = \alpha_1 GF_i + \alpha_2 RSS_i^j + \alpha_3 PD_i + \alpha_4 VP_i^j \quad (5)$$

$$VP_i = 1 - \frac{\sum_{j=0, j\neq i}^{N} |GF_j - LF_j^i|}{N} \quad (6)$$

where $\alpha_1+\alpha_2+\alpha_3+\alpha_4 = 1$. Two nodes with high *Global Fitness Score (GF_i)* (Eq. 8) calculated by accumulated average (Eq. 7) & variance of LF are selected as a monitor nodes for a timed interval, where one node acts as a Primary IDS Monitor.

$$\overline{GF_i} = \frac{\sum_{j=0}^{N} LF_i^j}{N} \quad (7)$$

$$GF_i = \beta_1 \overline{GF_i} + \beta_2 (1 - \frac{\sum_{j=0}^{N} |\overline{GF_i} - LF_i^j|}{N}) \quad (8)$$

where $\beta_1 + \beta_2 = 1$. IDS Monitor can relinquish its role prematurely in which case the process of selection is repeated with Secondary Monitor performing the temporary role of IDS Monitor.

5. EXPERIMENTAL RESULT

As an experiment, we set up intrusion checkpoint for received signal strength (RSS), round trip time (RTT), bandwidth and rate of packet drop on three mobile clients (laptops) running on 802.11g wireless controller (Fig. 4). These clients are authenticated using SSL and are kept stationary for experimental purposes. They exchange among themselves a two megabyte of training sequence

periodically that is fragmented at tunable intervals with client-3 always acting as a monitor. This tunes the PID controller, profiler and CDD for nominal operating conditions for the given condition. Additionally, the model consists of a traffic generator, environment disrupter (changing signal strength), and an attack module that simulates different types of attacks (802.11 Data/EAP Replay, Frame Injection, Password Guessing). Attack parameters consists of attack-speed [0,S_{max}] attack-period [0,T_{max}] and the attack-target [1,2,3] chosen at random. Traffic generator simulates real-time audio/video and TFTP traffic under random disruption. Upon event trigger by ICCS, all nodes transmit RSS, RTT, bandwidth and packet drop rate data to the monitor node (node-3) that executes the HMM mixture model and returns the estimated status. As a part of the recovery action, attack is scaled back upon a positive intrusion detection to allow the feedback control loop error converge to set-point. Fig. 5 shows the ROC characteristics under two attack scenarios by varying the sampling period parameter. In a disruptive environment (simulated by changing signal strength), the results are substantially better with a positive detection rate of 83-89% for sample intervals of 33-36 seconds. In the absence of PID controller, false positive rate increases between 10-13% because of premature transient responses.

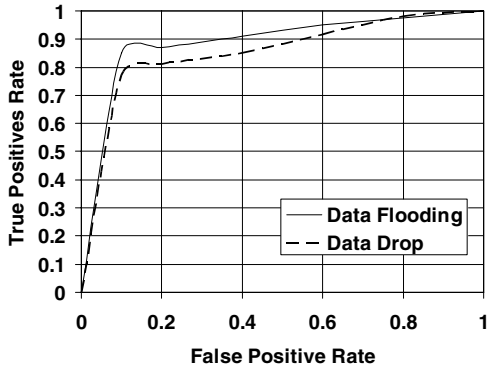

Figure 5: Receiver operating characteristics (ROC) curve showing the true positive detection/false positive ratio.

The distributed approach seems to provide a good filtration to the environmental and transient effects that otherwise result in false positive alarm. Environmental conditions results in similar observations from neighboring nodes and reduces the possibility of mis predictions. Our contribution to the IDS is holistic approach that deals with networks, system components, applications and the communication medium in a power efficient manner using distributed computations and feedback control. We not only reduce the false-positives but also reduce the computational cost using fair loading. The process can be further improved using an out-of-band agent (ARC processor) that is less intrusive to normal applications.

6. CONCLUSION

While intrusion prevention may be the first line of defense, it is not fool-proof. An exploit may use the weakest link (or node) to attack a network. This is more so in ad hoc networks, where wireless interface and MAC protocol make the node more prone to the attack. Real network traffic is also not perfect since legitimate traffic often contains the kinds of patterns typically associated with attacks which can significantly increase the false alarm rate. It is therefore essential to reduce the rate of false alarms for any IDS to be effective. Since the intrusion state cannot be inferred directly by monitoring any specific parameters, we need to predict an attack based on mixture of observable data-points, events, and current states. This leads to a statistical mechanism for intrusion prediction using HMM where observed data are represented as a weighted mixture component. Using this mechanism, an observed deviation from a normal behavior carries a higher probability of being in a non-normal state (or one of the attack states). Given the computational complexity of the HMM models, it is not practical to execute them on all battery-limited host nodes. Therefore, we enhanced the model by distributing the HMM processing such that all nodes contribute to the HMM processing in a periodic manner. We also contributed to the concept of feedback control mechanism that regulates the defensive response to every perceived abnormality. As explained earlier, this helps reduce the false alarm rate, which is one of the major problems in modern IDS. Modern silicon (CPU, I/O hubs, PCI express devices) contains performance counters that can be measured at moderate granularity. To avoid software overhead, these counters can be mapped to the feedback control modules. These modules can multiplex multiple measurements that help in battery and cost savings. While this methodology effectively solves some issues of IDS, especially IDS for MANET, it is not a complete solution; mechanisms that can provide more lead-time in identifying early signs of attacker's activities to minimize the damage are still needed. Modern intrusion detection systems also lack automated response due to high potential for inappropriate response and mis-diagnosis. Damage recovery is another area for improvement, lack of which will make it difficult to create a closed-loop control.

7. REFERENCES

[1] S. Ci, M. Guizani, H. H. Chen, and H. Sharif, "Self-regulating network utilization in mobile ad-hoc wireless networks," *IEEE Trans. Veh. Technol.*, vol. 55, no. 4, pp. 1302–1310, July 2006.

[2] X. Du, Y. Xiao, S. Guizani, and H. H. Chen, "A secure routing protocol for heterogeneous sensor networks, *in Proc. IEEE Globecom'06*, Nov. 2006, San Francisco, CA.

[3] L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, Feb. 1989.

[4] R. Khanna and H. Liu, "System approach to intrusion detection using hidden Markov model," in *Proc. 2006 Int. Conf. Commun. and Mobile Comput. (IWCMC'06)*, July 2006, pp. 349-354.

[5] S. S. Joshi and V. V. Phoha, "Investigating hidden Markov models capabilities in anomaly detection," in *Proc. 43rd Annual Southeast Regional Conf. (ACM-SE 43)*, Kennesaw, GA, Mar. 2005, pp. 98–103.

[6] A. Arnes, F. Valeur, G. Vigna, and R. Kemmerer, "Using hidden Markov models to evaluate the risks of intrusions: System architecture and model validation," in *Proc. Int. Symp. Recent Advances in Intrusion Detection (RAID)*, Hamburg, Germany, Sep. 2006.

[7] D. Ourston, S. Matzner, W. Stump, and B. Hopkins, "Applications of hidden Markov models to detecting multi-stage network attacks," in *Proc. 36th Annual Hawaii Int. Conf. (System Sciences, 2003)*, Hamburg, Germany, Jan. 2003.

[8] W. Wang, X. -H. Guan, and X.-L. Zhang, "Modeling program behaviors by hidden Markov models for intrusion detection," in *Proc. Int. Conf. Machine Learning and Cybernetics, 2004*, Aug. 2004, pp. 2830–2835.

[9] S. Zanero, "Behavioral Intrusion Detection," in *In ISCIS 2004*, 2004.

[10] D. Wagner and D. Dean, "Intrusion detection via static analysis," in *Proc. IEEE Symposium on Research in Security and Privacy*, Oakland, CA, 2001.

[11] S. Manganaris, M. Christensen, D. Serkle, and K. Hermix, "A data mining analysis of RTID alarms," *2nd Int. Workshop Recent Advances in Intrusion Detection*, Purdue Univ., West Lafayette, Indiana, USA, Sep. 1999.

[12] G. Widmer and M. Kubat, "Learning in the presence of concept drifting and hidden contexts," *Machine Learning*, vol. 23, pp. 69–101, 1996.

[13] W. Fan, "Systematic data selection to mine concept-drifting data streams," *ACM SIGKDD*, 2004.

[14] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, pp. 79–86, Mar. 1951.

[15] G. R. Grimmett and D. R. Stirzaker, *Probability and random processes*. Oxford, U.K.: Clarendon Press, 2nd edition, 1992.

[16] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Magazine*, pp. 47–59, Nov. 1996.