Distributed Change Detection for Worms, DDoS and other Network Attacks

Alvaro A. Cardenas, John S. Baras and Vahid Ramezani

***Abstract*— Self-propagating code (worms) and Distributed Denial of Service (DDoS) attacks are the most frequent and quite devastating attacks on communication networks and the Internet. In this paper we provide novel formulations for the rapid detection of these attacks in the control-theoretic framework of change detection. We present algorithms that effectively can detect worms from their temporal spreading characteristics. We describe the effects of the network topology on the algorithms and their performance. We next present algorithms for detecting DDoS while discriminating against changes in the normal traffic. This is accomplished by a distributed detection formalism where a concept of directionality is introduced and exploited. We then turn into attacks to routing protocols in mobile wireless networks. We develop change detection formulations involving Hidden Markov models, which match distribution of the number of hops in the mobile and wireless nodes. Using observations that suggest that this distribution is altered substantially in the presence of such attacks we develop and analyze algorithms for their detection.**

I. Introduction

Intrusion detection mechanisms usually monitor and detect the misuse of network resources by keeping a series of statistics related to the normal or acceptable use of the network. Continuous monitoring of the network statistics is performed and as soon as the monitored statistics cross certain thresholds or violate a fixed policy on network usage an alarm is raised.

Sequential detection theory provides an ideal framework to analyze and propose new algorithms for the quickest change detection of the monitored statistics. In this paper we use this approach to quickly detect attacks such as Worm spreading and Distributed Denial of Service. Due to the large scale of these attacks a distributed formulation where sensors are placed in different parts of the network is considered. In this way we are able to get a big view of the state of the network. Finally we consider monitoring the hop count distribution for distance vector routing algorithms as an approach to detect attacks to the routing protocols of wireless ad hoc networks. We assume the reader has some knowledge on change detection theory [1].

II. Change Detection for Worms

For clarity of presentation we will consider active worms as opposed to email worms. Active worms are programs that self-propagate across a network by exploiting vulnerabilities in widely used services offered by computers in the network. In order to locate the vulnerable computers, the worm probes different computer addresses at the specific port number of the service it is looking for. By exploiting the security flaw in the service offered by the computer, the worm can execute arbitrary code with elevated privileges, allowing it to copy and execute itself in the compromised machine. In order to reproduce, the worm scans for new vulnerable machines from each new compromised computer. The prevalence of active worms can be seen from some examples in the last couple of years: Code Red I (July 2001), Code Red II (August 2001), NIMDA (September 2001), Sapphire, also known as Slammer (January 2003) and Blaster (August 2003).

A. Why is it important to detect worms early in their development?

The top three categories of computer attacks are directly related to worms and other self-propagating hybrid threats which exploit multiple vulnerabilities across desktops and servers.

We would like to detect a worm as soon as possible in order to minimize the number of compromised hosts. A case example is the quick discovery and prompt action by System Administrators which prohibited Slapper from spreading further and prevented its damage [2]. Some highly contagious worms can also have side effects such as BGP routing instabilities [3] when they reach their peak. Currently however, detection usually relies via informal email discussion on a few key mailing lists. This process takes hours at a minimum, which is too slow for the rapidly-propagating worms.

Furthermore in [4] it is stated that the spread of the theoretical flash or Warhol worms will be so fast that no human-driven communication will suffice for adequate identification of an outbreak before nearly complete infection is achieved. It is therefore proposed to sponsor research in automated mechanisms for detecting worms based on their traffic patterns.

B. Detection algorithms

Although the spread of a worm increases traffic over a network, the worm itself is small (Code Red was 4KB), and it only takes 40 bytes for a TCP SYN packet to determine if a service is accessible, so detection cannot rely of bandwidth statistics. However, the self propagating code will try to use specific vulnerabilities that can be

This material is based upon work supported by the U.S. Army Research Office under Award No. DAAD19-01-1-0494 to the University of Maryland College Park.

The Authors are with The Institute for Systems Research and the Electrical and Computer Engineering Department, University of Maryland, College Park, MD, 20740

identified with certain port numbers. So in the rest of this chapter we will assume that the traffic monitoring variable X is the connection attempts (probes) to a given TCP/UDP port number(s). We will also assume most of the times a parametric pdf $f(X)$ on the traffic observations.

The use of host unreachable messages and connection attempts to routers as a way of detecting worms will be less reliable while the worm is getting off the ground if it uses a hit-list scanning [4]. The observations can be made at different participating ISPs enforcing policies for blocking self-propagating code once it is detected. So in our framework we assume that there is a baseline of connections to the given monitored port in all sensors (computers) of the network.

We explore the effect of aggregation from distributed sensors. This approach is motivated by the current infrastructure of distributed Intrusion Detection Systems. Further motivation is presented in [4] as the authors propose to foster the deployment of a widespread set of sensors for worm detection (possibly in the Internet backbone.)

We first introduce the simple model of detecting changes in the mean, and then we introduce a "signature" for worm detection, as a way to reduce the detection delay (or the false alarm rate).

1) Distributed detection of a change in the mean: Clearly the simplest approach to change detection is to detect a change in the mean.

Despite the abundance of techniques addressing the change detection problem, optimum schemes can mostly be found for the case where the data are independent and identically distributed (i.i.d.) and the distributions are completely known before and after the change time k_0 [5]. The cumulative sum (CUSUM) and the Shiryaev-Roberts statistics are the two most commonly used algorithms for change detection problems.

Let $\{X_k\}$ be the aggregate traffic from all the sensors in the network. To detect a change in the mean we assume $\{X_k\}$ is i.i.d with pdf $f^{(0)}$ before and $f^{(1)}$ after the change, such that the historical mean $E[f^{(0)}(X)]$ is less than the change mean $E[f^{(1)}(X)]$.

2) Detection of an exponential signal in noise: Clearly detecting a change in the mean might give rise to several false alarms as there might be cases where the observed traffic increases during the normal operation of the network. Furthermore, the i.i.d assumption of the observations after the change is too strong because each infected host will try in general to scan the same number of hosts in a given interval of time, and as more and more hosts become infected X_k will increase with k. In particular we know from simple population dynamic models that a worm scanning uniformly random the whole network will follow a logistic growth [4].

Let η be the population of infected hosts. Let r be the intrinsic growth rate (the growth rate when η is small) and let a be a given positive constant. Then the logistic growth satisfies the nonlinear ordinary differential equation

$$\frac{d\eta}{dt} = (r - a\eta)\eta \tag{1}$$

with solution

$$\eta(t) = \frac{N_0 B}{N_0 + (B - N_0)e^{-rt}} \tag{2}$$

where $B = r/a$ and N_0 is the population at time 0. Since we are interested in detecting a worm as soon as possible we will be interested in the behavior of $\eta(t)$ when it is small, i.e. we consider the exponential growth

$$\eta(t) = N_0 e^{rt} \tag{3}$$

The equivalent discrete time recursion is

$$\eta(k\Delta t) := \eta_k^d = N_0 m^k \tag{4}$$

(d stands for "discrete") where m is the discretized growth rate when η_d is small ($m = e^r$) and N_0 is the number of hosts compromised at $k = 0$.

For the detection problem we will assume that the values of N_0 and r (or m) are unknown. We will also consider a dummy signal η_k^{dummy} to represent any other growth pattern we want to discriminate (e.g. linear growth, a step function etc) from the growth of the worm η_k^d.

We assume a normal traffic aggregate

$$W_k = \sum_{l=1}^{L} W_{l,k} \tag{5}$$

distributed as $f^{(0)}(w_1, ..., w_k)$. Let X_k denote the aggregate observation from all sensors at time k, i.e.

$$X_k = \sum_{l=1}^{L} X_{l,k} \tag{6}$$

Our main assumption in this section is that the number of probes seen at the sensors will be proportional to the number of infected hosts $\alpha\eta_k^d$. The usual change detection hypothesis testing problem for the aggregate traffic (equation 6) would be as follows:

$$H_0 : x_k = \eta_k^{dummy} + w_k \quad \text{when } 1 \le k \le M$$

$$H_1 : \begin{array}{ll} x_k = \eta_k^{dummy} + w_k & \text{when } 1 \le k < k_0 \\ x_k = \alpha\eta_k^d + w_k & \text{when } k_0 \le k \le M \end{array}$$

However, we want k to restart to 1 whenever H_0 is accepted, so we use a sequential hypothesis test where the change time k_0 is implicitly given by the time in which the sequential test restarted and H_1 was accepted.

$$H_0 : x_k = \eta_k^{dummy} + w_k \quad \text{when } 1 \le k \le M$$

$$H_1 : \; x_k = \alpha\eta_k^d + w_k \text{ when } 1 \le k \le M$$

a) Exponential signal detection in noise: Since we assume we do not know the parameters α, N_0 and m, we compute the generalized likelihood ratio (GLR) in a given time window [1,...,M] and compare it to a threshold h. We also assume the dummy signal has some unknown parameter β (e.g. the slope in a linear growth). Therefore detection of the signal $\alpha\eta_k^d$ in noise w_k is achieved with the test:

$$\frac{\sup_{\alpha,N_0,m} f^{(0)}(x_k - \alpha\eta_k^d)}{\sup_\beta f^{(0)}(x_k - \eta_k^{dummy})} \underset{H_0}{\overset{H_1}{\gtrless}} h \tag{7}$$

b) Nonparametric regression detection: So far we have always been assuming a parametric distribution $f^{(0)}(w_1,...,w_k)$ for the normal traffic. This assumption is valid for a wide number of ports as the traffic seen can be regular. However in some cases the real distribution can be quite difficult to obtain. For example the number of probes seen to port 80 (WWW) or port 21 (FTP) for computers providing those services can exhibit long range dependence and multifractal behavior that can be difficult to capture with a parametric model. In order to deal with some of the more complicated traffic observations we propose a heuristic non-parametric change detection algorithm similar in essence to the problem of detection of an exponential signal in noise.

The idea is to do a linear regression on $\log(x_i)$. This regression will produce two parameters, a slope c and the error err from the estimated regression of $x_k, ..., x_{M+k}$. From this we compute the statistic $z_{M+k} = c/err$. We use a sliding window on k to compute the statistic. Then we apply the non-parametric version of CUSUM or the Girshik-Rubin-Shiryaev algorithms to z_{M+k} [6].

III. Change Detection for DDoS Attacks

A. Why is it important to quickly detect routers participating in a denial of service attack?

Almost all DDoS attacks involve multiple networks and attack sources, many of which have spoofed IP addresses to make detection even harder. An attempt of the victim to choke off the offending traffic requires network administrators to call upstream service providers, alerting them of the attack and having them shut down the traffic. That process has to be repeated all the way back to every attack source. So although DDoS are easily identified at the victim's site, it is natural to extend the quickest detection problem to transit networks (ISPs) for faster response to an attack.

At the ISP level, traffic anomalies are difficult to detect in the aggregated network traffic. Examination at per-flow basis at the IP level cannot usually scale up to the high-speed links in the transit networks, so a reasonable approach for transit networks carrying a large amount of traffic which cannot be analyzed at line rate should not keep the number of packets to a specific destination, as this might be too expensive during operation. Thus we are interested only in passively monitoring the aggregate traffic, without the need to store header information from the packets transmitted through the network.

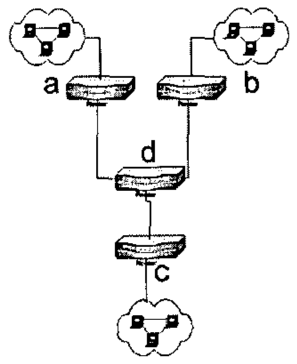

Fig. 1. A transit network composed of nodes a, b, c and d.

B. Detection Algorithms

1) Problem formulation : We take a new approach for identifying Distributed Denial of Service attacks by a set of nodes in a transit network. The basic idea is that at each highly connected node the data tends to aggregate from the distributed sources toward the destination, giving a sense of directionality to the attack. This directionality idea will provide a framework to design change detection algorithms that are going to be less sensitive to changes in the average intensity of the overall traffic and will focus on differentiating random fluctuations of the network traffic versus fluctuations where there is a clear change in the direction of the flow at a given node. We are considering packets in a very broad and general way, but clearly our approach can be extended to monitor certain specific packet types given the protocol. For example we might be interested in measuring only TCP SYN-ACK response packets for identifying a reflected distributed denial of service attack, or ICMP packets for identifying ping floods.

Assume we are monitoring node d in figure 1. Let $X_k^{d,m}$ denote the stochastic process representing the total number of packets sent by d through the link (d,m) (an ordered pair) at time step k, where $m \in \mathcal{N}(d)$ denotes a neighbor of d, and $\mathcal{N}(d)$ the set of neighbors of d. Let X_k^d denote the vector with the elements $X_k^{d,m}$ and let

$$\theta_0^d := \begin{bmatrix} E_0[X_k^{d,a}] \\ E_0[X_k^{d,b}] \\ E_0[X_k^{d,c}] \end{bmatrix} \tag{8}$$

We will be interested in changes of the form:

$$\theta_0^d + \nu\Upsilon_m \tag{9}$$

where ν is a non-negative scalar and Υ_m (in the case of three observed links $|\mathcal{N}(d)| = 3$) is one of the usual basis vectors of the three dimensional Euclidean space. Namely:

$$\Upsilon_a = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \Upsilon_b = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \Upsilon_c = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (10)$$

So in figure 1, if node d suddenly starts a broadcast, there will be a change in the mean of all processes. However we are not interested in such a change. Instead if there are attackers in the subnetworks attached to b and c, and they target a host in the network attached to a by flooding it, there will be a change in the direction Υ_a. Testing directions should help us in discriminating unwanted false alarms due to random fluctuations of the flows.

To formalize our ideas we consider the framework discussed in [1] of change detection in a known direction but unknown magnitude of the change. Our problem is a little bit different in that we are considering an M-ary sequential hypothesis testing problem and that we will not allow changes with negative or zero values for ν, i.e. we impose the restriction $\nu \geq 0$.

Thus the resulting change detection problem is:

$$\theta^d(k) = \begin{cases} \theta_0^d \text{ when } k < t_{change} \\ \theta_0^d + \nu\Upsilon_a \quad \text{or} \\ \theta_0^d + \nu\Upsilon_b \quad \text{or} \\ \theta_0^d + \nu\Upsilon_c \text{ when } k \geq t_{change} \end{cases} \quad (11)$$

where t_{change} is an unknown time step when the change occurs.

Since we have an unknown parameter ν we follow the generalized likelihood ratio (GLR) for a multihypothesis test: a test for each possible direction Υ_m, vs the null hypothesis: a change in all directions Υ_d. The null hypothesis is selected for discriminating a change in one direction vs a change of the overall traffic of the network either as an increase or decrease:

$$g_k^{d,m} = \max_{1\leq j\leq k} \log \frac{\sup_{\nu\geq c_1} \prod_{i=j}^k f_{\theta_0^d+\nu\Upsilon_m}(X_i^d)}{\sup_\lambda \prod_{i=j}^k f_{\theta_0^d+\lambda\Upsilon_d}(X_i^d)}$$

where λ is a scalar not necessarily greater than a positive constant c_1 unlike ν (i.e. we allow also for a decrease in the overall network traffic). The threshold $h^{d,m}$ for each of the tests is selected given a fixed false alarm rate probability.

To stop the test we can run all hypothesis in parallel and only the test $g_k^{d,m}$ that reaches its given threshold is stopped. However this is a heuristic procedure as optimal solutions to the problem of sequential testing of more than two hypotheses are, in general, intractable. A more elaborate stopping rule is presented in [7] with a proof of asymptotic optimality as the decision risks (or error probabilities) go to zero.

2) Sensor Fusion: So far we have been focusing on detecting a change in a single node. One of the main advantages in having several nodes under monitoring is that we can perform an aggregation of the statistics between the different nodes in order to decrease our detection delay

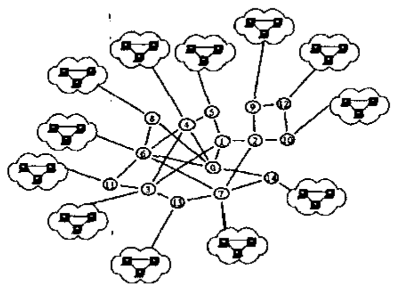

Fig. 2. The transit network

given a fixed false alarm rate probability. In particular if we are monitoring nodes far away from the destination, most of the local statistics will not yield an alarm and the attack might be unnoticed.

The alarm aggregation can be performed by several methods. Here we propose a simple heuristic that will apply to any distance vector routing protocol.

We want a mechanism to aggregate the different statistics at each monitored node, taking into account that the computed statistics for all nodes can vary to different scales of magnitude yielding a biased addition. To cope with this problem we compute the normalized statistic $\varphi_k^{d,m} := \frac{g_k^{d,m}}{h^{d,m}}$. If none of our monitored nodes has raised an alarm, the number of monitored nodes will be bounded by $\sum_d \varphi_k^{d,m}$. This can be in turned interpreted as a new upper bound for a collective threshold which can be selected given a false alarm rate probability.

Selecting which statistics to add is the key issue. In keeping with our directionality framework we will combine only the statistics relating two or more nodes to a common destination. The algorithm is as follows:

```
Given two nodes d and e;
For each link $d \rightarrow m$
    For each link $e \rightarrow n$
    If there is a node f reachable through $d \rightarrow m$ and $e \rightarrow n$
        then add their normalized statistic;
```

We now apply this formulation to the case of two monitored nodes (a natural extension follows when we are monitoring several nodes). Suppose we monitor nodes 6 and 3 in the transit network model shown in figure 2, where the transit network consists of 15 routers numbered from 0 to 14. Each cloud represents a stub network with its own routing domain.

The routing tables required for the aggregation algorithm are given in Tables 1 and 2. By simple inspection of the routing tables we see that we need to correlate the link (6,0) with (3,1) because nodes 6 and 3 use them (respectively) to reach nodes 0, 1 and 14. Similarly, the link (6,11) must be correlated with (3,11), link (6,4) with (3,4), link (6,7) with

Link	Routing to nodes
(6,7)	7,13,2,10,12,9
(6,0)	0,14,1
(6,4)	4,3,5
(6,11)	11,3
(6,8)	8
(6,subnetwork)	

TABLE I
ROUTING TABLE FOR NODE 6

Link	Routing to nodes
(3,1)	1,0,2,14,10,9,12
(3,13)	7,13
(3,4)	4,5
(3,11)	11,6,8
(3,subnetwork)	

TABLE II
ROUTING TABLE FOR NODE 3

(3,13), (6,7) with (3,1) and (6,8) with (3,11).

If we denote as H_i the hypothesis when node i or its subnetwork are under attack, then we have the following hypothesis testing problem created by the aggregation mechanism

$$\begin{aligned}
1_{(H_0 \vee H_1 \vee H_{14})} &= \varphi^{6,0} + \varphi^{3,1} > h_{0\vee1\vee14} \\
1_{(H_{11})} &= \varphi^{6,11} + \varphi^{3,11} > h_{11} \\
1_{(H_4 \vee H_5)} &= \varphi^{6,4} + \varphi^{3,4} > h_{4\vee5} \\
1_{(H_{13} \vee H_7)} &= \varphi^{6,7} + \varphi^{3,13} > h_{13\vee7} \\
1_{(H_2 \vee H_{10} \vee H_{12} \vee H_9)} &= \varphi^{6,7} + \varphi^{3,1} > h_{2\vee10\vee12\vee9} \\
1_{(H_8)} &= \varphi^{6,8} + \varphi^{3,11} > h_8
\end{aligned}$$

where $1_{()}$ is the indicator function of the Hypotheses. If we have fixed routes in the network, the thresholds $h_{i\vee...\vee j}$ can be computed to reach a given false alarm rate.

With this formulation not only can we improve our chances to detect "buried" attacks in single links by correlating statistics, but also diminish the impact of false alarms originating in individual nodes.

IV. CHANGE DETECTION FOR ROUTING ATTACKS IN AD-HOC NETWORKS

A. Problem formulation: mobility and hop count distribution

Our objective is to present a statistical framework that allows the incorporation of prior information about the normal behavior of the network and of network attacks in a principled way for the detection of known and unknown attacks. In order to avoid a large number of false alarms, we have to consider robust statistical models describing a baseline behavior for our feature of interest in MANETs. In contrast to other frameworks that allow anomaly detection [8], we focus on the dynamic behavior of the protocol rather than using static models.

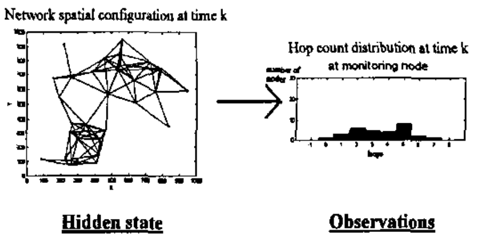
Network spatial configuration at time k
Hop count distribution at time k at monitoring node
Hidden state
Observations

Fig. 3. HMM interpretation

In a highly mobile ad hoc network, as viewed by a monitoring node, the hop count is an important statistic and in most cases can be monitored with no overhead. The evolution of this distribution is directly related to the changes in the topology of the network. Each configuration imposes certain constraints on the space of hop count distributions and as the topology of the network changes from one set of configurations to the next. The space of the configuration can be abstracted and viewed as representing the hidden states of the network and the hop count distribution as the observations.

In terms of the intrusion detection, the basic idea is that an attacker will change the routing information or maliciously modify the routing algorithm in such a way that our perceived evolution of the hop count distribution differs from the its dynamics under the "normal" conditions. When such a deviation persists, in a statistical sense to be described, we declare that an intrusion has occurred.

B. Statistical Model

We build a discrete Hidden Markov Model (HMM) [9] with parameters (π, A, B) for modeling the evolution of hop count distributions. HMMs were selected for several reasons. They provide an generative representation of our system as the hidden states of the HMM can be viewed as abstractions of different spatial configurations of the mobile nodes (figure 3) and the observations as the dynamic evolution of the hop count distribution. The parameters of discrete state HMMs can be specified or can be estimated efficiently while keeping a model with a low bias. The generative and intuitive nature of HMMs allows incorporation of prior knowledge and misuse detection by providing a *language model*, i.e. a model that provides the HMM with expert information on allowable state transitions which reflect our knowledge on mobility. Signature-like intrusion detection can also be incorporated by using HMM models of the attacks we already know.

For simplicity, we will assume a proactive distance vector routing protocol such as DSDV [10] in order to have access to all hop counts at any time.

If we have $N+1$ nodes, the hop count distribution at the time step k can be considered as a vector in $\{0,...,N\}^D$:

$X_k = [X_k^0, ..., X_k^{D-1}]'$ ($X_k^i \in \{0, ..., N\}$) where D is a limit we impose in the maximum number of hops we will consider, i.e. X_k^0 is the number of disconnected nodes, is X_k^1 the number of nodes 1 hop away, ..., X_k^{D-2} is the number of nodes $D-2$ hops away and X_k^{D-1} is the number of nodes $D-1$ or more hops away.

In order to consider a discrete HMM we need a way to deal with the high-dimensional observation vectors X_k. The number of all possible observations is $(N+1)^{D-1}$ since we are working with a hyperplane in $(N+1)^D$ with constraint $\sum_{i=0}^{D-1} X_k^i = N+1$. A natural approach is to encode X_k to Y_k, a member of a set of M codewords. A good selection of the codewords is nontrivial. One approach to obtain the codebook Ξ is to learn it from the normal operation of the network, or from simulations of expected node mobility. The learning algorithm can be a compression algorithm in which for a given fixed rate R, we try to find the codebook that minimizes a distortion function (usually a quadratic distortion is considered). Another approach to is to consider a set of M key reference distributions of the hop counts, chosen by an expert trying to define the observables of an anomalous behavior.

C. Detection

In order to continue in the change detection setup we follow a CUSUM procedure applicable to the case of dependent observations x_j with distributions f_{θ_1} and f_{θ_0} under hypotheses H_1 and H_0 respectively [11]:

$$S_n = \left\{ S_{n-1} + \log\left(\frac{f_{\theta_1}(x_n|x_{n-1}, ..., x_k)}{f_{\theta_0}(x_n|x_{n-1}, ..., x_k)} \right) \right\}^+ \qquad (12)$$

where x_k is the first sample after the last reset, i.e., $S_{k-1} = 0$. It is clear that this algorithm is only a reformulation of the sequential probability ratio test (SPRT) algorithm for the log-likelihood ratio: $\log\left(\frac{f_{\theta_1}(x_n|x_{n-1},...,x_k)}{f_{\theta_0}(x_n|x_{n-1},...,x_k)}\right)$ with the lower threshold selected at 0. The upper threshold h will be selected given a false alarm rate.

The attack models can be intuitively represented as HMMs $f_{\theta_1} = (\pi_{\theta_1}, A_{\theta_1}, B_{\theta_1})$. Prior knowledge and misuse detection can be introduced as previously discussed. We can also take an approach of anomaly detection by selecting the uniform distribution, i.e. $\forall \hat{x} \in \Xi$ we have $f_{\theta_1}(\hat{x}) = 1/M$ as the alternate hypothesis. By the principle of maximum entropy we can conclude that this is a way of not assuming anything about the attack and therefore it is particularly suited for detecting the attacks we do not know.

V. Experiments, Simulation and Results

We performed several experiments to evaluate the performance of the algorithms under a wide range of network traffic assumptions and network topologies. The experiments and simulations can be found in [12]. Here we summarize some of the results.

The worm detection problem is heavily dependent on the network topology and selection of monitoring nodes. In scale-free networks [13] a very small set of the highly connected nodes is sufficient for detection, and aggregation only improves the performance of the nonparametric statistics. However, if we select sensors at random or if we monitor a random network [13] then aggregation is very important for detection. Most of the parametric statistics perform comparably under a wide variety of conditions. However when the traffic deviates significantly from the assumed distribution, the best performance is obtained by the nonparametric statistics.

In the simulation of denial of service attacks, for local detection, testing for changes in the direction of the flow (our discrimination parameter from normal changes in the network) provides better performance than simply testing for change detection separately per link. In the distributed detection case, by correlating an overall flow directionality we were able to extract warning of attacks that would have been otherwise missed in local detection nodes.

For ad hoc networks, our HMMs provide an intuitive model of the network routing behavior, and a principled way for adding expert knowledge in the form of language models. Simple disruptions to the routing protocol such as a faulty node claiming a random distance to any other nodes can be detected with a system that learns the normal behavior of the network and uses the anomaly detection framework. Detection of more complex attacks such as a Blackhole or a Wormhole require incorporation of prior knowledge into the HMMs in the form of a normal behavior specification or as attack models.

References

[1] M. Basseville and I. Nikiforov, *Detection of Abrupt Changes: Theory and Application.* Englewood Cliffs, NJ: Prentice Hall, 1993.

[2] I. S. S. Inc., "Internet risk impact summary for june 25, 2002 - september 27, 2002."

[3] J. Cowie, A. Ogielski, B. Premore, and Y. Yuan, "Global routing instabilities during code red ii and nimbda worm propagation," 2002.

[4] S. Staniford, V. Paxson, and N. Weaver, "How to 0wn the internet in your spare time," in *Proceedings of the 11th USENIX Security Symposium (Security '02)*, 2002.

[5] G. V. Moustakides, "Quickest detection of abrupt changes for a class of random process," *IEEE Transactions on Information Theory*, vol. 44, no. 5, September 1998.

[6] B. Brodsky and B. Darkhovsky, *Nonparametric Methods in Change-Point Problems.* Kluwer Academic Publishers, 1993.

[7] V. P. Dragalin, A. G. Tartakovsky, and V. V. Veeravalli, "Multihypothesis sequential probability ratio tests-part 1:asymptotic optimality," *IEEE Transactions on Infromation Theory*, vol. 45, no. 7, November 1999.

[8] Y. Zhang, W. Lee, and Y. Huang, "Intrusion detection techniques for mobile wireless networks," in *ACM/Kluwer Mobile Networks and Applications (MONET)*, 2002.

[9] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, February 1989.

[10] C. Perkins, Ed., *Ad Hoc Networking.* Addison Wesley, 2000.

[11] B. Chen and P. Willet, "Detection of hidden markov model transient signals," *IEEE Transactions on Aerospace and Electronics Systems*, vol. 36, no. 4, pp. 1253–1268, October 2000.

[12] A. A. Cardenas, "Change detection algorithms for information assurance of computer networks," Master's thesis, University of Maryland, College Park, 2002.

[13] R. Albert and A. Barabasi, "Statistical mechanics of complex networks," *Reviews of Modern Physics*, pp. 47–97, jan 2002.