Anomaly Detection in IP Networks

Marina Thottan and Chuanyi Ji

***Abstract*—Network anomaly detection is a vibrant research area. Researchers have approached this problem using various techniques such as artificial intelligence, machine learning, and state machine modeling. In this paper, we first review these anomaly detection methods and then describe in detail a statistical signal processing technique based on abrupt change detection. We show that this signal processing technique is effective at detecting several network anomalies. Case studies from real network data that demonstrate the power of the signal processing approach to network anomaly detection are presented. The application of signal processing techniques to this area is still in its infancy, and we believe that it has great potential to enhance the field, and thereby improve the reliability of IP networks.**

***Index Terms*—Adaptive signal processing, autoregressive processes, eigenvalues and eigenfunctions, network performance, network reliability.**

I. Introduction

NETWORKS are complex interacting systems and are comprised of several individual entities such as routers and switches. The behavior of the individual entities contribute to the ensemble behavior of the network. The evolving nature of internet protocol (IP) networks makes it difficult to fully understand the dynamics of the system. Internet traffic was first shown to be composed of complex self-similar patterns by Leland *et al.* [1]. Multifractal scaling was discovered and reported by Levy-Vehel *et al.* [2]. To obtain a basic understanding of the performance and behavior of these complex networks, vast amounts of information need to be collected and processed. Often, network performance information is not directly available, and the information obtained must be synthesized to obtain an understanding of the ensemble behavior. In this paper, we review the use of signal processing techniques to address the problem of measuring, analyzing, and synthesizing network information to obtain normal network behavior. The normal network behavior thus computed is then used to detect network anomalies.

There are two main approaches to studying or characterizing the ensemble behavior of the network: The first is the inference of the overall network behavior through the use of network probes [3] and the second by understanding the behavior of the individual entities or nodes. In the first approach, which is often referred to as network tomography [4], there is no assumption made about the network, and through the use of probe measurements, one can infer the characteristics of the network. This is a useful approach when characterizing noncooperative networks or networks that are not under direct administrative control. In the case of a single administrative domain where knowledge of the basic network characteristics such as topology are available, an entity-based study would provide more useful information to the network administrator. Using some basic knowledge of the network layout as well as the traffic characteristics at the individual nodes, it is possible to detect network anomalies and performance bottlenecks. The detection of these events can then be used to trigger alarms to the network management system, which, in turn, trigger recovery mechanisms. The methods presented in this paper deal with entity-based measurements.

The approaches used to address the anomaly detection problem are dependent on the nature of the data that is available for analysis. Network data can be obtained at multiple levels of granularity such as end-user-based or network-based. End-user-based information refers to the transmission control protocol (TCP) and user datagram protocol (UDP) related data that contains information that is specific to the end application. Network-based data pertains to the functioning of the network devices themselves and includes information gathered from the router's physical interfaces as well as from the router's forwarding engine. Traffic counts obtained from both types of data can be used to generate a time series to which statistical signal processing techniques can be applied [5], [6]. However, in some cases, only descriptive information such as the number of open TCP connections, source-destination address pairs, and port numbers are available. In such situations, conventional approaches of rule-based methods would be more useful [7].

The goal of this paper is to show the potential to apply signal processing techniques to the problem of network anomaly detection. Application of such techniques will provide better insight for improving existing detection tools as well as provide benchmarks to the detection schemes employed by these tools. Rigorous statistical data analysis makes it possible to quantify network behavior and, therefore, more accurately describe network anomalies. The scope of this paper is to describe the problem of IP network anomaly detection in a single administrative domain along with the types and sources of data available for analysis. Special emphasis is placed on motivating the need for signal processing techniques to study this problem. We describe areas in which advances in signal detection theory have been useful. For example, there are no accurate statistical models for normal network operation, and this makes it difficult to characterize the statistical behavior of abnormal traffic patterns. In this paper, we present a technique based on abrupt change detection for addressing this challenge

Manuscript received October 4, 2002; revised April 2, 2003. This work was supported by DARPA under Contract F30602-97-C-0274 and the National Science Foundation under Grant ECS 9908578. The associate editor coordinating the review of this paper and approving it for publication was Dr. Rolf Riedl.

M. Thottan is with the Department of Networking Software Research, Bell Laboratories, Holmdel, NJ 07733 USA (e-mail: marinat@lucent.com).

C. Ji is with the Department of Electrical, Computer, and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: jic@ece.gatech.edu).

Digital Object Identifier 10.1109/TSP.2003.814797

[8]. Furthermore, there is no single variable or metric that captures all aspects of normal network function. This presents the problem of synthesizing information from multiple metrics, each of which have widely differing statistical properties. To address this issue, we use an operator matrix to correlate information from individual metrics [5]. The paper also identifies some of the challenges posed to the signal processing community by this new application.

The paper is organized as follows: Section II describes the characteristics of network anomalies along with some examples. The different sources from which network data can be obtained are described in Section III. Section IV surveys the commonly used methods for anomaly detection. In Section V, we describe a statistical technique to solve the same problem. We present an evaluation criteria for the effectiveness of the signal processing technique and summarize its performance in detecting various network anomalies. We describe four in-depth case studies on anomalous events in real networks as well as the performance of the statistical approach in each of these cases. Section VI describes relevant issues in the application of signal processing techniques to network data analysis. In Section VII, we discuss some of the open issues that provide interesting avenues to explore for future work.

II. Network Anomalies

Network anomalies typically refer to circumstances when network operations deviate from normal network behavior. Network anomalies can arise due to various causes such as malfunctioning network devices, network overload, malicious denial of service attacks, and network intrusions that disrupt the normal delivery of network services. These anomalous events will disrupt the normal behavior of some measurable network data. In this paper, we present techniques that can be employed to detect such types of anomalies.

The definition of normal network behavior for measured network data is dependent on several network specific factors such as the dynamics of the network being studied in terms of traffic volume, the type of network data available, and types of applications running on the network. Accurate modeling of normal network behavior is still an active field of research, especially the online modeling of network traffic [9]. There exists parsimonious traffic models that accurately capture fractal and multifractal scaling properties, such as the self-similar models introduced by Norros [10] and cascade models originally suggested by Crousse *et al.* [11]. While Crouse *et al.* concentrate on the signal processing issues and statistical matching to network traffic, Gilbert *et al.* provide a more network centric description of cascade-scaling [12]. Within these parsimonious modeling frameworks, different approaches can be used to do anomaly detection.

Today's commercially available network management systems continuously monitor a set of measured indicators to detect network anomalies. A human network manager observes the alarm conditions or threshold violations generated by a group of individual indicators to determine the status of the health of the network. Such alarm conditions represent deviations from normal network behavior and can occur before or during an anomalous event. These deviations are often associated with performance degradation in the network. Based on this intuitive model currently being used, we premise the following: *Network anomalies are characterized by correlated transient changes in measured network data that occur prior to or during an anomalous event*. The term *transient changes* refers to abrupt changes in the measured data that occurs in the same order as the frequency of the measurement interval. The duration of these abrupt changes varies with the nature of the triggering anomalous event.

A. Examples of Network Anomalies

Network anomalies can be broadly classified into two categories. The first category is related to network failures and performance problems. Typical examples of network performance anomalies are file server failures, paging across the network, broadcast storms, babbling node, and transient congestion [13], [14]. For example, file server failures, such as a web server failure, could occur when there is an increase in the number of *ftp* requests to that server. Network paging errors occur when an application program outgrows the memory limitations of the work station and begins paging to a network file server. This anomaly may not affect the individual user but affects other users on the network by causing a shortage of network bandwidth. Broadcast storms refer to situations where broadcast packets are heavily used to the point of disabling the network. A babbling node is a situation where a node sends out small packets in an infinite loop in order to check for some information such as status reports. Congestion at short time scales occurs due to hot spots in the network that may be a result of some link failure or excessive traffic load at that point in the network. In some instances, software problems can also manifest themselves as network anomalies, such as a protocol implementation error that triggers increased or decreased traffic load characteristics. For example, an *accept* protocol error in a super server (*inetd*) results in reduced access to the network, which, in turn, affects network traffic loads.

The second major category of network anomalies is security-related problems. Denial of service attacks and network intrusions are examples of such anomalies. Denial of service attacks occur when the services offered by a network are hijacked by some malicious entity. The offending party could disable a vital service such as domain name server (DNS) lookups and cause a virtual shutdown of the network [15], [16]. For this event, the anomaly may be characterized by very low throughput. In case of network intrusions, the malicious entity could hijack network bandwidth by flooding the network with unnecessary traffic, thus starving other legitimate users [6], [17]. This anomaly would result in heavy traffic volumes.

III. Sources of Network Data

Obtaining the right type of network performance data is essential for anomaly detection. The types of anomalies that can be detected are dependent on the nature of the network data. In this section, we review some possible sources of network data along with their relevance for detecting network anomalies. For the purpose of anomaly detection, we must characterize normal traffic behavior. The more accurately the traffic behavior can be modeled, the better the anomaly detection scheme will perform.

A. Network Probes

Network probes are specialized tools such as *ping* and *traceroute* [18] that can be used to obtain specific network parameters such as end-to-end delay and packet loss. Probing tools provide an instantaneous measure of network behavior. These methods do not require the cooperation of the network service provider. However, it is possible that the service providers could choose not to allow ping traffic through their firewall. Furthermore, the specialized IP packets used by these tools need not follow the same trajectory or receive the same treatment by network devices as do the regular IP packets. This method also assumes the existence of symmetric paths between given source-destination pairs. On the Internet, this assumption cannot be guaranteed. Thus, performance metrics derived from such tools can provide only a coarse grained view of the network. Therefore, the data obtained from probing mechanisms may be of limited value for the purpose of anomaly detection.

B. Packet Filtering for Flow-Based Statistics

In packet filtering, packet flows are sampled by capturing the IP headers of a select set of packets at different points in the network [19]. Information gathered from these IP headers is then used to provide detailed network performance information. For flow-based monitoring, a flow is identified by source-destination addresses and source-destination port numbers. The packet filtering approach requires sophisticated network sampling techniques as well as specialized hardware at the network devices to do IP packet lookup. Data obtained from this method could be used to detect anomalous network flows. However, the hardware requirements required for this measurement method makes it difficult to use in practice.

C. Data From Routing Protocols

Information about network events can be obtained through the use of routing peers. For example by using an open shortest path first (OSPF) peer, it is possible to gather all routing table updates that are sent by the routers [20]. The data collected can be used to build the network topology and provides link status updates. If the routers run OSPF with traffic engineering (TE) extensions, it is possible to obtain link utilization levels [21]. Since routing updates occur at frequent intervals, any change in link utilization will be updated in near real time. However, since routing updates must be kept small, only limited information pertaining to link statistics can be propagated through routing updates.

D. Data From Network Management Protocols

Network management protocols provide information about network traffic statistics. These protocols support variables that correspond to traffic counts at the device level. This information from the network devices can be passively monitored. The information obtained may not directly provide a traffic performance metric but could be used to characterize network behavior and, therefore, can be used for network anomaly detection. Using this type of information requires the cooperation of the service provider's network management software. However, these protocols provide a wealth of information that is available at very fine granularity. The following subsection will describe this data source in greater detail.

1) Simple Network Management Protocol (SNMP): SNMP works in a client-server paradigm [22]. The protocol provides a mechanism to communicate between the manager and the agent. A single SNMP manager can monitor hundreds of SNMP agents that are located on the network devices. SNMP is implemented at the application layer and runs over the UDP. The SNMP manager has the ability to collect management data that is provided by the SNMP agent but does not have the ability to process this data. The SNMP server maintains a database of management variables called the management information base (MIB) variables [23]. These variables contain information pertaining to the different functions performed by the network devices. Although this is a valuable resource for network management, we are only beginning to understand how this information can be used in problems such as failure and anomaly detection.

Every network device has a set of MIB variables that are specific to its functionality. MIB variables are defined based on the type of device as well as on the protocol level at which it operates. For example, bridges that are data link-layer devices contain variables that measure link-level traffic information. Routers that are network-layer devices contain variables that provide network-layer information. The advantage of using SNMP is that it is a widely deployed protocol and has been standardized for all different network devices. Due to the fine-grained data available from SNMP, it is an ideal data source for network anomaly detection.

2) SNMP—MIB Variables: The MIB variables [24] fall into the following groups: system, interfaces (*if*), address translation (*at*), internet protocol (*ip*), internet control message protocol (*icmp*), transmission control protocol (*tcp*), user datagram protocol (*udp*), exterior gateway protocol (*egp*), and simple network management protocol (*snmp*). Each group of variables describes the functionality of a specific protocol of the network device. Depending on the type of node monitored, an appropriate group of variables can be considered. If the node being monitored is a router, then the *ip* group of variables are investigated. The *ip* variables describe the traffic characteristics at the network layer. MIB variables are implemented as counters. Time series data for each MIB variable is obtained by differencing the MIB variables at two subsequent time instances called the polling interval.

There is no single MIB variable that is capable of capturing all network anomalies or all manifestations of the same network anomaly. Therefore, the choice of MIB variables depends on the perspective from which the anomalies are detected. For example, in the case of a router, the *ip* group of MIB is chosen, whereas for a bridge, the *if* group is used.

IV. Anomaly Detection Methods

In this section, we review the most commonly used network anomaly detection methods. The methods described are rule-based approaches, finite state machine models, pattern matching, and statistical analysis.

A. Rule-Based Approaches

Early work in the area of fault or anomaly detection was based on expert systems. In expert systems, an exhaustive database containing the rules of behavior of the faulty system is used to determine if a fault occurred [25], [26]. Rule-based systems are too slow for real-time applications and are dependent on prior knowledge about the fault conditions on the network [27]. The identification of faults in this approach depends on symptoms that are specific to a particular manifestation of a fault. Examples of these symptoms are excessive utilization of bandwidth, number of open TCP connections, total throughput exceeded, etc. These rule-based systems rely heavily on the expertise of the network manager and do not adapt well to the evolving network environment. Thus, it is possible that entirely new faults may escape detection. In [25], the authors describe an expert system model using fuzzy cognitive maps (FCMs) to overcome this limitation. FCM can be used to obtain an intelligent modeling of the propagation and interaction of network faults. FCMs are constructed with the nodes of the FCM denoting managed objects such as network nodes and the arcs denoting the fault propagation model.

Case-based reasoning is an extension of rule-based systems [26]. It differs from FCM in that, in addition to just rules, a picture of previous fault scenarios is used to make the decisions. A picture here refers to the circumstances or events that led to the fault. In order to adapt the case-based reasoning scheme to the changing network environment, adaptive learning techniques are used to obtain the functional dependence of relevant criteria such as network load, collision rate, etc., to previous trouble tickets [28]. The trouble ticketing system is used to perform two functions: Prepare for problem diagnostics through filtering, and infer the root cause of the problem. Using case-based reasoning for describing fault scenarios also suffers from heavy dependence on past information. Furthermore, the identification of relevant criteria for the different faults will, in turn, require a set of rules to be developed. In addition, using any functional approximation scheme, such as back propagation, causes an increase in computation time and complexity. The number of functions to be learned also increases with the number of faults studied.

B. Finite State Machines

Anomaly or fault detection using finite state machines model alarm sequences that occur during and prior to fault events. A probabilistic finite state machine model is built for a known network fault using history data. State machines are designed with the intention of not just detecting an anomaly but also possibly identifying and diagnosing the problem. The sequence of alarms obtained from the different points in the network are modeled as the states of a finite state machine. The alarms are assumed to contain information such as the device name as well as the symptom and time of occurrence. The transitions between the states are measured using prior events [29]–[31]. A given cluster of alarms may have a number of explanations, and the objective is to find the best explanation among them. The best explanation is obtained by identifying a near-optimal set of nodes with minimum cardinality such that all entities in the set explain all the alarms and at least one of the nodes in the set is the most likely one to be in fault. In this approach, there is an underlying assumption that the alarms obtained are true. No attempt is made to generate the individual alarms themselves. A review of such state machine techniques can be found in [32] and [33].

The difficulty encountered in using the finite state machine method is that not all faults can be captured by a finite sequence of alarms of reasonable length. This may cause the number of states required to explode as a function of the number and complexity of faults modeled. Furthermore, the number of parameters to be learned increases, and these parameters may not remain constant as the network evolves. Accounting for this variability would require extensive off-line learning before the scheme can be deployed on the network.

C. Pattern Matching

A new approach proposed and implemented by Maxion and others [34], [35] describes anomalies as deviations from normal behavior. This approach attempts to deal with the variability in the network environment. In this approach, online learning is used to build a traffic profile for a given network. Traffic profiles are built using symptom-specific feature vectors such as link utilization, packet loss, and number of collisions. These profiles are then categorized by time of day, day of week, and special days, such as weekends and holidays. When newly acquired data fails to fit within some confidence interval of the developed profiles then an anomaly is declared.

In [34], normal behavior of time series data is captured as templates and tolerance limits are set based on different levels of standard deviation. These limits were tested using extensive data analysis. The authors also propose a pattern matching scheme to detect address usage anomalies by tracking each address at 5-min intervals. A template of the mean and standard deviation on the usage of each address is then used to detect anomalous behavior. The anomaly vectors from any new data are checked using the template feature vector for a given anomaly and if a match occurs it is declared as indicating a fault. Similar techniques have been used to study service anomalies [35]. Here, the anomaly detector analyzes transaction records to produce alarms corresponding to service performance anomalies.

The efficiency of this pattern matching approach depends on the accuracy of the traffic profile generated. Given a new network, it may be necessary to spend a considerable amount of time building traffic profiles. In the face of evolving network topologies and traffic conditions, this method may not scale gracefully.

D. Statistical Analysis

As the network evolves, each of the methods described above require significant recalibration or retraining. However, using online learning and statistical approaches, it is possible to continuously track the behavior of the network. Statistical analysis has been used to detect both anomalies corresponding to network failures [5], as well as network intrusions [6]. Interestingly, both of these cases make use of the standard

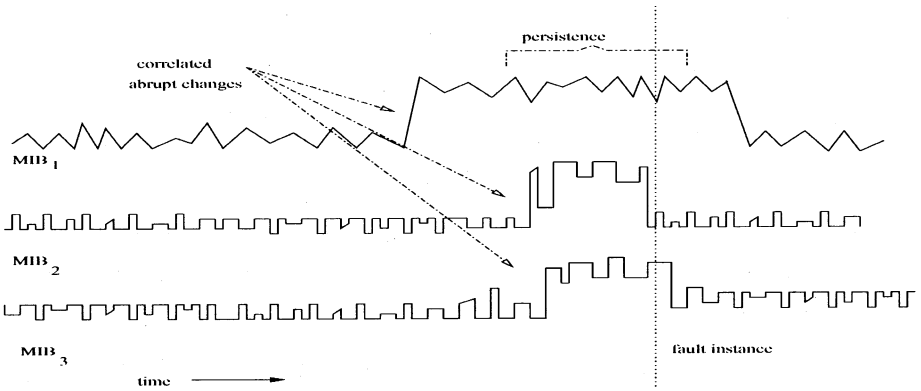

Fig. 1. Schematic representation of the proposed model for network anomalies.

sequential change point detection approach. The *Flooding Detection System*, which was proposed by the authors in [6], uses measured network data that describes TCP operations to detect SYN[1] flooding attacks. SYN flooding attacks capitalize on the limitation that TCP servers maintain all half-open connections. Once the queue limit is reached, future TCP connection requests are denied. The sequential change point detection employed here makes use of the nonparametric cumulative sum (CUSUM) method. Using this approach on trace-driven simulations, it has been shown that SYN flooding attacks can be detected with high accuracy and reasonably short detection times.

When detecting anomalies due to failures, we are confronted with the problem of detecting a host of potential failure scenarios. Each of these failure scenarios differ in their manifestations as well as their characteristics. Thus, it is necessary to obtain a rich set of network information that could cover a wide variety of network operations. The primary source for such in depth information is in the SNMP MIB data. Designing a failure detection system using MIB data necessitates the use of a general method since MIB variables exhibit varying statistical characteristics [5]. Furthermore, there is no accurate fault model available. The following section describes such a general failure detection approach from the perspective of a router.

V. Anomaly Detection Using Statistical Analysis of SNMP MIB

MIB variables provide information that is specific to the individual network devices. Since this work is on detecting network anomalies at the resolution of the device-level, this data source is sufficient. Furthermore, the widespread deployment and standardization of SNMP makes this data readily available on network devices. In this section, a statistical analysis method we developed using the theory of change detection is discussed in greater detail, along with its advantages. We also provide a detection theory-based performance criteria to evaluate the effectiveness of our approach. We present case studies of four different performance anomalies and provide some intuitive explanation on the usefulness of signal processing techniques to detect such network anomalies.

[1]SYN means packets used to synchronize sequence numbers to initiate a connenction.

A. Characterization of Network Anomalies

In statistical analysis, a network anomaly is modeled as correlated abrupt changes in network data. An abrupt change is defined as any change in the parameters of a time series that occurs on the order of the sampling period of the measurement. For example, when the sampling period is 15 s, an abrupt change is defined as a change that occurs in the period of approximately 15 s. This approach models the intuition of network managers who use hard threshold violations to generate alarms. The scheme used by most commercial management tools is similar to majority voting. However, statistical signal processing techniques reduce the number of false alarms as well as increase the probability of detection as compared with simple majority voting and hard thresholds [5].

Abrupt changes in time series data can be modeled using an auto-regressive (AR) process [8]. The assumption here is that abrupt changes are correlated in time, yet are short-range dependent. In our approach, we use an AR process of order $p = 1$ to model the data in a 5-min window. Intuitively, in the event of an anomaly, these abrupt changes should propagate through the network, and they can be traced as correlated events among the different MIB variables. This correlation property helps distinguish the abrupt changes intrinsic to anomalous situations from the random changes of the variables that are related to the network's normal function. Therefore, we propose that network anomalies can be defined by their effect on network traffic as follows: *Network anomalies are characterized by traffic-related MIB variables undergoing abrupt changes in a correlated fashion*. A pictorial representation of this is provided in Fig. 1.

Using the above model for network anomalies, the anomaly detection problem can be posed as follows:

Given a sequence of traffic-related MIB variables sampled at a fixed interval, generate a network health function that can be used to declare alarms corresponding to anomalous network events.

To detect anomalies from the perspective of a router, we focus on the *ip* layer. Three MIB variables are chosen from the *ip* MIB group. These variables represent a cross section of the traffic seen at the router. The variable *ipIR* (which stands for In Receives), represents the total number of datagrams received from all the interfaces of the router. *ipIDe* (which stands for In Delivers), represents the number of datagrams correctly delivered to the higher layers as this node was their final destination. *ipOR*

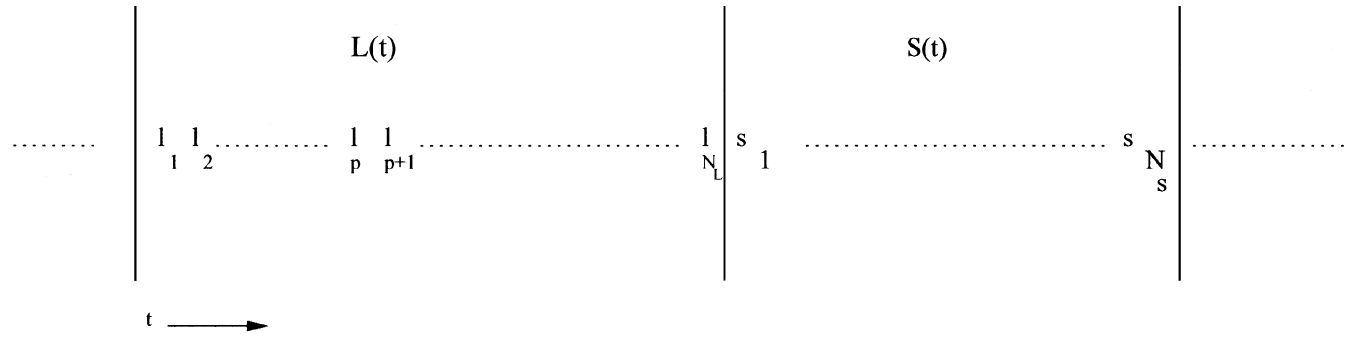

Fig. 2. Contiguous piecewise stationary windows $L(t)$: Learning window $S(t)$: Test window.

(which stands for Out Requests) represents the number of datagrams passed on from the higher layers of the node to be forwarded by the *ip* layer. The traffic associated with the *ipIDe* and *ipOR* variables comprise only a fraction of the entire network traffic. However, in the event of an anomaly, these are relevant variables since the router does some route table processing, which would be reflected in these variables. The three MIB variables chosen are not strictly independent. The average cross correlation of *ipIR* with *ipIDe* is 0.08 and with *ipOR* is 0.05. The average cross correlation between *ipOR* and *ipIDe* is 0.32.

B. Abrupt Change Detection

In our statistical approach, the network health function is obtained using a combination of abnormality indicators from the individual MIB variables. The abnormality in the MIB data is determined by detecting abrupt changes in their statistics. Since the statistical distribution of the individual MIB variables are significantly different it is difficult to do joint processing of these variables. Therefore, the abrupt changes in each of the MIB variables is first obtained. Change detection is done using a hypothesis test based on the generalized likelihood ratio (GLR) [36]. This test provides an abnormality indicator that is scaled between 0 and 1.

Abrupt changes are detected by comparing the variance of the residuals obtained from two adjacent windows of data that are referred to as the learning $(L(t))$ and test $(S(t))$ windows, as shown in Fig. 2. Residuals are obtained by imposing an AR model on the time series data in each of the windows. The likelihood ratio for a single variable is obtained as shown in (1)

$$\eta = \frac{\hat{\sigma}_L^{-\acute{N}_L}\hat{\sigma}_S^{-\acute{N}_S}}{\hat{\sigma}_L^{-\acute{N}_L}\hat{\sigma}_S^{-\acute{N}_S} + \hat{\sigma}_P^{-(\acute{N}_L+\acute{N}_S)}} \tag{1}$$

where $\hat{\sigma}_L$ and $\hat{\sigma}_S$ are the variance of the residual in the learning window and the test window, respectively, $\acute{N}_L = N_L - p$, where p is the order of the AR process, and N_L is the length of the learning window. Similarly, $\acute{N}_S = N_S - p$, where N_S is the length of the test window. $\hat{\sigma}_P$ is the pooled variance of the learning and test windows. The abnormality indicators thus obtained from the individual MIB variables are collected to form an abnormality vector $\vec{\psi}(t)$. The abnormality vector $\vec{\psi}(t)$ is a measure of the abrupt changes in normal network behavior.

C. Combining the Abnormality Vectors: $\vec{\psi}(t)$

The individual abnormality vectors must be combined to provide a health function. This network health function is obtained by incorporating the spatial dependencies between the abrupt changes in the individual MIB variables. This is accomplished using a linear operator A. Such operators are frequently used in quantum mechanics [37]. The linear operator is designed based on the correlation between the chosen MIB variables. In particular, the quadratic functional

$$f(\vec{\psi}(t)) = \vec{\psi}(t)A\vec{\psi}'(t) \tag{2}$$

is used to generate a continuous scalar indicator of network health. This network health indicator is interpreted as a measure of abnormality in the network, as perceived by the specific node. The network health indicator is bounded between 0 and 1 by an appropriate transformation of the operator A [5]. In the network health function, a value of 0 represents a healthy network, and a value of 1 represents maximum abnormality in the network.

The operator matrix A is an $M \times M$ matrix (M is the number of MIB variables). In order to ensure orthogonal eigenvectors that form a basis for R^M and real eigenvalues, the matrix A is designed to be symmetric. Thus, it has M orthogonal eigenvectors with M real eigenvalues. A subset of these eigenvectors can be identified to correspond to anomalous states in the network [5]. If λ_N and λ_M are the minimum and maximum eigenvalues that correspond to these anomalous states, the problem of detecting network anomalies can then be expressed as

$$t_a = \inf\{t\colon f(\vec{\psi}(t)) \geq \lambda_N\} \tag{3}$$

where t_a is the earliest time at which the functional $f(\vec{\psi}(t))$ exceeds λ_N. By virtue of the design of the operator matrix (discussed below), the function $f(\vec{\psi}(t))$ has an upper bound

$$f(\vec{\psi}(t)) \leq \lambda_M = 1. \tag{4}$$

Each time the condition expressed in (3) is satisfied, we have a declaration of an anomalous condition.

Design of the Operator Matrix A: The primary goal of the operator matrix is to incorporate the correlation between the individual components of the abnormality vector. The abnormality vector $\vec{\psi}(t)$ is a $(1 \times m)$ vector with components

$$\vec{\psi}(t) = [\,\psi_1(t) \quad \cdots \quad \psi_m(t)\,] \tag{5}$$

where each component of this vector corresponds to the abnormality associated with the individual MIB variables, as obtained from the likelihood ratio test. In order to complete the basis set so that all possible states of the system are included, an additional component $\psi_0(t)$ that corresponds to the normal functioning of the network is added. The final component allows for proper normalization of the input vector. The new input vector $\vec{\psi}(t)$

$$\vec{\psi}(t) = \alpha\,[\,\psi_1(t) \quad \cdots \quad \psi_m(t) \quad \psi_0(t)\,] \tag{6}$$

is normalized with α as the normalization constant ($\langle\vec{\psi}(t), \vec{\psi}(t)\rangle = 1$). By normalizing the input vectors the expectation of the observable $E(\lambda)$ of the operator can be constrained to lie between 0 and 1 [see (18)].

The appropriate operator matrix A will therefore be $(M+1)\times(M+1)$. We design the operator matrix to be Hermitian in order to have an eigenvector basis. Taking the normal state to be uncoupled to the abnormal states, we get a block diagonal matrix with an $M \times M$ upper block A_{upper} and a 1×1 lower block, as in (7), shown at the bottom of the page.

The elements of the upper block of the operator matrix A_{upper} are obtained as follows: When $i \neq j$, we have

$$A_{\text{upper}}(i, j) = |\langle\psi_i(t), \psi_j(t)\rangle| \tag{8}$$

$$= \frac{1}{T}\left|\sum_{t=1}^{T}\psi_i(t)\psi_j(t)\right| \tag{9}$$

which is the the ensemble average of the two point spatial cross-correlation of the abnormality vectors estimated over a time interval T [38]. For $i = j$, we have

$$A_{\text{upper}}(i, i) = 1 - \sum_{j\neq i} A(i, j). \tag{10}$$

Using this transformation ensures that the maximum eigenvalue of the matrix A_{upper} is 1.

The $a_{(M+1)(M+1)}$ element indicates the contribution of the healthy state to the indicator of abnormality for the network node. Since the healthy state should not contribute to the abnormality indicator, the component $a_{(M+1)(M+1)}$ is assigned as ϵ, which, in the limit, tends to 0. Therefore, for the purpose of detecting faults, we only consider the upper block of the matrix A_{upper}.

The entries of the matrix describe how the operator causes the components of the input abnormality vector to mix with each other. The matrix A_{upper} is symmetric, real, and the elements are non-negative and, hence, the solution to the characteristic equation

$$A_{\text{upper}}\vec{\phi} = \lambda\vec{\phi} \tag{11}$$

consists of orthogonal eigenvectors $\{\vec{\phi}_i\}_{i=1}^{M}$ with eigenvalues $\{\lambda_i\}_{i=1}^{M}$. The eigenvectors obtained are normalized to form an orthonormal basis set. The first M components of $\vec{\psi}(t)$ can therefore be decomposed as a linear combination of the eigenvectors of A_{upper}, namely, $\vec{\phi}_i$

$$\vec{\psi}'(t) = \sum_{i=1}^{M} c_i\vec{\phi}_i. \tag{12}$$

The $(M+1)$th component of $\vec{\psi}(t)$ is present only for normalization purposes and is therefore omitted in the subsequent calculation of network abnormality. Incorporating the spatial dependencies through the operator transforms the abnormality vector $\vec{\psi}(t)$ as

$$A_{\text{upper}}\vec{\psi}'(t) = \sum_{i=1}^{M} c_i\lambda_i\vec{\phi}_i. \tag{13}$$

Here, c_i measures the degree to which a given abnormality vector falls along the ith eigenvector. This value c_i can be interpreted as a probability amplitude and c_i^2 as the probability of being in the ith eigenstate.

A subset of the eigenvectors $\{\vec{\phi}_i\}_{i=N}^{M}$, where $1 < N < M$, is called the fault vector set and can be used to define a faulty region. The fault vectors are chosen based on the magnitude of the components of the eigenvector. The eigenvector that has components proportional to $[1\ 1\ 1]$ (since it is normalized) is identified as the most faulty vector since it corresponds to maximum abnormality in all its components. Furthermore, based on our fault model of correlated abrupt changes, the eigenvector proportional to the $[1\ 1\ 1]$ vector signifies the maximum correlation between all the abnormality indicators.

If a given input abnormality vector can be completely expressed as a linear combination of the fault vectors

$$\vec{\psi}'(t) = \sum_{i=N}^{M} c_i\vec{\phi}_i \tag{14}$$

then we say that the abnormality vector falls in the fault domain. The extent to which any given abnormality vector lies in the fault domain can be obtained in the following manner: Since

$$A = \begin{bmatrix} a_{11} & a_{12} & . & . & a_{1(M-1)} & a_{1M} & 0 \\ a_{21} & a_{22} & . & . & a_{2(M-1)} & a_{2M} & 0 \\ . & . & . & . & . & . & 0 \\ . & . & . & . & . & . & 0 \\ a_{M1} & a_{M2} & a_{M3} & a_{M.} & a_{M(M-1)} & a_{MM} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{(M+1)(M+1)} \end{bmatrix} \tag{7}$$

any general abnormality vector $\vec{\psi}(t)$ is normalized, we have the following condition:

$$\sum_{i=1}^{M} c_i^2 = 1. \tag{15}$$

As there are M different values for c_i, an average scalar measure of the transformation in the input abnormality vector is obtained by using the quadratic functional

$$f(\vec{\psi}(t)) = \vec{\psi}(t) A \vec{\psi}'(t). \tag{16}$$

Using (16) and the fact that $\langle \vec{\phi}_i \vec{\phi}_j \rangle = \delta_{ij}$ the Kronecker delta, we have

$$\vec{\psi}(t) A \vec{\psi}'(t) = \sum_{i=1}^{M} c_i^2 \lambda_i \tag{17}$$

$$= E(\lambda). \tag{18}$$

The measure $E(\lambda)$ is the indicator of the average abnormality in the network as perceived by the node.

Now, consider an input abnormality vector that lies completely in the fault domain. Using the condition in (15), we obtain a bound for $E(\lambda)$ as

$$\min_{\lambda_i \in \{\lambda_N, \lambda_{N+1}, \ldots \lambda_M\}} (\lambda_i) \le E(\lambda) \le \max_{\lambda_i \in \{\lambda_N, \lambda_{N+1}, \ldots, \lambda_M\}} (\lambda_i) \tag{19}$$

where λ_i are the eigenvalues corresponding to the set of fault vectors. Thus, using these bounds on the functional $f(\vec{\psi}(t))$, an alarm is declared when

$$E(\lambda) > \min_{\lambda_i \in \{\lambda_N, \lambda_{N+1}, \ldots, \lambda_M\}} (\lambda_i). \tag{20}$$

Note that since the maximum eigenvalue of A_{upper} is 1, $E(\lambda) \le 1$. The maximum eigenvalue by design is associated with the most faulty eigenvector.

1) Performance Evaluation of the Statistical Analysis Method: The performance of the statistical algorithm is expressed in terms of the prediction time T_p and the mean time between false alarms T_f. Prediction time is the time to the anomalous event from the nearest alarm preceding it. A true anomaly prediction is identified by a declaration that is correlated with an accurate fault label from an independent source such as *syslog* messages and/or trouble tickets. Therefore, anomaly prediction implies two situations: a) In the case of predictable anomalies such as file server failures and network access problems, true prediction is possible by observing the abnormalities in the MIB data, and b) in the case of unpredictable anomalies such as protocol implementation errors, early detection is possible as compared with the existing mechanisms such as *syslog* messages and trouble reports. Any anomaly declaration that did not coincide with a label was declared a false alarm. The quantities used in studying the performance of the agent are depicted in Fig. 3. τ is the number of lags used to incorporate the persistence criteria in order to declare alarms corresponding to fault situations. Persistence criteria implies that for an anomaly to be declared, the alarms must persist for τ consecutive time lags.

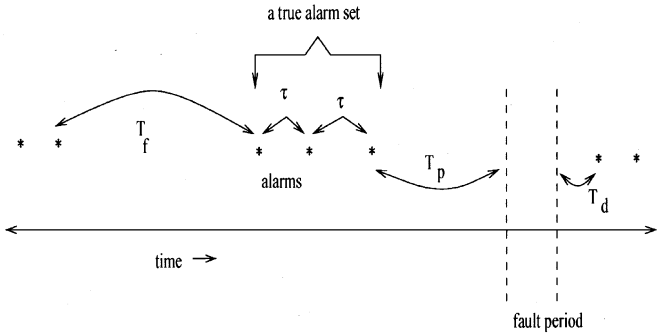

Fig. 3. Quantities used in performance analysis.

TABLE I
SUMMARY OF RESULTS OBTAINED USING THE STATISTICAL TECHNIQUE

Anomaly Type	Avg.Prediction Time T_p(mins)	Avg.Detection Time T_d(mins)	Avg.Mean Time Between False Alarms T_f(mins)
File server failures	26	30	455
Network Access problems	26	23	260
Protocol Error	15	-	-
Runaway process	1	-	235

In some cases, alarms are obtained only after the anomaly has occurred. In these instances, we only detect the problem. The time for detection T_d is measured as the time elapsed between the occurrence of the anomaly and the declaration of the alarm. There are also some instances where alarms were obtained both preceding and after the anomaly. In these cases, the alarms that follow are attributed to the *hysteresis effect* of the anomaly.

2) Application of the Statistical Approach to Network Data From SNMP MIB: The statistical techniques described above have been successfully used to detect eight out of nine file server failures in the campus network and 14 out of 15 file server failures on the enterprise network. Interestingly, the same algorithm with no modifications was able to detect all eight instances of network access problems, one protocol implementation error, and one run-away process on an enterprise network.[2] Thus, we see that the statistical techniques are able to easily generalize to detect different types of network anomalies. Further evidence of this is provided by using case studies from real network failures. A summary of the results obtained using the statistical techniques is provided in Table I. It was observed that a plain majority voting scheme on the variable level abnormality indicators was able to detect only file server failures and not any of the other three types of failures [13].

3) Case Studies: In this section, we present examples of network failures obtained from two different production networks: an enterprise network and a campus network. Both these networks were being actively monitored and were well designed to meet customer requirements. The types of anomalies observed

[2]In the collected data set, there was only one instance each of protocol implementation error and runaway process

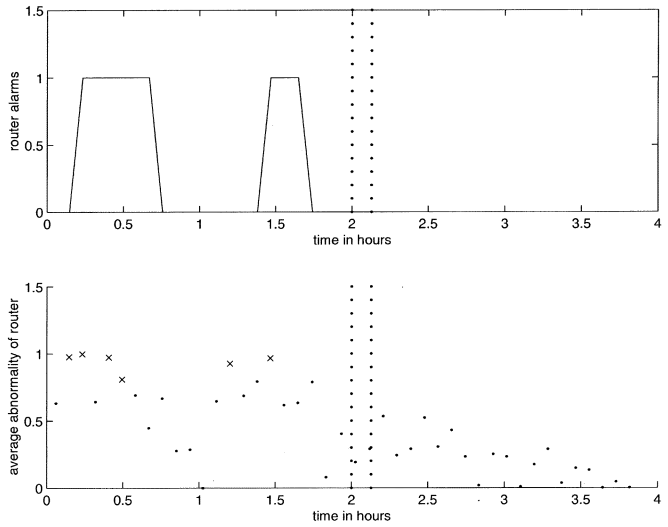

Fig. 4. Case (1) file server failure. Average abnormality at the router.

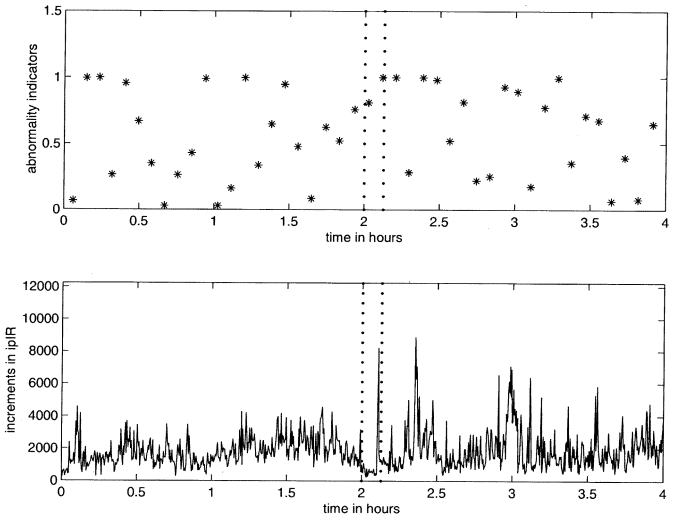

Fig. 5. Case (1) file server failure. Abnormality indicator of *ipIR*.

were the following: file server failures, protocol implementation errors, network access problems, and runaway processes [5]. Most of these anomalous events were due to abnormal user activity, except for protocol implementation errors. However, all of these cases did affect the normal characteristics of the MIB data and impaired the functionality of the network. We show that by using signal processing techniques, it is possible to detect the presence of network anomalies prior to their being detected by the existing alarm systems such as *syslog*[3] messages and trouble tickets.

Case Study (1): File Server Failure Due to Abnormal User Behavior: In this case study, we describe a scenario corresponding to a file server failure on one of the subnets of the campus network. Twelve machines on the same subnet and 24 machines outside the subnet reported the problem via *syslog* messages. The duration of the failure was from 11:10 am to 11:17 am (7 min) on December 5, 1995, as determined by the *syslog* messages. The cause of the file server failure was an unprecedented increase in user traffic (*ftp* requests) due to the release of a new web-based software package.

This case study represents a predictable network problem where the traffic related MIB variables show signs of abnormality before the occurrence of the file server failure. The fault was predicted 21 min before the server crash occurred. Figs. 4–7 show the output of the statistical algorithm at the router and in the individual *ip* layer variables. The fault period is shown by vertical dotted lines. In Fig. 4, for router health, the "x" denotes the alarms that correspond to input vectors that are abnormal.

The variable level indicators capture the trends in abnormality. Note that there is a drop in the mean level of the traffic in the *ipIR* variable immediately prior to the failure. Among the three variables considered, the variables ipOR and ipIDe are the most well-behaved, i.e., not bursty, and the ipIR variable is bursty. Because the *ipIDE* and *ipOR* variables are less bursty, they lend easily to conventional signal processing techniques (see Figs. 6 and 7).

[3]*Syslog* messages are system generated messages in response to some failure.

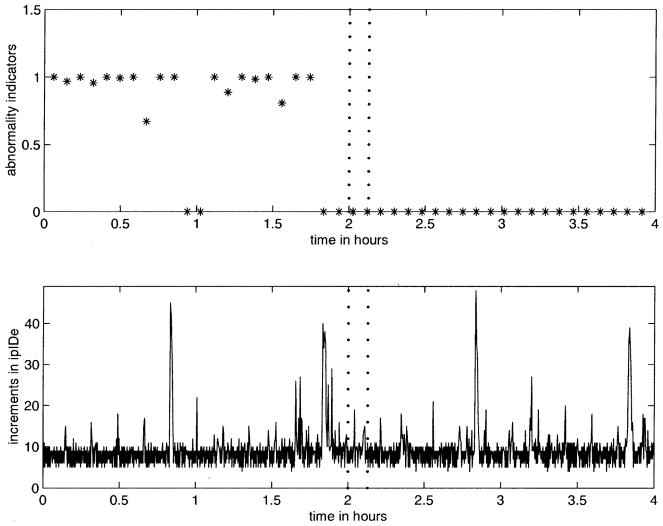

Fig. 6. Case (1) file server failure. Abnormality indicator of *ipIDe*.

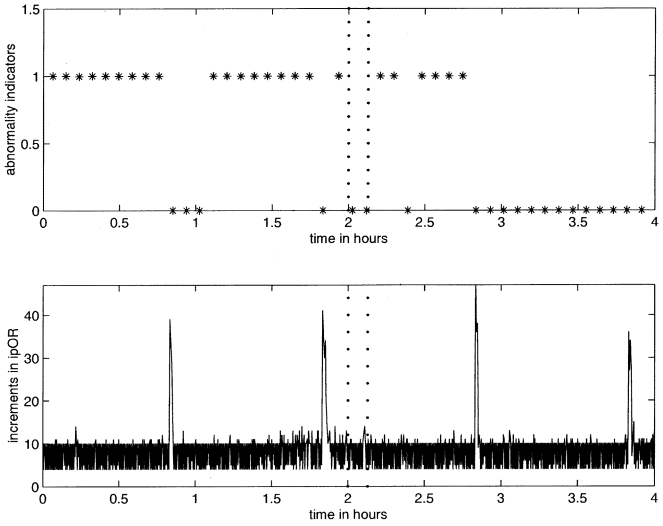

Fig. 7. Case (1) file server failure. Abnormality indicator of *ipOR*.

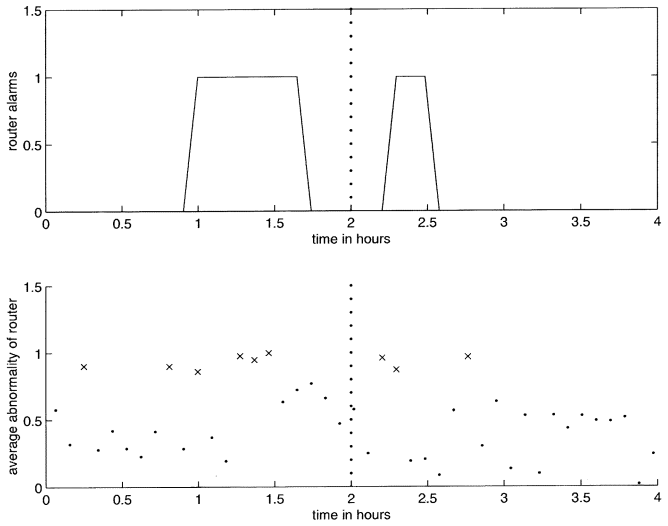

Fig. 8. Case (2) protocol error. Average abnormality at the router.

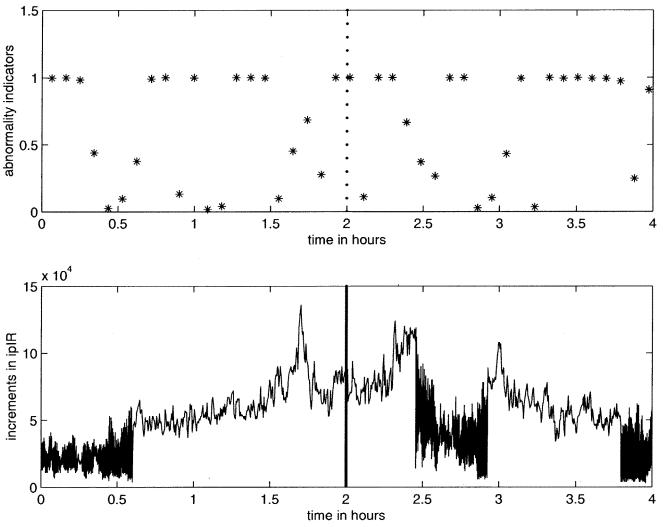

Fig. 9. Case (2) protocol error. Abnormality indicator of *ipIR*.

However, the combined information from all the three variables is able to capture the traffic behavior at the time of the anomaly, as shown in Fig. 4. There are very few alarms at the router level, and the mean time between false alarms in this case was 1032 min (approx 17 h).

Case Study (2): Protocol Implementation Errors: This case study is one where the fault itself is not predictable but the symptoms of the fault can be observed. Typically, a protocol implementation error is an undetected software problem that could get triggered by a specific set of circumstances on the network. One such fault detected on the enterprise network was that of a super server *inetd* protocol error. The super server is the server that listens for incoming requests for various network servers, thus serving as a single daemon that handles all server requests from the clients. The existence of the fault was confirmed by *syslog* messages and trouble tickets. *Syslog* messages reported an *inetd* error. In addition, other faulty daemon process messages were also reported during this time. Presumably, these faulty daemon messages are related to the super server protocol error. During the same time interval, trouble tickets also reported problems such as the inability to connect to the web server, send mail, or print on the network printer, as well as difficulty in logging onto the network. The super server protocol problem is of considerable interest since it affected the overall performance of the network for an extended period of time.

The prediction time of this network failure relative to the *syslog* messages was 15 min. The existing trouble ticketing scheme only responded to the fault situation and, hence, detected the failure after its onset.

Figs. 8–11 show the alarms generated at the router and the abnormality indicators at the individual variables. Again, the combined information from all the three variables captures the abnormal behavior leading up to the fault. Note that since this problem was a network-wide failure, the *ipIR* and the *ipIDe* variables show significant changes around the fault region. Since there was a distinct change in behavior in two of the three variables, the combined abnormality at the router was less prone to error. Thus, there were no false alarms reported in this data set, but rather, persistent alarms were observed just before the fault.

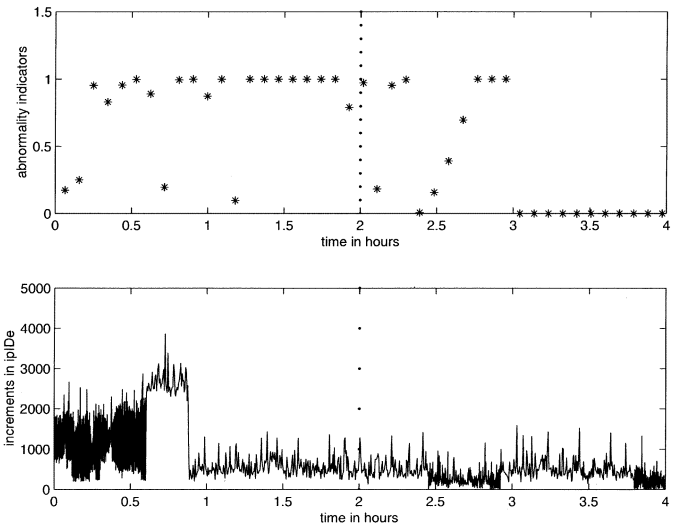

Fig. 10. Case (2) protocol error. Abnormality indicator of *ipIDe*.

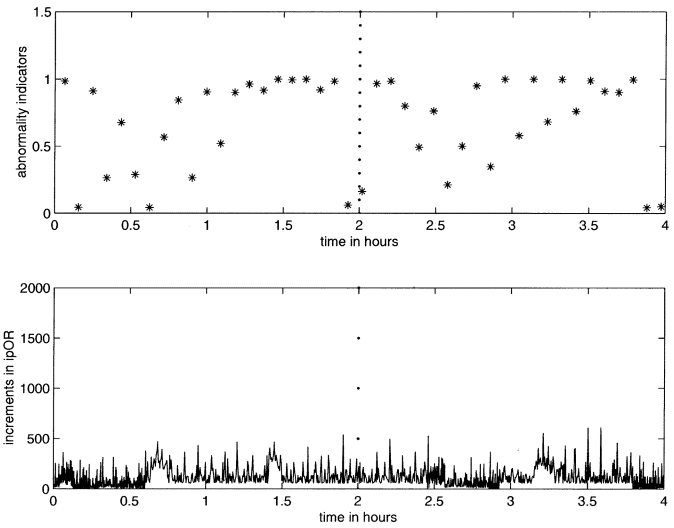

Fig. 11. Case (2) protocol error. Abnormality indicator of *ipOR*.

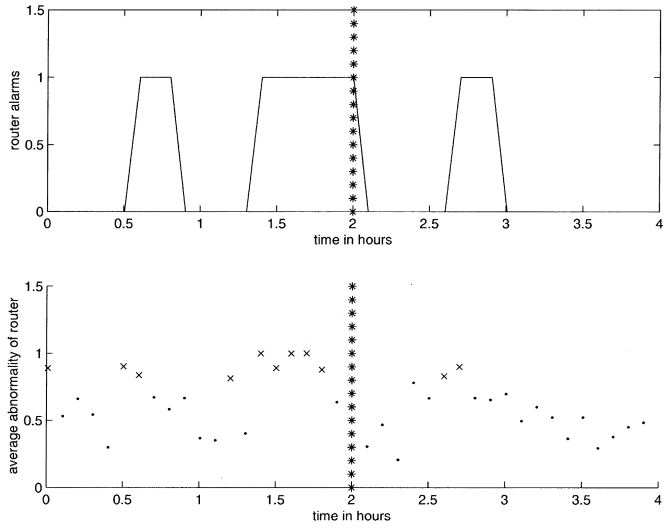

Fig. 12. Case (3) network access. Average abnormality at the router.

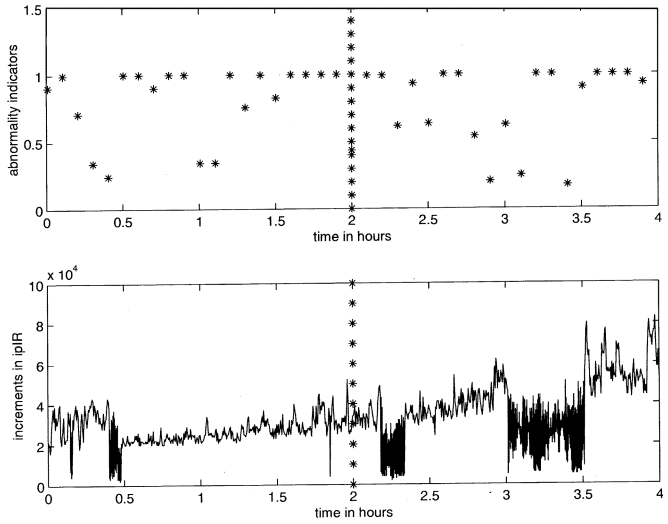

Fig. 13. Case (3) network access. Abnormality indicator of *ipIR*.

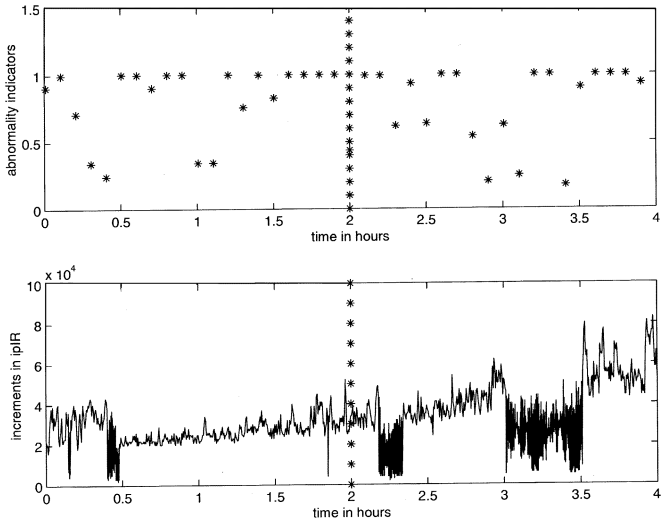

Fig. 14. Case (3) network access. Abnormality indicator of *ipIDe*.

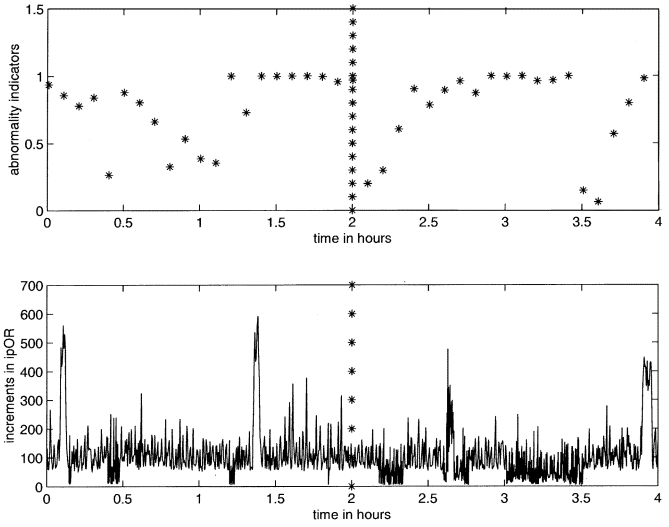

Fig. 15. Case (3) network access. Abnormality indicator of *ipOR*.

Case Study (3): Network Access Problems: Network access problems were reported primarily in the trouble tickets. These faults were often not reported by the *syslog* messages. Due to the inherent reactive nature of trouble tickets, it is hard to determine the exact time when the problem occurred. The trouble reports received ranged from the network being slow to inaccessibility of an entire network domain. The prediction time was 6 min. The mean time between false alarms was 286 min (4 h and 46 min). Figs. 12–15 show the alarms obtained at the router level as well as the abnormality indicators at the variables. Note that the *ipIR* variable shows a gradual increase in the baseline as it nears the fault region.

Case Study (4): Runaway Processes: A runaway process is an example of high network utilization by some culprit user that affects network availability to other users on the network. A runaway process is an example of an unpredictable fault but whose symptoms can be used to detect an impending failure. This is a commonly occurring problem in most computation-oriented network environments. Runaway processes are known to be a security risk to the network. This fault was reported by the trouble tickets but much after the network had run out of process identification numbers. In spite of having a large number of *syslog* messages generated during this period, there was no clear message indicating that a problem had occurred. The prediction time was 1 min, and the mean time between false alarms was 235 min (about 4 h). Figs. 16–19 show the performance of the statistical technique in the detection of the runaway process. In this scenario, the *ipIR* variable shows a noticeable change in mean immediately prior to the fault being detected by conventional schemes such as *syslog* and trouble tickets. However, the statistical analysis method captures this anomalous behavior ahead of the *syslog* reports, as seen in the *ipIR* variable, as well as in the combined router indicator.

4) Effectiveness of Statistical Techniques: The effectiveness of the statistical approach can be seen from its ability to distinguish between different failures. Once an alarm is obtained, using the behavior of the abnormality indicators 1 h prior to the anomaly time, we were able to identify the nature of the anomaly [39]. Since network anomalies typically cause deviations from

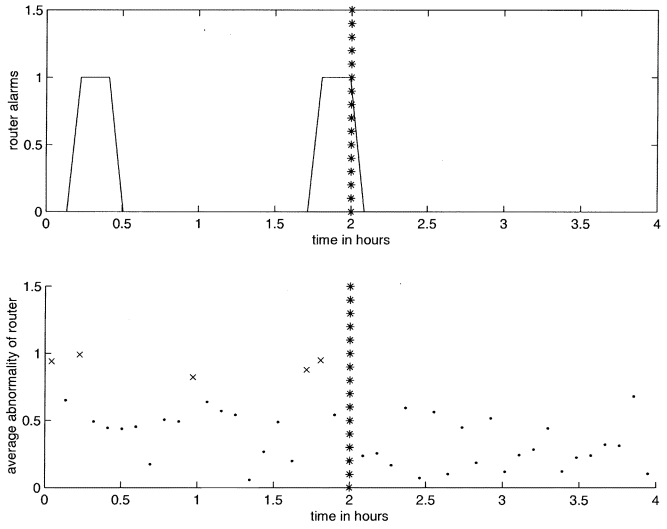

Fig. 16. Case (4) runaway process. Average abnormality at the router.

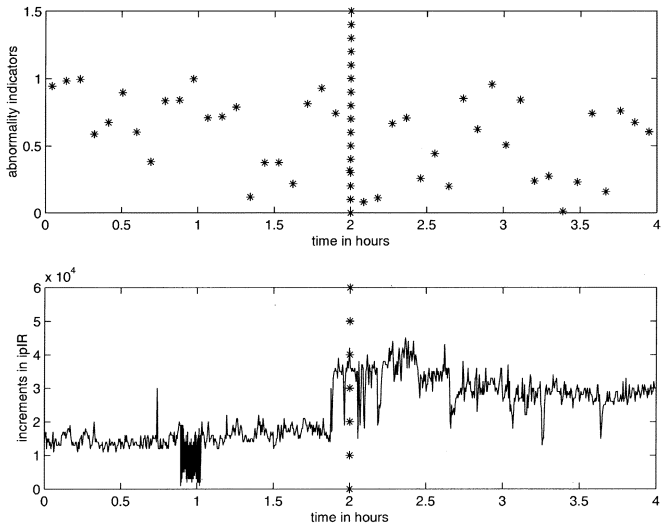

Fig. 17. Case (4) runaway process: Abnormality indicator of *ipIR*.

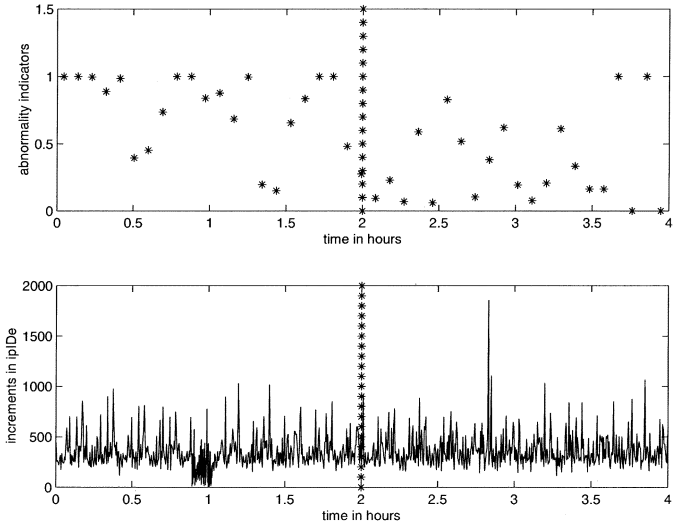

Fig. 18. Case (4) runaway process: Abnormality indicator of *ipIDe*.

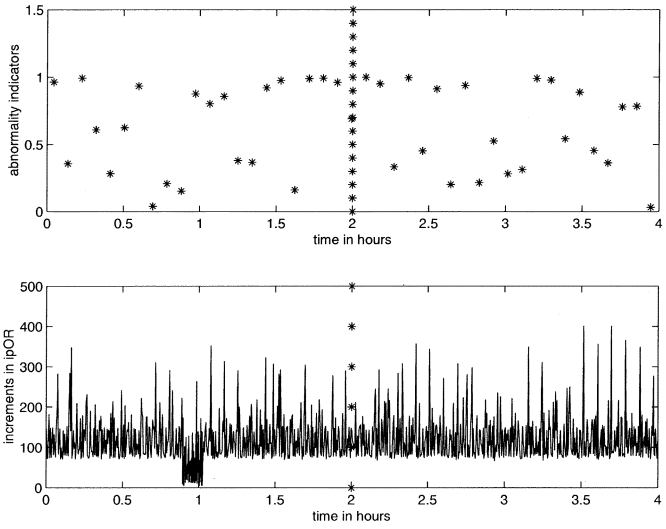

Fig. 19. Case (4) runaway process. Abnormality indicator of *ipOR*.

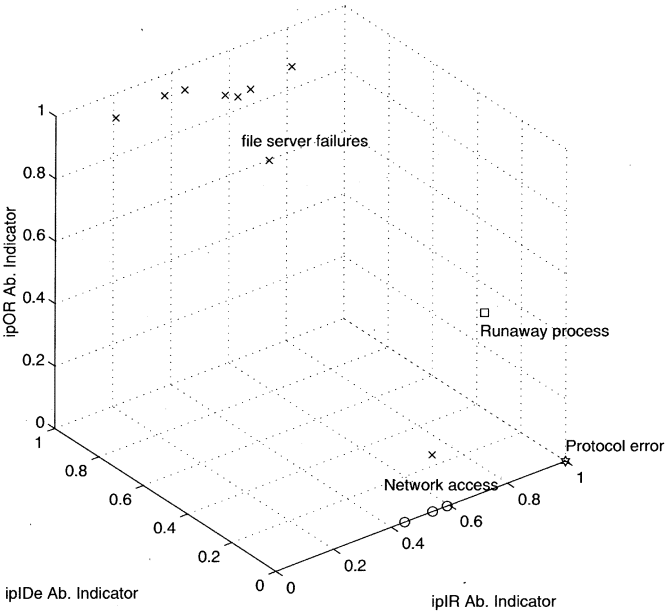

Fig. 20. Classification of faults using the average (over 1 h) of the abnormality vector. x: file server failures; o: network access problems; *star*: protocol error; *square*: runaway process.

normal behavior in the measured data prior to the actual manifestation of the failure, it is possible to observe in advance the failure signatures corresponding to the impending failure. The average values of the abnormality indicators computed over this period are used to locate the anomaly in the problem space defined by the anomaly vectors. As shown in Fig. 20, the four anomaly types are clustered in different areas of the problem space. The Euclidean distance between the center vector of the file server failures and the network access problems is approximately 1.16. The standard deviation for the network file server cluster is 0.43, and that for the network access cluster is 0.07. These results show that the two clusters do not overlap.

We have limited data on the other two types of anomalies, but it is interesting to note that they are distinct from both file server failures and network access problems. In the case of file server

failures (shown as "x" in Fig. 20), the abnormality in the *ipOR* and *ipIDe* variables are much more significant than in *ipIR*. On the contrary, network access problems (shown as "o" in Fig. 20) are expressed only in the *ipIR* variable. The fact that these faults were predicted or detected by the quadratic functional $f(\psi(t))$, which isolates a very narrow region of the problem space, suggests that *the abnormality in the feature vectors increases as the fault event approaches.*

D. More Related Work

In our early work, we used duration filter heuristics to obtain real-time alarms for anomaly detection in conjunction with MIB variables [40]. In [41] and [42], Bayesian Belief networks were used together with MIB variables for nonreal time detection of anomalies. Signal processing and statistical approaches are also gaining more applications in detecting anomalies related to malicious events [6] and network services such as VoIP [43].

VI. Discussion

The statistical signal processing tools presented here are versatile in their applicability to time series data obtained from other sources of network information such as probing and packet filtering techniques. In [5], the authors present the application of the methods described above to the *interface* layer variables. The statistical techniques using the CUSUM approach [6] and the likelihood ratio test have been shown to be applicable to two sources of traffic traces and to two different network topologies, respectively. In this section, we provide a discussion on some issues relevant to using the statistical analysis methods on measured network data.

It was observed that the use of fine-grained data significantly improves detection times since the confidence of statistical analysis techniques is only constrained by sample sizes. For example, in the work presented here, using MIB variables, a sampling frequency of 15 s was used. However, it would be possible to obtain a finer sampling frequency if the polling entities were optimally located [44]. The primary limiting factor to increasing the polling frequency in the case of MIB data is the priority given to processing SNMP packets by the router being polled. Independent of the load on the router, it was observed that typically there is a 20-ms delay in poll responses.

In terms of time synchronization, there is a requirement that the chosen feature variables are all of the same time granularity. To the best of our knowledge, the problem of simultaneously handling multiple feature variables with different time granularities is still an open problem. Time synchronization issues also arise from the difference in time stamps between the polled entity and the polling node. This issue can be resolved using the network time protocol (NTP) MIB, which provides a uniform network-wide time stamp. There is also work done on designing algorithms to adjust for clock skew [45].

The operator matrix described here can be easily applied to any other system that can be described using time series data. The number of feature variables define the eigenstates of the operator matrix. Therefore, if a broader set of anomalies must be detected, then additional feature vectors must be added. Thus, the scope of the detector is primarily limited by the dimensionality of the operator matrix. The construction of the operator will be similar to the methods described in this paper.

VII. Conclusion and Future Directions

This paper provides a review of the area of network anomaly detection. Based on the case studies presented, it is clear that there is a significant advantage in using the wide array of signal processing methods to solve the problem of anomaly detection. A greater synergy between the networking and signal processing areas will help develop better and more effective tools for detecting network anomalies and performance problems. A few of the open issues in the application of statistical analysis methods to network data are discussed below.

The change detection approach presented in this paper makes the assumption that the traffic variables are quasistationary. However, it was observed that some of the MIB variables exhibit nonstationary behavior. A method to quantify the bursty behavior of the MIB variables could lead to using better models for the traffic and improve the false alarm rates at the variable level, thus increasing the optimality of statistical methods. An accurate estimation of the Hurst parameter for the MIB variables was difficult due to the lack of high-resolution data [46]. Often, the alarms corresponding to anomalous events have to check for a persistence criteria to reduce the number of false alarms. The major reason for false alarms come from the abnormality indicators obtained for the bursty variables such as *ipIR*. Increasing the order of the AR model may help in reducing the false alarm rate, but there is a tradeoff since the resolution in detection time would decrease. Another open issue is that not all abrupt changes in MIB data correspond to network anomalies. Thus, the accurate definition of the nature of the abrupt changes corresponding to anomalous events is essential for increased detection accuracy.

The SNMP protocol runs over the UDP transport mechanism and, therefore, could result in lost SNMP queries and responses from the devices. From the signal processing perspective, this could result in missing samples, and it is necessary to design efficient algorithms to deal with missing data.

From the study presented here, it is clear that signal processing techniques can add significant advantage to existing network management tools. By improving the capability of predicting impending network failures, it is possible to reduce network downtime and increase network reliability. Rigorous statistical analysis can lead to better characterization of evolving network behavior and eventually lead to more efficient methods for both failure and intrusion detection.

Acknowledgment

The authors would like to thank D. Hollinger, N. Schimke, R. Collins, and C. Hood for their generous help with the campus data collection. They also acknowledge Lucent Technologies for providing data on the enterprise network and thank K. Vastola for helpful discussions on the topic.

References

[1] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Trans. Networking*, vol. 2, pp. 1–15, Feb. 1994.

[2] J. L. Vehel, E. Lutton, and C. Tricot, *Fractals in Engineering: From Theory to Industrial Applications*. New York: Springer-Verlag, 1997, pp. 185–202.
[3] N. G. Duffield, F. L. Presti, V. Paxson, and D. Towsley, "Inferring link loss using striped unicast probes," in *Proc. IEEE INFOCOM*, 2001.
[4] M. Coates, A. Hero, R. Nowak, and B. Yu, "Internet tomography," *IEEE Signal Processing Mag.*, vol. 19, pp. 47–65, May 2002.
[5] M. Thottan, "Fault detection in ip networks," Ph.D. dissertation, Rensselaer Polytech. Inst., Troy, NY, 2000. Under patent with RPI.
[6] H. Wang, D. Zhang, and K. G. Shin, "Detecting syn flooding attacks," in *Proc. IEEE INFOCOM*, 2002.
[7] O. Akashi, T. Sugawara, K. Murakami, M. Maruyama, and N. Takahashi, "Multiagent-based cooperative inter-as diagnosis in encore," in *Proc. NOMS*, 2000.
[8] M. Basseville and I. Nikiforov, *Detection of Abrupt Changes, Theory and Application*. Englewood Cliffs: Prentice-Hall, 1993.
[9] T. Ye, S. Kalyanaraman, D. Harrison, B. Sikdar, B. Mo, H. T. Kaur, K. Vastola, and B. Szymanski, "Network management and control using collaborative on-line simulation," in *Proc. CNDSMS*, 2000.
[10] I. Norros, "A storage model with self-similar input," *Queueing Syst.*, vol. 16, pp. 387–396, 1994.
[11] R. H. Reidi, M. S. Crouse, V. J. Ribeiro, and R. G. Baraniuk, "A multifractal wavelet model with application to network traffic," *IEEE Trans. Inform. Theory*, vol. 45, pp. 992–1018, Mar. 1999.
[12] A. C. Gilbert, W. Willinger, and A. Feldmann, "Scaling analysis of conservative cascades with applications to network traffic," *IEEE Trans. Inform. Theory*, vol. 45, pp. 971–991, Mar. 1999.
[13] M. Thottan and C. Ji, "Using network fault predictions to enable ip traffic management," *J. Network Syst. Manage.*, 2000.
[14] R. Maxion and F. E. Feather, "A case study of ethernet anomalies in a distributed computing environment," *IEEE Trans. Reliability*, vol. 39, pp. 433–443, Oct. 1990.
[15] G. Vigna and R. A. Kemmerer, "Netstat: A network based intrusuion detection approach," in *Proc. ACSAC*, 1998.
[16] J. Yang, P. Ning, X. S. Wang, and S. Jajodia, "Cards: A distributed system for detecting coordinated attacks," in *Proc. SEC*, 2000, pp. 171–180.
[17] S. Savage, D. Wetherall, A. R. Karlin, and T. Anderson, "Practical network support for ip traceback," in *Proc. ACM SIGCOMM*, 2000, pp. 295–306.
[18] CAIDA. Cooperative association for internet data analysis. [Online]. Available: http://www/caida.org/Tools.
[19] W. S. Cleveland, D. Lin, and D. X. Sun, "Ip packet generation: Statistical models for tcp start times based on connection rate super position," in *Proc. ACM SIGMETRICS*, 2000, pp. 166–177.
[20] R. Caceres, N. G. Duffield, A. Feldmann, J. Friedmann, A. Greenberg, R. Greer, T. Johnson, C. Kalmanek, B. Krishnamurthy, D. Lavelle, P. P. Mishra, K. K. Ramakrishnan, J. Rexford, F. True, and J. E. van der Merwe, "Measurement and analysis of ip network usage and behavior," *IEEE Commun. Mag.*, pp. 144–151, May 2000.
[21] P. Aukia, M. Kodialam, P. Koppol, T. Lakshman, H. Sarin, and B. Suter, "Rates: A server for mpls traffic engineering," *IEEE Network Magazine*, pp. 34–41, Mar./Apr. 2000.
[22] W. Stallings, *SNMP, SNMPv2, and CMIP The Practical Guide to Network Management Standards*, 5th ed. Wellesley, MA: Addison-Wesley, 1994.
[23] K. McCloghrie and M. Rose, "Management information base for network management of tcp/ip-based internets: Mib 2,", RFC1213, 1991.
[24] M. Rose, *The Simple Book: An Introduction to Internet Management*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1996.
[25] T. D. Ndousse and T. Okuda, "Computational intelligence for distributed fault management in networks using fuzzy cognitive maps," in *Proc. IEEE ICC*, Dallas, TX, Jun. 1996, pp. 1558–1562.
[26] L. Lewis, "A case based reasoning approach to the management of faults in communication networks," in *Proc. IEEE INFOCOM*, vol. 3, San Francisco, CA, Mar. 1993, pp. 1422–1429.
[27] A. S. Franceschi, L. F. Kormann, and C. B. Westphall, "Performance evaluation for proactive network management," in *Proc. IEEE ICC*, Dallas, TX, June 1996, pp. 22–26.
[28] L. Lewis and G. Dreo, "Extending trouble ticket systems to fault diagnosis," *IEEE Network*, vol. 7, pp. 44–51, Nov. 1993.
[29] I. Katzela and M. Schwarz, "Schemes for fault identification in communication networks," *IEEE/ACM Trans. Networking*, vol. 3, pp. 753–764, Dec. 1995.
[30] I. Rouvellou and G. Hart, "Automatic alarm correlation for fault identification," in *Proc. IEEE INFOCOM*, Boston, MA, Apr. 1995, pp. 553–561.
[31] A. Bouloutas, G. Hart, and M. Schwartz, "On the design of observers for failure detection of discrete event systems," in *Network Management and Control*. New York: Plenum, 1990.
[32] A. Lazar, W. Wang, and R. Deng, "Models and algorithms for network fault detection and identification: A review," in *Proc. IEEE Int. Contr. Conf.*, 1992.
[33] G. Jakobson and M. D. Weissman, "Alarm correlation," *IEEE Network*, vol. 7, pp. 52–59, Nov. 1993.
[34] F. Feather and R. Maxion, "Fault detection in an ethernet network using anomaly signature matching," in *Proc. ACM SIGCOMM*, vol. 23, San Francisco, CA, Sept. 1993, pp. 279–288.
[35] S. Papavassiliou, M. Pace, A. Zawadzki, and L. Ho, "Implementing enhanced network maintenance for transaction access services: Tools and applications," *Proc. IEEE Int. Contr. Conf.*, vol. 1, pp. 211–215, 2000.
[36] H. L. V. Trees, *Detection, Estimation, and Modulation Theory*. New York: Wiley, 1971, vol. 1.
[37] C. Cohen-Tannoudji, B. Diu, and F. Laloe, *Quantum Mechanics*, 2nd ed. New York: Wiley, 1977, vol. 1.
[38] M. Kirby and L. Sirovich, "Application of the Karhunen–Loeve procedure for the characterization of human faces," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 103–108, Jan. 1990.
[39] M. Thottan, "Network fault classification using abnormality vectors," in *Proc. IFIP/IEEE Conf. Integrated Network Manage.*, Seattle, WA, 2001.
[40] M. Thottan and C. Ji, "Proactive anomaly detection using distributed intelligent agents," *IEEE Network*, vol. 12, pp. 21–27, Sept./Oct. 1998.
[41] C. Hood and C. Ji, "Proactive network fault detection," in *Proc. IEEE INFOCOM*, vol. 3, Kobe, Japan, Apr. 1997, pp. 1147–1155. [Online]. Available: http://neuron.ecse.rpi.edu/.
[42] C. Hood, "Intelligent detection for fault management for communication networks," Ph.D. dissertation, Rensselaer Polytech. Inst., Troy, NY, 1996.
[43] M. Mandjes, I. Saniee, S. Stolyar, and R. Schmidt, "Load characterization, overload prediction and load anomaly detection for voice over ip traffic," in *Proc. 38th Annu. Allerton Conf. Commun., Contr. Comput.*, Nov. 2000.
[44] L. Li, M. Thottan, B. Yao, and S. Paul, "Distributed network monitoring with bounded link utilizationin ip networks," in Proc. IEEE Infocom., 2003, to be published.
[45] L. Zhang, Z. Liu, and C. Xia, "Clock synchronization algorithms for network measurements," in *Proc. IEEE INFOCOM*, 2002.
[46] M. Wei, "Computer communication network traffic: Modeling, analysis and tests," M.S. Thesis, Rensselaer Polytechnic Inst., Troy, NY, 1996.

Marina Thottan received the Ph.D. degree in electrical engineering from Rensselaer Polytechnic Institute, Troy, NY, in 2000.

She is currently a Member of Technical Staff with the Networking Software Research Center, Bell Laboratories, Holmdel, NJ. Her research interests are in network measurements, IP QoS, time series analysis, and signal detection theory.

Chuanyi Ji received the B.S. degree (with honors) from Tsinghua University, Beijing, China, in 1983, the M.S. degree from the University of Pennsylvania, Philadelphia, in 1986, and the Ph.D. degree from the California Institute of Technology, Pasadena, in 1992, all in electrical engineering.

She is an Associate Professor a with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta. She was on the faculty at Rensselaer Polytechnic Institute (RPI), Troy, NY, from 1991 to 2001. She spent her sabbatical at Bell Labs—Lucent, Murray Hill, NJ, in 1999 and was a visiting faculty member at the Massachusetts Institute of Technology, Cambridge, in Fall 2000. Her research lies in the areas of network management and control and adaptive learning systems. Her research interests are in understanding and managing complex networks, applications of adaptive learning systems to network management, learning algorithms, statistics, and information theory.

Dr. Ji received the CAREER award from the National Science Foundation in 1995 and an Early Career award from RPI in 2000.