Combining Filtering and Statistical Methods for Anomaly Detection

Augustin Soule
LIP6-UPMC

Kavé Salamatian
LIP6-UPMC

Nina Taft
Intel Research

Abstract

In this work we develop an approach for anomaly detection for large scale networks such as that of an enterprize or an ISP. The traffic patterns we focus on for analysis are that of a network-wide view of the traffic state, called the traffic matrix. In the first step a Kalman filter is used to filter out the "normal" traffic. This is done by comparing our future predictions of the traffic matrix state to an inference of the actual traffic matrix that is made using more recent measurement data than those used for prediction. In the second step the residual filtered process is then examined for anomalies. We explain here how any anomaly detection method can be viewed as a problem in statistical hypothesis testing. We study and compare four different methods for analyzing residuals, two of which are new. These methods focus on different aspects of the traffic pattern change. One focuses on instantaneous behavior, another focuses on changes in the mean of the residual process, a third on changes in the variance behavior, and a fourth examines variance changes over multiple timescales. We evaluate and compare all of these methods using ROC curves that illustrate the full tradeoff between false positives and false negatives for the complete spectrum of decision thresholds.

1 Introduction

Traffic anomalies such as attacks, flash crowds, large file transfers and outages occur fairly frequently in the Internet today. Large enterprise networks often have a security operations center where operators continuously monitor the network traffic hoping to detect, identify and treat anomalies. In smaller networks, these tasks are carried out by general network administrators who are also carry out other day-to-day network maintenance and planning activities. Despite the recent growth in monitoring technology and in intrusion detection systems, correctly detecting anomalies in a timely fashion remains a challenging task.

One of the reasons for this is that many of today's security solutions yield equipment that collects and analyzes traffic from one link at a time. Similarly many research efforts consider anomaly detection on a per link basis [2, 8, 3]. To detect traffic anomalies one typically seeks to characterize, or build a model, of what constitutes normal behavior. After filtering out normal looking traffic, anomaly detection methods analyze the residual traffic pattern for deviations. Considering only one link is limiting. Since any flow will traverse multiple links along its path, it is intuitive that a flow carrying an anomaly will appear in multiple links, thus increasing the evidence to detect it. Instead in this paper, we focus on using data from all the links in an enterprise or ISP network simultaneously. Since any anomaly has to traverse multiple links on route to its destination, an anomaly has the potential to be visible in any of the links its traverses. Since we cannot know in advance where anomalies will originate, nor the path they will take, it is advantageous to consider the behavior of all the links in an enterprise simultaneously when developing both a model of "normal" traffic and a method for analyzing the "residuals".

A traffic matrix is a representation of the network-wide traffic demands. Each traffic matrix entry describes the average volume of traffic, in a given time interval, that originates at a given source node and is headed towards a particular destination node. In an enterprise network these nodes may be computers, whereas in an ISP network the end nodes can be routers. In this paper we propose to use predictions of traffic matrix behavior for the purposes of anomaly detection.

Since a traffic matrix is a representation of traffic volume, the types of anomalies we might be able to detect via analysis of the traffic matrix are volume anomalies [12]. Examples of events that create volume anomalies are denial-of-service attacks (DOS), flash crowds and alpha events (e.g., non-malicious large file transfers), as well as outages (e.g., coming from equipment failures).

Obtaining traffic matrices was originally viewed as a challenging task since it is believed that directly measuring

them is extremely costly as it requirements the deployment of monitoring infrastructure everywhere, the collection of fine granularity data at the flow level, and then the processing of large amounts of data. However in the last few years many inference based techniques have been developed (such as [22, 23, 19, 18, 20, 6] and many others) that can estimate traffic matrices reasonably well given only per-link data such as SNMP data (that is widely available). These techniques focus on estimation and not prediction.

In this paper we build upon one of our previous techniques [20, 18] for traffic matrix estimation by using it to provide predictions of future values of the traffic matrix. A traffic matrix is a dynamic entity that continually evolves over time, thus estimates of a traffic matrix are usually provided for each time interval (e.g., most previous techniques focus on 5 or 10 minute intervals). We predict the traffic matrix one step (e.g., 5 minutes) into the future. One of the key ideas behind our approach lies in the following observation. Five minutes after the prediction is made, we obtain new link-level SNMP measurements, and then estimate what the actual traffic matrix should be. We then examine the difference between our prediction (made without the most recent link-level measurements) and the estimation (made using the most recent measurements). If our estimates and predictor are usually good, then this difference should be close to zero. When the difference is sizeable we become suspicious and analyze this residual further to determine whether or not an anomaly alert should be generated.

We compare four different methods for signalling alerts when analyzing residual traffic. The simplest method compares the instantaneous residual traffic to a threshold. The second method considered is a small variation on the deviation score idea presented in [2]. Their key idea is to compare a local (temporally) variance calculation with a global variance assessment. The deviation score used in [2] is computed using output signals of a wavelet transform applied to IP flow level data from a single link. We apply this idea of comparing the local to the global variance on our filtered residual signal. In our third scheme, we apply wavelet analysis only on the filtered traffic (in [2] wavelet analysis is applied directly on the original signal). We signal an alert when the detail signal (now a type of residual) at each of a few different timescales exceeds a threshold. We raise an alarm only if the threshold is exceeded at multiple timescales. The fourth method uses a generalized likelihood ratio test to identify the moment an anomaly starts, by identifying a change in mean rate of the residual signal. These last two methods, introduced here for the first time, are particular applications of known statistical techniques to the anomaly detection domain.

Our approach is different from other approaches in that usually anomaly detection is performed directly on monitored data that is captured at the target granularity level. Instead we perform anomaly detection on origin-destination (OD) flows, a granularity of data that we infer from other measurements (link statistics). Our study shows that it is possible to follow such an approach towards a positive outcome.

To validate our methods we use both real data from the Abilene network and a synthetic anomaly generator that we developed. These two approaches are complementary as their advantages and disadvantages are opposite. The advantage of evaluating using real world traces is that we test our methods on actual anomalies that have occurred in the Internet. The disadvantage of using only collected traces is that the statistical parameters of the anomaly cannot be varied. One cannot therefore identify the limits of a method. For example, one cannot ask "would we still detect the anomaly if its volume were lower?". Using synthetically generated anomalies in which we carefully control the anomaly parameters, we can stress test and identify the limits of an algorithm. However the synthetic anomalies are limited because we have no evidence of them in the Internet. Our approach to validation thus employs both of these approaches in order to extract the benefits of each.

We use ROC (Receiver Operating Characteristic) curves as our key evaluation criteria. ROC curves have received wide usage in medical diagnosis and signal detection theory, but relatively little in network security. A ROC curve is a graphical representation of the tradeoff between the false positive and false negative rates for every possible decision threshold. We include a brief description of the meaning and theory behind ROC curves to illustrate a general methodology for analysing network security solutions. We use these to compare our four solutions. The advantage of this approach is that it permits scheme comparison throughout the entire range of decision thresholds. This eliminates the difficulty that arises when one tries to compare methods each of which uses a particular and seemingly ad hoc threshold choice. In addition, we also present the performance of these methods in terms of their detection time. This is important as most anomaly detection methods incur some lag time before reaching a decision. Finally we assess the false positive and false negative rates our schemes yield as the volume of an anomaly is varied from low-volume anomalies to high-volume ones.

The most important, and only, work to date that uses a network-wide perspective for volume anomaly detection was that of [12]. In this work, the authors used the ensemble of all the links in a network and performed Principle Components Analysis to reduce the dimensionality of the data. They illustrate that by projecting onto a small number of principal components one could filter out the "normal" traffic. The traffic projected onto the remaining components is analyzed for anomalies using a G-statistic test on the predictive error. While our paper essentially tackles the same problem, our work differs in numerous

ways: i) we process the incoming link data using kalman filters rather than PCA analysis and generate traffic matrix predictions; ii) the granularity we focus on is that of OD flows whereas they use link data when analyzing residuals. (Note, they use the OD flows as a secondary step, after detecting an anomaly, in order to identify the source); iii) they consider a single test on the residual traffic whereas we propose two new ones and conduct a comparative evaluation of four schemes; iv) our method for validation differs since we supplement the Abilene data with synthetic anomaly testing; v) our evaluation is different because we make use of ROC curves for evaluation, examine detection lag times as well as sensitivity to anomaly volume sizes.

Section 2 describes how we model the OD flows and our solution for traffic matrix prediction. The methods for analyzing filtered traffic and determining how to detect an anomaly are presented in Section 3. We discuss our approach to validation and fully describe our synthetic anomaly generator in Section 4. All of our evaluations and the results are shown in Section 5.

2 Modeling Normal Traffic

We assume that the monitoring infrastructure in our network can easily obtain per-link statistics on byte counts (as in SNMP today). From this we want to infer the traffic matrix that includes all pairs of origin-destination (OD) flows. This is the classic traffic matrix estimation problem. If we design a realistic model for the evolution of the network's traffic matrix, then we can use this to filter our usual behavior. For the sake of completeness we now summarize our linear dynamic state space model for the OD flows and our Kalman filter method for estimating the traffic matrix. This was originally presented in [18]. We expand on our previous work by illustrating how this can be used to make future predictions of the traffic matrix and describe the resulting residual processes that can be obtained when filtering via this approach.

Since the OD flows are not directly observable (measurable with today's technology) from the network, we refer to them as *hidden network states* or simply as *network states*. The link load levels (e.g., total bytes per unit time) are directly observable in networks, and are captured via the SNMP protocol that is widely deployed in most commercial networks today. Because the total traffic on a link is the sum of all the OD flows traversing that link, the relationship between SNMP data and OD flows can be expressed by the linear equation $Y_t = A_t X_t + V_t$, where Y_t represents the vector of link counts vector at time t, and X_t is the OD flows organized as a vector (hidden network states). A_t denotes the routing matrix whose elements $a_t(i,j)$ are 1 if OD flow j traverses link i, and zero otherwise. (In some networks fractional routing is supported.) The term V_t captures the stochastic measurement errors associated with the data collection step. All these parameters are defined for a general discrete time t.

To capture the dynamic evolution of OD flows we need a model that specifies X_{t+1} as a function of X_t. We seek a model that can be used for prediction of the OD flows one step into the future. Providing an efficient model that captures traffic dynamics is not so simple. It has been observed that traffic entering the network is characterized by highly variable behavior in time [13]. There are many sources of this variability, including daily periodic behavior, random fluctuations with relatively small amplitude, and occasional bursts. Sudden changes in the traffic are not uncommon and can be related to different benign causes such as the addition of new customers, network equipment failures, flash crowds or to malicious activities such as attacks conducted against the network. Ignoring the attacks for the moment, our model for OD flows must be rich enough to incorporate these sources of variability for normal traffic. It is also known that both temporal correlations within a single OD flow exist, and that spatial correlations across some OD flows occurs [18].

We adopt a linear state space model to capture the evolution of OD flows in time. This predictive model relates the network state X_{t+1} to X_t as follows: $X_{t+1} = C_t X_t + W_t$, where the state transition matrix C_t captures temporal and spatial correlations in the system, and W_t is a noise process that accounts for both the randomness in the fluctuation of a flow, and the imperfection of the prediction model. Linear stochastic predictive models, combined with Gaussian noise, have been successfully applied to a large spectrum of monitoring problems.

The matrix C_t is an important element of the system. A diagonal structure for C_t indicates that only temporal correlations are included in the model of an OD flow. When C_t has off-diagonal elements that are non-zero, then spatial correlation across OD flows have been incorporated into the model. For traffic matrix estimation, using a non-diagonal matrix for C_t is preferable so that one can benefit from incorporating spatial correlation (as used in [20]). When traffic matrix estimation is carried out, the main task is that of taking a total byte count for each link and partitioning it among the the multiple OD flows traversing that link. When an anomaly occurs on a link, it is possible for an anomaly (originating within one OD flow) to get spread across all the OD flows on that link during the estimation procedure. To avoid this phenomenon, that would make it more difficult to detect anomalies in OD flows, we use a diagonal structure for C_t (unlike the model used in [18]).

Putting the above elements together, our complete model is that of a linear state space dynamical system, that relates the observables (Y_t) to the unobservables (X_t), and is given by,

$$\begin{cases} X_{t+1} & = C_t X_t + W_t \\ Y_t & = A_t X_t + V_t \end{cases} \quad (1)$$

We assume both the state-noise W_t and the measurement-noise V_t to be uncorrelated, zero-mean Gaussian white-noise processes and with covariance matrices Q_t and R_t:

$$E[W_k W_l^T] = \begin{cases} Q_k, \text{ if } k = l \\ 0, \text{otherwise} \end{cases}$$

$$E[V_k V_l^T] = \begin{cases} R_k, \text{ if } k = l \\ 0, \text{otherwise} \end{cases}$$

$$E[W_k V_l^T] = 0 \quad \forall k, l \tag{2}$$

These assumptions might appear restrictive however a large body of research in the control theory literature has been devoted to Kalman filtering robustness. The lessons learned from this literature are that because of the feedback mechanism, and ongoing readjustment of estimated values, Kalman Filters are robust to model imprecision as well as to some deviation from gaussianity in the noise. The rule of thumb for reaching a certain level of robustness is to use noise with slightly larger variance for W_t than obtained by direct evaluation of noise.

Given the above assumptions and a set of observations $\{Y_1, ..., Y_{t+1}\}$, the task is to determine the estimation filter that at the $(t+1)$-*st* instance in time generates an optimal estimate of the state X_{t+1}, which we denote by $\hat{X}_{t+1}$. Optimality is defined in the sense of Minimum Variance Error Estimator that is defined as follows:

$$E[||X_{t+1} - \hat{X}_{t+1}||^2] = E[(X_{t+1} - \hat{X}_{t+1})^T (X_{t+1} - \hat{X}_{t+1})] \tag{3}$$

The classical tool for dealing with this type of problem is the well known Kalman Filter [10]. It addresses the general problem of trying to estimate a discrete state vector when the observations are only a linear combination of this underlying state vector. The Kalman filter estimates the system state process by using a two step approach, that iterates for each time t. We use $\hat{X}_{t|i}$ we refer to the estimation of X_t based on time $i, t \geq i$. (We introduce here the more general case of time-varying systems, where all the parameters are indexed by time.)

• **Prediction Step**: Let $\hat{X}_{t|t}$ denote the estimate of the state at time t given all the observations up to time t (i.e. Y^t). This term has a variance that is denoted by $P_{t|t}$. Let $\hat{X}_{t+1|t}$ denote the one step predictor. This prediction is made using all the observed data up to time t. Since the model $X_{t+1} = C_t X_t + W_t$ includes the noise term W_t (with covariance Q_t), this prediction will have some associated variability, that is denote as $P_{t+1|t}$. In the prediction step, we are given $\hat{X}_{t|t}$ and $P_{t|t}$, and compute both our prediction, and the variance of this prediction, as follows.

$$\begin{cases} \hat{X}_{t+1|t} & = C_t \hat{X}_{t|t} \\ P_{t+1|t} & = C_t P_{t|t} C_t^T + Q_t \end{cases} \tag{4}$$

• **Estimation Step**: In this step, the kalman filter updates the state estimate $X_{t+1|t+1}$, and its variance ($P_{t+1|t+1}$) by using a combination of their predicted values and the new observation Y_{t+1}. The new estimate at time $t+1$ is given by,

$$\begin{cases} \hat{X}_{t+1|t+1} = & \hat{X}_{t+1|t} + K_{t+1}[Y_{t+1} - A_{t+1}\hat{X}_{t+1|t}] \\ P_{t+1|t+1} = & (I - K_{t+1}A_{t+1})P_{t+1|t}(I - K_{t+1}A_{t+1})^T \\ & + K_{t+1}R_{t+1}K_{t+1}^T \end{cases} \tag{5}$$

The new estimate at time $t+1$ for $\hat{X}_{t+1|t+1}$ is computed using the prediction from the previous time instant $\hat{X}_{t+1|t}$ that is adjusted by a correction factor. Consider the latter part of this equation. By multiplying our prediction $\hat{X}_{t+1|t}$ by A_t, we generate a prediction for the link counts $\hat{Y}_{t+1}$. Hence the term in brackets $[Y_{t+1} - A_{t+1}\hat{X}_{t+1|t}] = Y_{t+1} - \hat{Y}_{t+1}$ is the error in our prediction of the link counts. This term is multiplied by the matrix K_{t+1} that is called Kalman gain matrix. It is obtained by minimizing the conditional mean-squared estimation error $E[\tilde{X}_{t+1|t+1}^T \tilde{X}_{t+1|t+1} | Y^t]$ where the estimation error is given by $\tilde{X}_{t|t} = \hat{X}_{t|t} - X_t$. By applying some basic linear algebra, we can write it as:

$$K_{t+1} = P_{t+1|t}\, A_{t+1}^T [A_t P_{t+1|t} A_{t+1}^T + R_{t+1}]^{-1} \tag{6}$$

Hence this second step takes the new observation of Y when it becomes available, and corrects its previous prediction. The above equations together with the initial conditions of the state of the system $\hat{X}_{0|0} = E[X_0]$ and the associated error covariance matrix $P_{0|0} = E[(\hat{X}_{0|0} - X_0)(\hat{X}_{0|0} - X_0)^T]$ define the discrete-time sequential recursive algorithm, for determining the linear minimum variance estimate, known as Kalman Filter.

In our previous paper [18], the traffic matrix is populated (i.e. estimated) using $\hat{X}_{k+1|k+1}$. Nevertheless, it is clear that the Kalman filter gives more information than only estimates. Using the predictive ability of the filter it is possible to estimate the future evolution of the traffic matrix. The correction step in Equation (5) essentially captures the part of the process that our model could not predict. It is this unpredictable part that we want to track for anomaly detection. Based on the study in [18], we know that the Kalman filter method for estimating the traffic matrix works well. Hence most of the time, the correction factors are negligible. Now if at some time instant we see a large correction of our prediction , we could flag this as anomalous and generate an alert.

We are thus motivated to examine the errors that our one-step predictor generates. The errors in our prediction of the link values are denoted by,

$$\epsilon_{t+1} = Y_{t+1} - A_{t+1}\hat{X}_{t+1|t},$$

In Kalman filtering terminology this error is typically the innovation process. It is the difference between the observed (measured) value Y_{t+1} and its prediction $A_{t+1}\hat{X}_{t+1|t}$. The innovation process ϵ_t is considered to be white gaussian noise with a covariance matrix given by :

$$E\left[\epsilon_{t+1}\epsilon_{t+1}^T\right] = A_{t+1}P_{t+1|t}A_{t+1}^T + R_{t+1}. \quad (7)$$

Since in our case, we are interested in anomalies in the OD flows, we can define, by extension, the **residual** η_{t+1},

$$\eta_{t+1} = \hat{X}_{t+1|t+1} - \hat{X}_{t+1|t} = K_{t+1}\epsilon_{t+1},$$

that is the difference between the new estimate of the state ($\hat{X}_{t+1|t+1}$), corrected using the most recent measurement at time $(t+1)$, and its prediction $\hat{X}_{t+1|t}$ made based only on information available up to time t. It is also a measure of the new information provided by adding another measurement in the estimation process. Using Equation (5), we can see that the error in the OD flow estimate is related to the error in the link estimate via $\eta_{t+1} = K_{t+1}\epsilon_{t+1}$.

This is also a zero-mean gaussian process, whose variance S_{t+1} can be easily derived as

$$S_{t+1} = E[\eta_{t+1}\,\eta_{t+1}^T] = K_{t+1}(A_{t+1}P_{t+1|t}A_{t+1}^T + R_{t+1})K_{t+1}^T \quad (8)$$

The residual process can be shown to be asymptotically uncorrelated, *i.e.* $E\left[\eta_t\,\eta_l^T\right] = 0,\ t \neq l$. This can be understood by observing that asymptotically the gain matrix of Kalman filter K_{t+1} converge to a fixed point $\bar{K}$. The residual is an important measure of how well an estimator is performing. A non-zero residual could mean that an anomaly has occurred, and in the next section 3, we present a few schemes for further examining this residual time series to detect anomalies.

In this section, we presented the Kalman filtering method in its general settings under non-stationary assumptions. In the following sections, we will assume a stationary situation where the matrices A, C, Q and R are constant in time, making it possible to drop their subscripts. However, the rest of the methodology presented in this paper can easily be generalized to incorporate time dependency.

There is an issue of calibration for using such a Kalman filter model because the matrices C, Q and R need to be calibrated. We developed and presented in [20] an Expectation Maximization based approach for calibrating these matrices. In [20] we showed that for reliable OD flow estimation we need to recalibrate the Kalman filter every few days when the underlying model changes. When there are anomalies, this might suggest that the model should be recalibrated every time an anomaly occurs. However, one interesting result of this current paper is that this recalibration step is often not needed if the goal is just anomaly detection. For example, in applying our anomaly detection schemes on the Abilene data, we found that no recalibration was needed for 7 days (covering 74 anomalies). Hence the requirements for recalibration appear to be stronger for traffic matrix estimation than for anomaly detection.

3 Analyzing Residuals

Before explaining our four methods for examining residuals to look for anomalies, we discuss some important issues regarding sources of errors, understanding the meaning of decision thresholds, and how they are selected. In doing so, we explain our methodology for comparing different anomaly detection schemes.

There are two sources of errors that can appear in the residual process. One is from errors in the underlying traffic model, while the second will come from anomalies in the traffic. Suppose, for a moment, that we consider any general random process Z_t that we want to check for anomalies. Let $\hat{Z}_t$ denote our prediction for this process based upon a model. Since our model may not be exact, we let ζ_t denote the expected prediction error, a zero-mean random variable with known covariance. If we define ξ_t as the anomaly term at time t, we can write :

$$Z_t = \hat{Z}_t + \zeta_t + \xi_t.$$

In this equation ξ_t is a random variable accounting for the unexpected change caused by the anomalies, *i.e.* $\xi_t = 0$ if there are no anomalies and $\xi_t \neq 0$ when there is an anomaly.

There is an important decision to be made as to which data granularity to examine in order to try to observe anomalies. We can consider either looking at the prediction errors observed on the link data Y_t or the estimation errors on the OD flows X_t. Our experience showed us that detection schemes work better when operating at the granularity level of the OD flow rather than at that of the link. Although we cannot observe the OD flow directly, we can observe the error in our prediction of the OD flow and that turns out to be plenty sufficient for our purposes. We point out that the four schemes we discuss for examining errors can be applied to either type of error. These methodologies require only that we understand the covariance process of the associated ζ_t process.

To detect anomalies on the SNMP link counts, one should use the statistics of the innovation process in place of the statistics of ζ. This is readily available in our model since it is equivalent to the statistics of the innovation process in the Kalman Filter. The innovation obtained as the output of the Kalman filter is exactly the prediction error $\zeta_t + \xi_t$.

Anomaly detection on OD flows is more tricky as the prediction error is not directly observable (as OD flows are hidden). However, the good news is that the covariance of ζ is known and equal to $P_{t+1|t+1}$. Moreover, the residual $\eta_{t+1} = \hat{X}_{t+1|t+1} - \hat{X}_{t+1|t} = K_{t+1}\epsilon_{t+1}$ can be observed

and its covariance can be derived as S_{t+1}. And last but not least the estimation error ζ and the residual η are correlated gaussian processes, *i.e* one might use one for estimating the other and the least squared error estimator is :

$$\zeta_t + \xi_t \approx -K_t A_t P_{t|t-1} S_t^{-1} \eta_t \tag{9}$$

The approximation comes from the fact that this is just an estimation of an unobserved value (the OD flows estimation error) based on an observed value (residual).

3.1 Anomaly detection as a statistical test

We now wish to illustrate how any anomaly detection scheme can be viewed as a statistical hypothesis test. To do this, we first explain how such tests are evaluated. The tested are evaluated by exploring the fundamental tradeoff between the false positive and false negative rates. Hypothesis testing explains how to pick decision thresholds when faced with balancing this particular tradeoff.

All four of the schemes we use to evaluate the residuals rely on the selection of a threshold that is used to decide whether or not an alarm is raised. In fact, any anomaly or change detection method will require that a threshold be selected. In our evaluation of these methods we consider all possible thresholds for each method. We do this by assessing the performance of our method using Receiver Operation Characteristic (ROC) curves.

ROC curves have been developed in the context of signal detection [5], and have been widely used for medical analysis purposes [24]. ROC curves are useful because the describe the full tradeoff between false positives and false negatives over the complete spectrum of operating conditions (i.e., decision threshold settings). In an ROC curve, we plot the false positive rate on the x-axis and one minus the false negative rate on the y-axis. The y-axis thus represents the true positives (the anomalies we want to catch). An algorithm is considered very good if its ROC curve climbs rapidly towards the upper left corner of the graph. This means that we detect a very high fraction of the true anomalies with only a few false positives. Sample ROC curves can be seen in Figure 1 (to be fully explained later).

To quantify how quickly the ROC curve rises to the upper left hand corner, one simply measures the area under the curve. The larger the area, the better the algorithm. ROC curves are essentially parametric plots as each point on the curve corresponds to a different threshold. Each point illustrates a particular tradeoff between false positives and false negatives. Each algorithm results in one curve, and by comparing these curves we can compare algorithms. The curve with the largest area underneath it corresponds to the better algorithm. Since each curve represents the entire range of thresholds, we can compare algorithms throughout their entire region.

ROC curves are grounded in statistical hypothesis testing. As mentioned earlier, any anomaly detection method will at some point use a statistical test to verify whether or not a hypothesis (e.g., there was an anomaly) is true or false. Recall that ξ_t is our residual process and should be zero (or roughly zero) when there is no anomaly. We can form the hypothesis $\mathcal{H}_0 : \xi_t = 0$ for the case when there is no anomaly. We can form an alternate hypothesis $\mathcal{H}_1 : \xi_t ! = 0$ for the case when there an anomaly occurs. This last hypothesis is difficult to handle mathematically, so for the sake of simplicity of exposition, we rewrite the alternate hypothesis as $\mathcal{H}_1 : \xi_t = \mu$. (Conceptually we can continue to think of this as the case when an anomaly occurs). The random variable ξ_t in each hypothesis is assumed to have some distribution. Upon observing a sample of this random variable we compare it to a threshold to decide if we reject $\mathcal{H}_0$ (thereby accepting $\mathcal{H}_1$) or vice versa.

Let FPR denote the false positive rate, the probability that we detect an anomaly given there was no anomaly. Put otherwise, this is the likelihood that we reject $\mathcal{H}_0$ when it was true. The false negative rate, FNR, is the probability that we detect nothing when an anomaly occurs (or the likelihood that we accept $\mathcal{H}_0$ when we should have rejected it). In order to decide whether or not to accept $\mathcal{H}_0$, we compare our observation of ξ_t to a threshold. The Neyman-Pearson criteria says that we should construct this decision threshold to maximize the probability of detection (true positives) while not allowing the probability of false alarm to exceed some value α.

The optimization problem to solve is to find the maximum probability of detection (1-FNR) such that $FPR \leq \alpha$. The likelihood ratio is defined as the ratio of FPR/FNR. The Neyman-Pearson lemma says that the optimal decision threshold is one that satisfies the likelihood ratio test.

$$\frac{FPR}{FNR} \leq T(\alpha)$$

In solving for $T(\alpha)$ (i.e., deriving the curve), each point of this curve corresponds to one value of the decision threshold. In practice, this curve is plotted as the correct detection rate,*i.e.* $1 - FNR$ as a function of false positive rate FPR thus yielding the ROC curve.

For a fixed $FPR = \alpha$, all values $1 - FNR \leq \beta^*$ are achievable by a non-optimal anomaly detector, or equivalently all points below the optimal ROC curve can be achieved. The ROC curve can be derived analytically typically only under simple assumptions (such as ξ_t is Gaussian). In this case the derived curve is an optimal curve. The optimal curve is not a perfect solution (i.e., 100% true positive detection and 0% false positives) because usually there is some inherent noise in the process this limits the best decision one can make.

As a simple example, consider the case when ξ_t is a gaussian random variable with a cumulative distribution

given by Φ. The ROC curve for the hypothesis $\mathcal{H}_0 : \xi_t = 0$ *vs.* $\mathcal{H}_1 : \xi_t = \mu$ is given by :

$$1 - FNR = 1 - \Phi(\Phi^{-1}(FPR) - \mu)$$

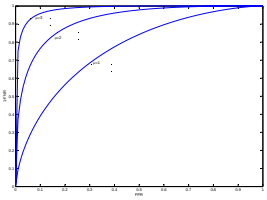


Figure 1: Optimal ROC curve for a gaussian hypothesis testing between $\mathcal{H}_0 : \zeta = 0$ *vs.* $\mathcal{H}_1 : \zeta = \mu$.

Fig. 1 shows the ROC curve for this case for three different alternate hypotheses $\mathcal{H}_1$. In practice, the optimal ROC curves cannot be derived, which limits our ability to see how far a particular detection scheme is from optimal, where optimal is determined based on the underlying noise of the system. However since each scheme yields a different ROC curve, these remain a powerful means of comparison across schemes. If one curve has less area beneath it, then it is clearly inferior, regardless of the threshold level selected.

3.2 Basic analysis using Variance

The first anomaly detector that will be described is also the simplest one. As seen previously in normal operational condition one might assume that $\xi_t = 0$ and that the prediction error ζ_t follows a process with mean 0 and known variance. Under the situation that the statistics of ζ_t, the prediction error are fully known, it is easy to construct a statistical test following the Neyman-Pearson theorem. For this purpose we might use the construction given in Eq. 9.

The approach consists of constructing the process $\tau_t = -K_t A_t P_{t|t-1} S_t^{-1} \eta_t$ and rising an alarm for an OD pair i whenever $\tau_{ti} > T \times \sqrt{(P_{t+1|t+1})_{ii}}$ where T is the threshold. Actually, this approach verifies if the prediction error is inside a confidence interval. This anomaly detector is the optimal one for the case where $\zeta_t + \eta_t$ follow a gaussian distribution. However, if this hypothesis in not precisely true (as frequently in practice), application of this anomaly detector will lead to a ROC curve that is lower than the optimal one.

An interesting property of this method is that the test is verified as soon as a new observation has been processed by the Kalman filter and it can therefore trigger an anomaly very fast. However the drawback of the approach is that each test is being done independently of past observations. This might lead to high false positive rate when the process ζ_t has a heavier tail than the gaussian. One might want to have a less sensitive approach that will not raise an alarm based on only one observation diverging from the bound.

3.3 CUSUM and Generalized Likelihood Ratio test

The previous method missed an essential fact, since we are in the context of random processes, tests executed at each time t are not independent. The classical approach for detecting a change in a random process is the CUSUM (Cumulative Summation) method and its variants [4]. The main intuition behind the CUSUM method is that when a change occurs the log-likelihood ratio of an observation y_i, defined as $s_i = \log \frac{L_1(y)}{L_0(y)}$, shifts from a negative value to a positive one (as after the change hypothesis $\mathcal{H}_1$ becomes more likely). This means that the log-likelihood of observing a sequence of N observations $\{y_0^{N-1}\}$, defined as $S_{N-1} = \sum_{i=0}^{N-1} s_i$, that was decreasing with N, begins to increase after the change. The minimum value of S_j gives an estimate of the change point. Therefore a simple statistical test for change detection consists of testing whether :

$$S_k - \min_{0 \leq j \leq k} S_j > T,$$

where S_k is the log-likelihood ratio defined previously and T is a threshold. After a change has been detected, the time of change can be estimated as :

$$\hat{t}_c = \arg \min_{0 \leq j \leq k} \{S_j\}$$

The previously described CUSUM algorithm has been widely used for anomaly detection. However it suffers from a key drawback. It is stated in the context of a simple hypothesis, where the alternative hypothesis $\mathcal{H}_1$ should be completely defined, *i.e.* the level of the change or in other terms the intensity of the anomaly should be known *a priori*. However in practical settings, this is exactly unknown as by definition anomalies are not predictable.

A solution for this issue is provided by the *General Likelihood Ratio Test*. In this approach the level of change in the CUSUM algorithm is replaced by its maximum likelihood estimate. To describe the approach let's fix a scenario. Suppose an anomaly occurs and this results in a shift in the mean of the residual process. After the shift, the estimation error will no longer be a zero mean random variable of variance σ (σ is assumed to be known), but instead is translated to a mean μ, that is unknown, and the same variance. The GLR algorithm uses a window of estimation error $\{\tau_j^{j+N-1}\}$ and applies for each $i, j \leq i \leq j+N-1$ the following test. It first estimates the mean of the estimation error over the window $\{i, \dots j+N-1\}$ as

$$\hat{\mu} = \frac{1}{j+N-1-i} \sum_{l=i}^{j+N-1} \tau_l$$

It then performs a simple CUSUM test with $\hat{\mu}$ as the level change value and we raise an alarm if a change is detected. We implemented here a variant of the classical GLR method described in [7]. This method is very powerful since there exists a proof that this is the best estimator when level change μ and variance σ are unknown. However its main drawback is that it adds some delay for the detection of the anomaly since it needs some observations after the anomaly to estimate the deviation level. The detection delay will not be constant and will depend on the anomaly. For example, the effect of small volume anomalies on the mean will propagate slowly and thus may not be detected as quickly as large volume anomalies.

3.4 Multiscale analysis using variance

Multi scale analysis has been proposed as a promising approach to make robust anomaly detectors and is now commonly accepted as a powerful tool. The rational behind using multiscale analysis is that anomalies should appear at different time scales and by monitoring these multiple scales one should be able to reduce the False Positive Rate, because a change appearing on only one time scale will not trigger an alarm.

We implemented a multi-scale analysis based on a cascade decomposition of the original signal τ_t into a low frequency approximation a_t^L and a cascade of details d_t^i. The multi-scale decomposition lead to the following relation :

$$\tau_t = a_t^L + \sum_{i=1}^{L} d_t^i.$$

where :

$$\begin{aligned} d_t^i &= \sum_s \tau_s 2^{-i} \psi(2^{-i}s - t),\ i = 1, \ldots, L, \\ a_t^L &= \sum_s \tau_s 2^{-L} \phi(2^{-L}s - t), \end{aligned}$$

and $\psi(.)$ is a mother wavelet function and $\phi(.)$ its corresponding scaling functions [14].

Now, an anomaly detection mechanism, similar to that described in the basic analysis using variance subsection, is applied to each details time series. For each level $l \in [1, L]$we create a 0-1 sequence: each time instant t is assigned either a 0 or 1 where 0 indicates that no anomaly was detected and 1 means an anomaly was flagged. By summing across these 0-1 time series, for a given time instant, we have the number of times that an anomaly was detected across all the details signals. The larger this numer, the more time scales at which the anomaly was detected. (In practice, we sum not over a single time instant, but over a small window in each signal). An anomaly flag is raised if the anomaly is detected at a sufficient number of scales. The computation of the wavelet introduces a lag in the detection; this lag will be a function of of the largest scale used.

3.5 Multi scale variance shift

This method is derived from [2]. In this paper the authors detect the difference between the local and the global variance of the process. They first remove the trend of the signal using a wavelet transform, *i.e.* the remove the approximation part of a wavelet transform. Thereafter they use a small window to compute a local variance. Whenever the ratio between this local variance and the global variance (computed on all the data) exceeds a threshold T then an alarm is triggered.

This method is in fact a special case of the multiscale analysis previously described, where only two scales are analyzed, the scale at which the global variance is calculated and the local scale where the local variance is calculated. The approach can be assimilated to wavelet transform with a Haar wavelet. The other interesting point of the approach is that it detects a variation in the variance of the process in place of detecting a variation in the mean as previously described approaches. It is noteworthy that other approaches could also be adapted to detecting changes in variance in place of the mean.

This method will also experience a detection lag time, since the wavelet approach introduces a lag due to the time needed to compute the wavelet transform in the two scales. The width of the window of time over which to computes the local variance is very important and will depend on the duration of the anomaly to detect.

4 Validation Methodology

The validation of any anomaly detection method is always fraught with difficulty. The challenge comes from our inability to establish the "ground truth". Among the most interesting performance metrics for such methods are the false positive and false negative rates. However computing these rates requires us to know exactly which events (and corresponding point in time) were anomalies and which were not. One common approach to evaluating anomaly detection algorithms is to collect live data in the form of a packet or flow level trace, and then to have this trace "labeled". Labeling or marking a trace is the procedure by which each anomalous event is identified along with its start and finish time. Perhaps the best way to do this in today's world is for a security operations expert to do the labeling either via visual inspection or with the help of tools. They have a wealth of real world experience that is hard to automate. Although this is currently our best option, the labeling method is not perfect as operators can make mistakes, either missing an anomaly or generating a false

positive. The advantage of using labeled traces is that they capture real world events. The disadvantage is that such traces contain a fixed number of events whose parameters cannot be varied. For example, one cannot ask "suppose the volume of the attack had been a little lower, would our algorithm have caught it?"

A second approach to validation is to synthetically generate attacks. The advantage of this approach is that the parameters of an attack (attack rate, duration, number of flows involved, etc.) can be carefully control. One can then attempt to answer the above question. This enables sensitivity testing of any detection algorithm. Clearly the disadvantage is that these attacks have not happened anywhere and thus may be of less interest.

We believe that a good approach to validation of an anomaly detection algorithm should contain both of the above approaches, so as to obtain the benefits of each method. For our set of real world data with anomalies, we obtained four weeks of traffic matrix data from the Abilene Internet2 backbone network. Abilene is a major academic network, connecting over 200 US universities and peering with research networks in Europe and Asia. This data was labeled using the method in [12]. We developed our own synthetic anomaly generator and implemented it in Matlab. This is described in detail further below.

4.1 Abilene Data

The Abilene backbone has 11 Points of Presence (PoP) and spans the continental US. The data from this network was collected from every PoP at the granularity of IP level flows. The Abilene backbone is composed of Juniper routers whose traffic sampling feature was enabled. Of all the packets entering a router, 1% are sampled at random. Sampled packets are aggregated at the 5-tuple IP-flow level and aggregated into 5 minute bins. This thus dictates the underlying time unit of all of our estimations and detections. The raw IP flow level data is converted into a PoP-to-PoP level matrix using the procedure described in [11]. Since the Abilene backbone has 11 PoPs, this yields a traffic matrix with 121 OD flows. Note that each traffic matrix element corresponds to a single OD flow, however, for each OD flow we have a four week long time series depicting the evolution (in 5 minute increments) of that flow over the measurement period.

4.2 Synthetic Anomaly Generation

Our approach to synthetically generation anomalies makes use of the Abilene traffic matrix. The idea is to select either one, or a set of, OD flows to be involved in the anomaly, and then to add anomalies on top of the baseline traffic level for those OD flows. Our reasons for adding anomalies on top of the existing traffic matrix are as follows. We want to detect anomalies using the ensemble of all network links, we need to populate the load of the entire network. Other methods are available such as [17]. But they generate single link packet traces while we need multi-link SNMP data. Thus we rely on our measured dataset for generating malicious data.

Using the abilene traffic matrix allows us to recreate realistic loads network-wide. This includes all the many sources of variability exhibited on the set of network links.

To inject an anomaly into this network, we use a three step procedure. These procedure is carried out for each OD flow involved in the anomaly.

1. Extract the long-term statistical trend from the selected OD flow. The goal is to capture the diurnal pattern by smoothing the original signal.

2. Add Gaussian noise onto the smoothed signal.

3. Add one of the anomalies as described in Table 1 on top of this resulting signal.

These three steps are depicted pictorially in Figure 2. It was shown in [19] that OD pairs in an ISP exhibit strong diurnal patterns. These 24-hour cycles represent normal types of variability in aggregated traffic. Another normal source of variability in OD flows simply comes from noise [13], and thus the first two steps are intended to represent the level of traffic in an OD flow right before the anomaly starts; this should look like regular non-anomalous traffic.

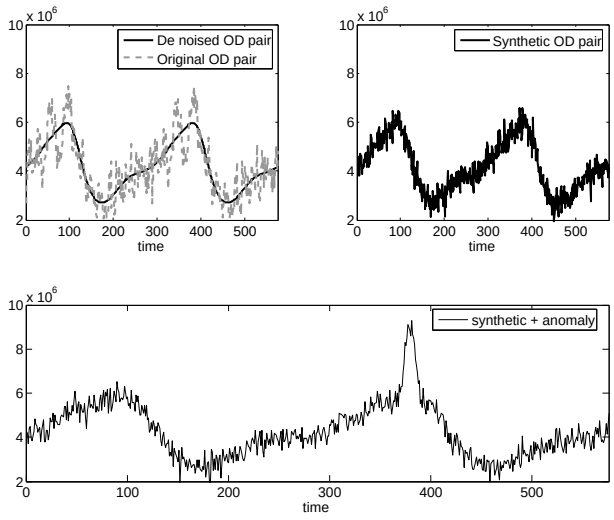


Figure 2: Three steps for synthetic generation of an anomaly.

We extract the diurnal trend using a discrete wavelet transform; wavelet methods here useful since these trends are typically non-stationary. Evidence of the ability

of spectral methods to capture the underlying trends in highly aggregated traffic has been observed in [19, 8, 2]. We compute the first five approximation signals using a Daubechies-5 mother wavelet with 5 levels. We keep the approximation signal at the 5th level, thus filtering out everything except this smoothed signal. This smoothed, or de-noised, signal is shown in the top left plot of Figure 2 as the solid line. We add to this baseline signal a zero mean Gaussian noise who variance is computed as follows. We take the first 5 detailed signals from our wavelet transform, and compute the variance of the sum of the 5 detailed signals. A sample signal produced after step 2 is depicted in the upper right plot of Figure 2. An important reason to use a signal that has been smoothed and only supplemented with Gaussian noise is to ensure that there is no anomaly in this OD flow other than the one we are about to add.

The last step is to add an anomaly onto this baseline traffic. This is depicted in the bottom plot of Figure 2 where we see the anomaly added on top of the filtered OD flow. In our synthetic anomaly generator we characterize each anomaly by four parameters, namely, *volume*, *duration*, *number of OD flows involved*, and a *shape* function. The *shape* function refers to the rate of increase when the anomaly begins (also called *ramp up*), as well as the rate of decrease as the anomaly tapers off. We include four different shape functions: ramp, exponential, square and step. The ramp function is further characterized by a slope parameter, and the exponential shape by its rate parameter. Our intent is to define a feasible range for each of these parameters such that we are able to capture the general behavior of known anomaly types as well as to encompass a broader range of behaviors.

As pointed out in [15], there are unfortunately no comprehensive studies yet that provide detailed statistical descriptions of a broad set of volume anomalies. There are a handful of studies [2, 11, 16, 9, 3] that provide useful pieces of information towards this end. The characterization part of these studies often touch briefly on a wide variety of metrics, from attack rate and duration to others such as the distribution of source or victim IP addresses, type of protocol involved in the attack, and the effect on the end system (e.g., number of sessions open), etc. Some of these studies do provide a few statistics on the parameters we wish to calibrate. Whenever possible, we draw upon these works and include their findings as particular examples. As it is hard to generalize from these specific cases, we allow our parameters to vary through a broader range than those found in these studies.

The types of anomalies we would like to be able to mimic include: DDOS, flash crowd, alpha, outages and ingress/egress shift. Since we focus on detecting changes in traffic volume patterns, we do not include other anomalies such as worms and scans. A *DDOS* attack represents a flooding attack against a single destination. These attacks can have either a single source (DOS) or many distributed sources (DDOS). The latter occurs when many machines (called 'zombies') are compromised and a single attacker sends commands to all of the zombies enabling them to jointly flood a victim. A *flash crowd* occurs when there is a surge in demand for a service and is typically manifested by a large number of clients trying to access, and thus overwhelming, a popular Web site. Flash crowds can be predictable (e.g. a scheduled baseball game, or a software release) or unpredictable (e.g., news breaking event) [9].

An *alpha* anomaly refers to the transfer of a file(s) with an unusually large number of bytes. This typically involves one OD flow as there is a single source and a single destination. An *outage* refers to scenarios such as failures which can cause the load on a link to drop to zero. Such drops can either be short-lived or long-lived, and the short-lived outages are not infrequent since failures of one sort or another are fairly commonplace in the Internet today [1]. An *egress shift* occurs when the destination of an OD flow moves from one node to another. This can happen in a traffic matrix if there is a change in a BGP peering policy, or even a failure, as many OD flows can have multiple possible exit points from an ISP. Policy changes could also cause a shift of ingress point for a particular destination. In [21] the authors showed that traffic movement due to ingress or egress shifts, although not frequent, does indeed happen. None of these anomalies, other than DDOS attacks, are malicious. Yet all of them will generate potentially sudden and large shifts in traffic patterns, thus appearing anomalous.

In Table 1 we list our five parameters characterizing an anomaly. For each parameter we list the options for values, or value ranges, that the parameter can take on. We allow the duration to be anything from minutes, to hours, to days and for forever. We include the forever case as this includes the ingress and egress shift anomalies that will last until there is another policy change. Since [21] indicates these events are not that frequent, we can view the shift in traffic pattern as "permanent". The duration of an anomaly can vary throughout a large range, and it is unclear what the future will bring. Although most DDOS attacks observed, in the backscatter data of [16], lasted between 5 and 30 minutes, there were some outliers lasting less than 1 minute and others that lasted several days. Similarly, the majority of the DDOS events in the Abilene data of [11], lasted less than 20 minutes; a few outliers exceeds 2 hours. Alpha and flash crowd events could be of any length, although typically alpha events would be shorter than flash crowd events. In general, we do not include events whose order or magnitude of duration are less than minutes because we are adding these events on top of the Abilene data that is available to us with a minimum time interval of 5 minutes.

We change the traffic volume in two ways when anomalies occur. Sometimes we use a multiplicative factor δ that

is multiplied by the baseline traffic to generate the new traffic load. Using $\delta \sim 0$, we can easily capture outage scenarios. When an egress shift occurs, we assume that a subset of the prefixes travelling between the source and destination router are being shifted to a new exit point. This will shift a portion of the router-to-router traffic (as these policies are more likely to affect only a subset of the IP level prefixes) from the old OD pair to the new one. Removing 10%, for example, of the original OD flow's data is simply captured by using $\delta = 0.9$. This amount of traffic is added into the new OD flow using the constant additive term Δ. Allowing $1 \leq \delta \leq 2$, we can capture a variety of either alpha, flash crowd or DOS events. Note that because we are considering aggregate flows at the router to router level, doubling the traffic from an ingress router is already an enormous increase in traffic load. Large increases can occur when there are many end hosts behind the router that are involved in the anomaly (e.g., zombies, flash crowd). We don't consider $\delta > 2$ because such attacks are so obviously irregular that they are trivial to detect. We also allow a change in volume to be indicated by simply adding a constant factor, Δ, into the existing volume. This can capture the effect of a DDOS attack in which many zombies flood a victim at their maximum rate.

The *number of sources and destinations* indicates the number of OD flows involved in an anomaly. The notation $(1, 1)$ refers to a single source and a single destination. This could happen either for a DOS attack or an alpha event. The case of $(N, 1)$ arises for DDOS and flash crowds. In the case of a link failure, all the OD flows traversing the link are affected. The case of $(2, 2)$ can occur for an ingress or egress shift. By this we mean that there are two OD flows involved (that share either a common source or destination). One of these flows will experience an increase in volume, while the other experiences an equal amount of decrease. We do not include the case of (k, k) because we assume that one BGP policy will change at a time.

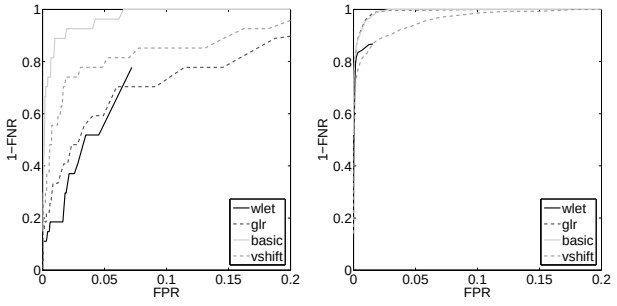
As mentioned earlier, our shape function can take on one of four possible forms: a ramp, exponential, square or step function. The shape function is multiplied by the extra volume amount before it is added onto the baseline traffic. This thus determines the ramp up and drop-off behavior of most anomalies. Not only are these shapes intuitively useful, but there is also some evidence for them in existing datasets. In [3] the authors found that a flash crowd can be characterized by a rapid rise in traffic that is then followed by a gradual drop-off over time. It also has been shown for flash crowd events that although their ramp up can be very quick, it is typically not instantaneous [9]. The initial increase of a DDOS attack could be captured by a ramp; this allows us the flexibility of representing scenarios in which the zombies reach their maximum flood rates in succession (medium slope) or via a very sharp rise [3] (steep slope). Outage anomalies could exhibit a near instantaneous drop in volume and thus we include the 'square' function. Alpha events could exhibit either a near instantaneous increase in volume or a ramp up. The step function is included to represent the ingress or egress shift anomalies because in these cases the change in traffic pattern is permanent (at least until the next policy change).

When we generate an anomaly we randomly select the values for these four parameters. Some combinations of them will look like the anomalies we have discussed. By varying each of the four characteristics in our generator, we can create a wide variety of anomalies.

5 Results

5.1 False Positive and False Negative Performance

We start by looking at the performance of our methods in the Abilene network. The abilene data contains 27 anomalies. Within each method, for each value of the threshold, we examine the entire traffic matrix (thus traversing all anomalies and non-anomalies). We can thus compute one false positive percentage and one false negative percentage for each threshold configuration of a scheme. The performance of our 4 methods on the Abilene data is depicted in the ROC curve of Figure 3(a). We see clearly that the basic method performs best. For a false positive rate of 7%, it misses no anomalies (100% true positives), while the next best method catches about 85% of the true anomalies for the same false positive rate. The wavelet method was unable to achieve 0% false negatives. Thus we observe an incomplete curve that does not reach the FNR $= 0$ limit, even with a huge threshold.



(a) Abilene (b) Synthetic

Figure 3: ROC curves using Abilene and Synthetic data

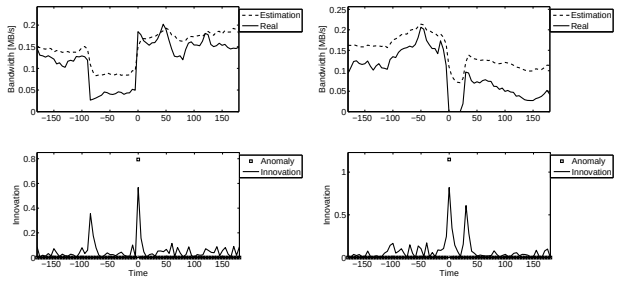
We now examine the performance of our algorithms using our synthetic anomaly generator. We generated about 500 different anomalies by varying the parameters of our generator. For these attacks, the duration was varied ran-

Parameter	Duration	Volume	Num (Src,Dst)	Shape
possible	Minutes	Δ	$(1, 1)$	Ramp
values	Hours	$1 \leq \delta \leq 2$	$(N, 1)$	Exponential
	Days	$\delta = 1.1$ or 0.9	$(2, 2)$	Square
	Forever	$\delta \sim 0$	all ODs on 1 link	Step

Table 1: Anomaly description parameters. Δ is an additive factor, δ is a multiplicative factor.

domly between 5 min and 2 hours. The volume of the original OD flow added on top of the anomalous OD pair ranged between 40% and 140%. The number of OD pairs involved was between 1 and 7 OD pairs per anomaly, and those selected were randomly chosen. The performance of our four schemes for these 500 scenarios is presented in Figure 3(b). For this data, the basic and GLR performed best and equivalently. It is interesting to note that the ranking of the four schemes, in terms of the ROC curve areas is not entirely consistent between the Abilene data and the synthetic ones. The main difference occurs with the GLR method that does not perform very well for the Abilene data but does for the synthetic data. The reason may lie in the statistical properties of the anomalies themselves. In our synthetic generator the way we add extra volume is equivalent to changing the mean of the OD flow for the duration of the anomaly. Since the GLR method is focused on detecting changes in the mean, it does well. It is possible that the anomalies in the Abilene data experience variance changes as well as mean changes. If this were true, it would explain why the vshift method is second best for the Abilene data. We leave the exploration of the statistical properties of the anomalous moments for future work.

When using marked traces we should be careful. There is always the risk that an anomaly is undetected or a normal behavior is marked as an anomaly. We conducted a visual inspection to remove any false positive(s) detected using the algorithms presented in [12]. We did not check for the false negatives. Consider the examples in Figures 4(a) and 4(b). On the top plots we show how a single OD pair evolves in time. The dashed line is our kalman filter estimation of this OD flow. We can see how it tracks the changes in the OD flow. On the bottom plots we show the residual process for each of these two example flows. We also include the markings produced by the labeling algorithm in [12]. Each box greater than 0 means that an anomaly was marked at this time. In figure 4(a) our residual process indicates that there were two anomalies, while the labeling procedure only marks one of them. According to our methodology above, we would thus label the first spike as a false positive since we use the labeling method to represent the "truth". This anomaly could easily have been a legitimate one. A similar situation arises for our second example flow. For these two examples, a simple



(a) OD pair #19 (b) OD pair #32

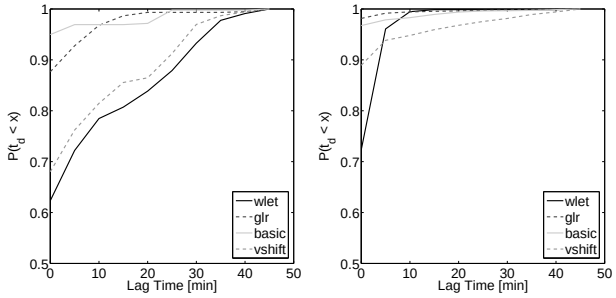
Figure 4: Example of the Innovation of an OD pair

visual inspection of the upper curve is enough to indicate that these events should have been True Positives since the two anomalies per flow indicate the beginning and ending of the anomaly. Because our algorithm may be able to detect events that the labeling algorithm of [12] does not, yet we use this algorithm to compute the FP ratio, it means that our computed false positive rate should be considered as an upper bound instead of the true value of the false positive ratio.

5.2 Detection Time

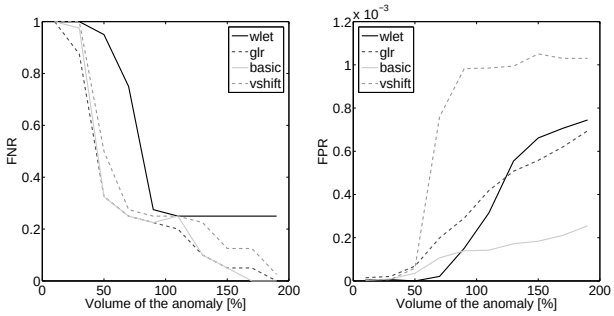
One of the critical performance aspects of any anomaly detection algorithm is the speed with which it can detect the anomaly. The onset of attacks and/or anomalies on the Internet today is extremely rapid thus creating real-time requirements for anomaly detection algorithms that are challenging. Few, if any, of the previous work we have seen, evaluate their algorithms in terms of detection time. We define detection lag as the time at which we detect a true anomaly minus the time the anomaly began. Since the underlying time unit of our traffic matrix data is 5 minutes, each additional lag corresponds to an increment of 5 minutes. (Note that our methods are not intrinsically tied to a 5 minute time interval.)

Each anomaly in the two sets (Abilene and synthetic) generates one sample detection lag value. We ensemble all these values and summarize them using a cumulative distribution. The results for the Abilene data are shown in Fig-



(a) Abilene (b) Synthetic

Figure 5: CDF of the detection lag using Abilene and Synthetic traces



(a) FNR (b) FPR

Figure 6: FNR and FPR as a function of the anomaly size

ure 5(a) while the results for the synthetic data are shown in Figure 5(b). In both cases, the basic method and GLR methods exhibit excellent detection times. In the case of Abilene data, the GLR method detected 90% of the anomalies with no lag, while the basic method detected 95% of the anomalies with no lag. For the synthetic data, the GLR curve is not visible because it lies on the line where the y-axis is 1 (underneath the basic curve). For the synthetic cases, both the GLR and basic methods were able to detection 100% of the anomalies with no lag at all. The wavelet analysis method performs less well; in particular there appear to be some difficult anomalies that can take over half an hour to detect. It is interesting that the vshift method performs well for the synthetic data but not for the Abilene data. In the synthetic case to detect an anomaly the volume should be high enough to raise an alarm as soon as it is observed otherwise it remains undetected and we cannot computes a lag time. Whereas in the Abilene data the vshift method is able to detect a subtle deviation in the statistics of the process and therefore need more samples to detect it. The motivation for using a wavelet method was an intuition that "an anomaly should diffuse itself at several time scales". However, in the results the anomalies appear differently at different time scales, and hence this approach was not very powerful in detecting anomalies. Other uses of wavelet methods in this context might prove more beneficial. For example, they might be useful for classifying anomalies since wavelet methods can give a rich description of the anomaly dynamics. This interesting problem is out of the scope of this paper.

5.3 Sensitivity Analysis

It is intuitive that enormous anomalies will be easy to detect and that very tiny ones are going to be missed. It is interesting to explore the space in between and see the impact of the false positive and false negative ratios as the volume of anomalies get smaller and smaller. In Figure 6(a) we plot the false negative ratio versus the percentage increase in the anomalous flow. To get a broad range of anomalies, for each tested volume level, we generate 50 anomalies with various start times or number of OD flows involved. We did this for 10 different volumes with $1.2 \leq \delta \leq 2$.

Figure 6(a) matches our intuition. If the OD flow increases by only 10 or 20% of its original value, we going to miss the anomalies. However the drop off of three methods is similar and fairly quick in the range of 40 - 100%. This implies that if the load from an ingress node doubled, it should be easy to catch all anomalies (low missed anomaly rate). Note that this justifies the fact that we don't use $\delta > 2$ in our synthetic anomaly generator.

The curve for the false positive rate (Figure 6(b)) is surprising. Initially we would have expected for this also to be a decreasing curve. But, as the anomaly becomes larger ($\delta \geq 1.5$) all the flows sharing a link have their estimates corrected by a large amount. Thus the error is spread inside the kalman filter to normal OD flows. This in turn increases the innovations leading to more false positives. This will not impact the ability to detect an anomaly but rather cloud the identity of the OD flow carrying the anomaly.

6 Conclusions

Our solution to tackling volume anomalies in large networks consists of many parts. First we select an interesting granularity level at which to perform anomaly detection, namely that of a traffic matrix. Second we use kalman filters to filter the predictable part of the traffic and to isolate the prediction error. The form of our model allows us to obtain the prediction error on the unobservable part of the network system (the OD flows) as well as for the observable part (link loads). Third, we proposed two detection schemes, but compared the performance of four of them. Finally we discuss how to make decisions about the pres-

ence of anomalies through the use of statistical hypothesis testing. We argue that the main measure of performance of an anomaly detector should be the ROC curve that explicitly captures the relationship between false positive and false negative rates. We give a mathematical foundation for this approach through the Neyman-Pearson theorem that identifies how to select decision thresholds when balancing the false positive and false negative tradeoff.

We considered four detection schemes that differ in the statistical change they seek to detect. Interestingly, but perhaps not surprisingly in retrospect, we found that the GLR method (whose goal is to detect changes in the mean) performs best when the anomaly is one that causes a change in the mean (e.g., in the synthetic cases). Similarly we found that the 'vshift' method performs better for the Abilene data than the synthetic data. We hypothesize that this occurs because the statistical properties of the anomalies themselves in the Abilene data contain changes in the variance of the residual traffic process. (We intend to verify this in future work by adding extra features into our synthetic anomaly generator that will alter the variance of the anomaly.) If the latter hypothesis is true, the implication is that the statistical change method that works best is the one checking the parameter that undergoes a deviation in the anomaly. On the one hand, this is motivation to do a study of the statistical properties of anomalies themselves. On the other hand, it suggests that the best method for network administrators could be a composite method that makes use of multiple different kinds of tests.

In our study, the wavelet based method did not perform well. Due to the popularity of wavelet based analyses, this raises interesting questions as to when wavelet analysis is and isn't useful for the problem domain of anomaly detection. Most importantly, from a practical point of view, it is good news that the simplest method performed best across all validation tests. This could be due to the fact that the Kalman model for the OD flows correctly models the normal traffic and thus the first filtering step is successful itself in isolating anomalies.

Aknowledgements

We would like to thank Anukool Lakhina for generously sharing his labeled Abilene traces with us. We are also grateful to Simon Crosby for talking to us about ROC curves.

References

[1] A.MARKOPOULOU, IANNACCONE, G., BHATTACHARYYA, S., CHUAH, C., AND DIOT, C. Characterization of Failures in an IP Backbone. In *In: IEEE Infocom* (March 2004).

[2] BARFORD, P., KLINE, J., PLONKA, D., AND RON, A. A signal analysis of network traffic anomalies. *ACM Sigcomm IMW* (2002).

[3] BARFORD, P., AND PLONKA, D. Characterisitics of of network traffic flow anomalies. In *ACM IMW* (Nov. 2001).

[4] BASSEVILLE, M., AND NIKIFOROV, I. Detection of abrupt changes: theory and application, 1993.

[5] EGAN, J. *Signal Detection Theory and ROC Analysis*. Academic Press, 1975.

[6] GUNNAR, A., JOHANSSON, M., AND TELKAMP, T. Traffic matrix estimation on a large ip backbone - a comparison on real data. In *ACM IMC* (Oct. 2004).

[7] HAWKINS, D. M., QQUI, P., AND KANG, C. W. The changepoint model for statistical process control. *Journal of Quality Technology 35*, 4 (october 2003).

[8] HUSSAIN, A. *Measurement and Spectral Analysis of Denial of Service Attacks*. PhD thesis, USC, May 2005.

[9] JUNG, J., KRISHNAMURTHY, B., AND RABINOVICH, M. Flash crowds and denial of service attacks: Characterization and implications for cdns and web sites. In *ACM WWW Conference* (May 2002).

[10] KAILATH, T., SAYED, A. H., HASSIBI, B., SAYED, A. H., AND HASSIBI, B. *Linear Estimation*. Prentice Hall, 2000.

[11] LAKHINA, A., CROVELLA, M., AND DIOT., C. Characterization of network-wide anomalies in traffic flows. In *ACM IMC* (2004).

[12] LAKHINA, A., CROVELLA, M., AND DIOT., C. Diagnosing network-wide traffic anomalies. In *ACM Sigcomm* (2004).

[13] LAKHINA, A., PAPAGIANNAKI, K., CROVELLA, M., DIOT, C., KOLACZYK, E., AND TAFT, N. Structural analysis of network traffic flows. In *ACM Sigmetrics* (2004).

[14] MALLAT, S. *A Wavelet Tour of Signal Processing*. Academic Press, 1999.

[15] MIRKOVIC, J., AND REIHER, P. A taxonomy of ddos attack and ddos defense mechanisms. In *ACM CCR* (April 2004).

[16] MOORE, D., VOELKER, G. M., AND SAVAGE, S. Inferring internet Denial-of-Service activity. In *Proceedings of the 10th USENIX Security Symposium* (2001), pp. 9–22.

[17] SOMMERS, J., YEGNESWARAN, V., AND BARFORD, P. A framework for malicious workload generation. In *IMC* (New York, NY, USA, 2004), ACM Press, pp. 82–87.

[18] SOULE, A., LAKHINA, A., TAFT, N., PAPAGIANNAKI, K., SALAMATIAN, K., NUCCI, A., CROVELLA, M., AND DIOT, C. Traffic matrices: Balancing measurements, inference and modeling. In *ACM Sigmetrics* (2005), ACM Press.

[19] SOULE, A., NUCCI, A., CRUZ, R., LEONARDI, E., AND TAFT, N. How to identify and estimate the largest traffic matrix elements in a dynamic environment. In *ACM Sigmetrics* (New York, 2004).

[20] SOULE, A., SALAMATIAN, K., AND TAFT, N. Traffic matrix tracking using kalman filters. *ACM LSNI Workshop* (2005).

[21] TEIXEIRA, R., DUFFIELD, N., REXFORD, J., AND ROUGHAN, M. Traffic matrix reloaded: Impact of routing changes. In *PAM* (2005).

[22] ZHANG, Y., ROUGHAN, M., DUFFIELD, N., AND GREENBERG, A. Fast accurate computation of large-scale ip traffic matrices from link loads. In *ACM Sigmretrics* (2003), ACM Press, pp. 206–217.

[23] ZHANG, Y., ROUGHAN, M., LUND, C., AND DONOHO, D. An information-theoretic approach to traffic matrix estimation. In *ACM Sigcomm* (2003), ACM Press, pp. 301–312.

[24] ZWEIG, M. H., AND CAMPBELL, G. Receiver-operating characteristic (roc) plots: a fundamental evaluation tool in clinical medicine. In *Clinical Chemisty* (1993), vol. 93(4).