Robust Anomaly Detection in Dynamic Networks *

Jing Wang† and Ioannis Ch. Paschalidis‡

***Abstract*—We propose two robust methods for anomaly detection in dynamic networks in which the properties of normal traffic are time-varying. We formulate the robust anomaly detection problem as a *binary composite hypothesis testing problem* and propose two methods: a *model-free* and a *model-based* one, leveraging techniques from the theory of large deviations. Both methods require a family of Probability Laws (PLs) that represent normal properties of traffic. We devise a 2-step procedure to estimate this family of PLs. We compare the performance of our robust methods and their vanilla counterparts, which assume that normal traffic is stationary, for two types of typical traffic patterns. We validate our methods using synthetic traffic traces obtained from a software tool we have developed for this purpose. Simulation results show that our robust methods perform better than their vanilla counterparts in dynamic networks.**

***Index Terms*— Robust statistical anomaly detection, large deviations theory, set covering, binary composite hypothesis testing.**

I. Introduction

A network anomaly is any potentially malicious traffic sequence that has implications for the security of the network. Although automated online traffic anomaly detection has received a lot of attention, this field is far from mature.

Network anomaly detection belongs to a broader field of system anomaly detection whose approaches can be roughly grouped into two classes: *signature-based anomaly detection*, where known patterns of past anomalies are used to identify ongoing anomalies [1, 2], and *change-based anomaly detection* that identifies patterns that substantially deviate from normal patterns of operations [3, 4]. [5] showed that the detection rates of systems based on pattern matching are below 70%. Furthermore, such systems cannot detect *zero-day attacks*, i.e., attacks not previously seen, and need constant (and expensive) updating to keep up with new attack signatures. In contrast, *change-based anomaly detection* methods are considered to be more economic and promising since they can identify novel attacks. In this work we focus on *change-based anomaly detection* methods, in particular on *statistical anomaly detection* that leverages statistical methods.

Standard *statistical anomaly detection* consists of two steps. The first step is to learn the "normal behavior" by analyzing past system behavior; usually a segment of records corresponding to normal system activity. The second step is to identify time instances where system behavior does not appear to be normal by monitoring the system continuously.

For anomaly detection in networks, [4] presents two methods to characterize normal behavior and to assess deviations from it based on the *Large Deviations Theory* (LDT) [6]. Both methods consider the traffic, which is a sequence of flows, as a sample path of an underlying stochastic process and compare current network traffic to some reference network traffic using LDT. One method, which is referred to as the *model-free* method, employs the method of types [6] to characterize the type (i.e., empirical measure) of an independent and identically distributed (i.i.d.) sequence of network flows. The other method, which is referred to as the *model-based* method, models traffic as a *Markov Modulated Process.* Both methods rely on *a stationarity assumption* postulating that the properties of normal traffic in networks do not change over time.

However, the *stationarity assumption* is rarely satisfied in contemporary networks. For example, it is well known that the Internet traffic of weekdays and weekends is very different [7]. Internet traffic is also influenced by macroscopic factors such as important holidays and elections. Similar phenomena arise in local area networks as well. We will call a network *dynamic* if its traffic exhibits time-varying behavior.

The challenges for anomaly detection of dynamic networks are two-fold. First, the methods used for learning the "normal behavior" are usually quite sensitive to the presence of non-stationarity. Second, the modeling and prediction of transient or time-dependent behavior is hard. To address these challenges, we generalize the vanilla *model-free* and *model-based* methods from [4] and develop what we call the *robust model-free* and *robust model-based* methods.

The structure of the paper is as follows. Sec. II presents a *model-free* and a *mode-based* method to solve a general binary composite hypothesis testing problem. Sec. III formulates the anomaly detection problem in dynamic networks as a binary composite hypothesis testing problem and applies the methods presented in Sec. II. Sec. IV introduces a software tool we developed for the generation of test data and gives an in-depth explanation of the simulated network. The development of this tool was motivated by the relative dearth of good quality traces with labeled anomalies. In the same section, we present results from our robust methods and compare them with their vanilla counterparts. Finally, Sec. V provides concluding remarks.

II. Binary composite hypothesis testing

In this section, we consider a problem of testing whether a sequence $\mathcal{G} = \{g^1, \ldots, g^n\}$ is a sample path of a stochastic process $\mathscr{G}$ (hypothesis $\mathcal{H}_0$). The stochastic process $\mathscr{G}$ is assumed to be discrete-time thus can be denoted as $\mathscr{G} = \{G^1, \ldots, G^n\}$.

* Research partially supported by the NSF under grants CNS-1239021, IIS-1237022, by the DOE under grant DE-FG52-06NA27490, by the ARO under grants W911NF-11-1-0227 and 61789-MA-MUR, by the ONR under grant N00014-10-1-0952, and by the NIH/NIGMS under grant GM093147.

† Division of Systems Engineering, Boston University, 8 St. Mary's St., Boston, MA 02215, wangjing@BU.EDU.

‡ Department of Electrical and Computer Engineering and Division of Systems Engineering, Boston University, 8 St. Mary's St., Boston, MA 02215, yannisp@bu.edu, http://ionia.bu.edu/.

All random variables G^i are discrete and their sample space is a finite alphabet $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{|\Sigma|}\}$, where $|\Sigma|$ denotes the cardinality of Σ. All observed symbols g^i belong to Σ, too. We assume the joint probability distribution $p_{\boldsymbol{\theta}}(G^1, \ldots, G^n)$ is parameterized by $\boldsymbol{\theta} = \{\theta^1, \ldots, \theta^n\} \in \Omega^n$, where Ω is the space where the elements of $\boldsymbol{\theta}$ take values. This problem is a *binary composite hypothesis testing problem.*

Because the joint distribution $p_{\boldsymbol{\theta}}(G^1, \ldots, G^n)$ becomes complex when n is large, we focus on two types of simplification. One is to assume all random variables G^i are i.i.d., the other is to assume the stochastic process $\mathscr{G}$ is a Markov chain.

A. A model-free method

We propose a *model-free* method that assumes the random variables G^i are i.i.d. Each G^i takes the value σ_j with probability $p_\theta^F(G^i = \sigma_j)$, $j = 1, \ldots, |\Sigma|$, which is parametrized by $\theta \in \Omega$. We refer to the vector $\mathbf{p}_\theta^F = (p_\theta^F(G^i = \sigma_1), \ldots, p_\theta^F(G^i = \sigma_{|\Sigma|}))$ as the *model-free* Probability Law (PL) associated with θ. Then the family of *model-free* PLs $\mathscr{P}^F = \{\mathbf{p}_\theta^F : \theta \in \Omega\}$ characterizes the stochastic process $\mathscr{G}$.

To characterize the observation $\mathcal{G}$, let

$$\mathscr{E}_F^{\mathcal{G}}(\sigma_j) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(g^i = \sigma_j), \qquad j = 1, \ldots, |\Sigma|, \tag{1}$$

where $\mathbf{1}(\cdot)$ is an indicator function. Then, an estimate for the underlying *model-free* PL based on the observation $\mathcal{G}$ is $\boldsymbol{\mathcal{E}}_F^{\mathcal{G}} = \{\mathscr{E}_F^{\mathcal{G}}(\sigma_j) : \ j = 1, \ldots, |\Sigma|\}$, which is called the *model-free* empirical measure of $\mathcal{G}$.

Suppose $\boldsymbol{\mu} = (\mu(\sigma_1), \ldots, \mu(\sigma_{|\Sigma|}))$ is a *model-free* PL and $\boldsymbol{\nu} = (\nu(\sigma_1), \ldots, \nu(\sigma_{|\Sigma|}))$ is a *model-free* empirical measure. To quantify the difference between $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$, we define the *model-free divergence* between $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ as

$$D_F(\boldsymbol{\nu} \| \boldsymbol{\mu}) \triangleq \sum_{j=1}^{|\Sigma|} \hat{\nu}(\sigma_j) \log \frac{\hat{\nu}(\sigma_j)}{\hat{\mu}(\sigma_j)}, \tag{2}$$

where $\hat{\nu}(\sigma_j) = \max(\nu(\sigma_j), \varepsilon)$ and $\hat{\mu}(\sigma_j) = \max(\nu(\sigma_j), \varepsilon), \forall j$ and ε is a small positive constant introduced to avoid underflow and division by zero.

Definition 1
(Model-Free Generalized Hoeffding Test). The model-free generalized Hoeffding test *[8] is to reject $\mathcal{H}_0$ when $\mathcal{G}$ is in the set:*

$$S_F^* = \{\mathcal{G} \mid \inf_{\theta \in \Omega} D_F(\boldsymbol{\mathcal{E}}_F^{\mathcal{G}} \| \mathbf{p}_\theta^F) \geq \lambda\},$$

where λ is a detection threshold and $\inf_{\theta\in\Omega} D_F(\boldsymbol{\mathcal{E}}_F^{\mathcal{G}} \| \mathbf{p}_\theta^F)$ is referred to as the generalized model-free divergence *between $\boldsymbol{\mathcal{E}}_F^{\mathcal{G}}$ and $\mathscr{P}^F = \{\mathbf{p}_\theta^F : \theta \in \Omega\}$.*

A similar definition has been proposed for robust localization in sensor networks [9]. One can show that this generalized Hoeffding test is asymptotically (as $n \to \infty$) optimal in a generalized Neyman-Pearson sense; we omit the technical details in the interest of space.

B. A model-based method

We now turn to the *model-based* method where the random process $\mathscr{G} = \{G^1, \ldots, G^n\}$ is assumed to be a Markov chain. Under this assumption, the joint distribution of $\mathscr{G}$ becomes $p_\theta(\mathscr{G} = \mathcal{G}) = p_\theta^B(g^1) \prod_{i=1}^{n-1} p_\theta^B(g^{i+1} \mid g^i)$, where $p_\theta^B(\cdot)$ is the initial distribution and $p_\theta^B(\cdot \mid \cdot)$ is the transition probability; all parametrized by $\theta \in \Omega$.

Let $p_\theta^B(\sigma_i, \sigma_j)$ be the probability of seeing two consecutive states (σ_i, σ_j). We refer to the matrix $\mathbf{P}_\theta^B = \{p_\theta^B(\sigma_i, \sigma_j)\}_{i,j=1}^{|\Sigma|}$ as the *model-based* PL associated with $\theta \in \Omega$. Then, the family of *model-based* PLs $\mathscr{P}^B = \{\mathbf{P}_\theta^B : \theta \in \Omega\}$ characterizes the stochastic process $\mathscr{G}$.

To characterize the observation $\mathcal{G}$, let

$$\mathscr{E}_B^{\mathcal{G}}(\sigma_i, \sigma_j) = \frac{1}{n}\sum_{l=2}^{n} \mathbf{1}(g^{l-1} = \sigma_i, g^l = \sigma_j), \ i, j = 1, \ldots, |\Sigma|. \tag{3}$$

We define the *model-based* empirical measure of $\mathcal{G}$ as the matrix $\boldsymbol{\mathcal{E}}_B^{\mathcal{G}} = \{\mathscr{E}_B^{\mathcal{G}}(\sigma_i, \sigma_j)\}_{i,j=1}^{|\Sigma|}$. The transition probability from σ_i to σ_j is simply $\mathscr{E}_B^{\mathcal{G}}(\sigma_j|\sigma_i) = \frac{\mathscr{E}_B^{\mathcal{G}}(\sigma_i,\sigma_j)}{\sum_{j=1}^{|\Sigma|} \mathscr{E}_B^{\mathcal{G}}(\sigma_i,\sigma_j)}$.

Suppose $\boldsymbol{\Pi} = \{\pi(\sigma_i, \sigma_j)\}_{i,j=1}^{|\Sigma|}$ is a *model-based* PL and $\mathbf{Q} = \{q(\sigma_i, \sigma_j)\}_{i,j=1}^{|\Sigma|}$ is a *model-based* empirical measure. Let $\hat{\pi}(\sigma_j|\sigma_i)$ and $\hat{q}(\sigma_j|\sigma_i)$ be the corresponding transition probabilities from σ_i to σ_j. Then, the *model-based divergence* between $\boldsymbol{\Pi}$ and $\mathbf{Q}$ is

$$D_B(\mathbf{Q} \parallel \boldsymbol{\Pi}) = \sum_{i=1}^{|\Sigma|}\sum_{j=1}^{|\Sigma|} \hat{q}(\sigma_i, \sigma_j) \log \frac{\hat{q}(\sigma_j|\sigma_i)}{\hat{\pi}(\sigma_j|\sigma_i)}, \tag{4}$$

where $\hat{q}(\sigma_i, \sigma_j) = \max(q(\sigma_i, \sigma_j), \varepsilon)$, $\hat{\pi}(\sigma_i, \sigma_j) = \max(\pi(\sigma_i, \sigma_j), \varepsilon)$ for some small positive constant ε introduced to avoid underflow and division by zero. Similar to the *model-free* case, we present the following definition:

Definition 2
(Model-Based Generalized Hoeffding Test). The model-based generalized Hoeffding test *is to reject $\mathcal{H}_0$ when $\mathcal{G}$ is in the set:*

$$S_B^* = \{\mathcal{G} \mid \inf_{\theta \in \Omega} D_B(\boldsymbol{\mathcal{E}}_B^{\mathcal{G}} \| \mathbf{P}_\theta^B) \geq \lambda\},$$

where λ is a detection threshold and $\inf_{\theta\in\Omega} D_B(\boldsymbol{\mathcal{E}}_B^{\mathcal{G}} \| \mathbf{P}_\theta^B)$ is referred to as the generalized model-based divergence *between $\boldsymbol{\mathcal{E}}_F^{\mathcal{G}}$ and $\mathscr{P}^B = \{\mathbf{P}_\theta^B : \theta \in \Omega\}$.*

In this case as well, asymptotic (generalized) Neyman-Pearson optimality can be established.

III. Network anomaly detection

In this section, we apply the *model-free* and *model-based generalized Hoeffding test* to the anomaly detection of dynamic networks. Although we focus on *host-based anomaly detection*, in which we monitor the incoming and outgoing packets of a server, our methods can be easily adapted to monitor routers. Based on the transmission time and the source-destination information in the headers, we first aggregate the network packets into flows. Our flow representation is slightly different

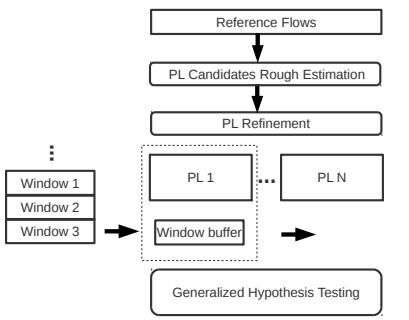

Fig. 1. Structure of the algorithms.

from that of commercial vendors like Cisco NetFlow. Hereafter, we will use "flows," "traffic," and "data" interchangeably.

Fig. 1 outlines the structure of our methods. We assume that the normal flows are governed by an underlying stochastic process $\mathscr{G}$. Since we do not know the families of *model-free* or *model-based* PLs for $\mathscr{G}$ explicitly, we assume a set of reference flows and propose a 2-step procedure to estimate them. We first inspect each feature separately to generate a family of candidate PLs, which is then reduced to a smaller family of PLs. We then aggregate the network flows that need to be evaluated into window buffers according to their start of transmission time. For each window, the algorithm applies the *model-free* and *model-based generalized Hoeffding test*.

A. Network traffic representation and flow aggregation

Let $\mathcal{S} = \{\mathbf{s}^1, \ldots, \mathbf{s}^{|\mathcal{S}|}\}$ denote the collection of all packets on the host which is monitored. In *host-based anomaly detection*, the host is either the source or the destination of the communication. As a result, only the source IP addresses for the incoming packets and the destination IP addresses for the outgoing packets are unknown; we call these the user IP addresses. Denote the user IP address in packet $\mathbf{s}^i$ as $\mathbf{x}^i$, whose format will be discussed later. The size of $\mathbf{s}^i$ is $b^i \in [0, \infty)$ in bytes and the start time of transmission is $t^i \in [0, \infty)$ in seconds. Using this convention, packet $\mathbf{s}^i$ can be represented as $(\mathbf{x}^i, b^i, t^i_s)$ for all $i = 1, \ldots, |\mathcal{S}|$.

Since packets are many, we consolidate network traffic by grouping packets into flows. We compile a sequence of packets $\mathbf{s}^1 = (\mathbf{x}^1, b^1, t^1_s), \ldots, \mathbf{s}^m = (\mathbf{x}^m, b^m, t^m_s)$ with $t^1_s < \cdots < t^m_s$ into a *flow* $\mathbf{f} = (\mathbf{x}, b, d_t, t)$ if $\mathbf{x} = \mathbf{x}^1 = \cdots = \mathbf{x}^m$ and $t^i_s - t^{i-1}_s < \delta_F$ for $i = 2, \ldots, m$ and some prescribed $\delta_F \in (0, \infty)$. Here, the *flow size* b is the sum of the sizes of the packets that comprise the flow. The *flow duration* is $d_t = t^m_s - t^1_s$. The *flow transmission time* t equals the start time of the first packet of the flow t^1_s. In this way, we can translate the large collection of packets $\mathcal{S}$ into a relatively small collection of flows $\mathcal{F} = \{\mathbf{f}^1 = (\mathbf{x}^1, b^1, d^1_t, t^1), \ldots, \mathbf{f}^{|\mathcal{F}|} = (\mathbf{x}^{|\mathcal{F}|}, b^{|\mathcal{F}|}, d^{|\mathcal{F}|}_t, t^{|\mathcal{F}|})\}$.

We first distill the "user space" into something more manageable while enabling us to characterize network behavior of user groups instead of just individual users. For simplicity of notation, we only consider IPv4 address. If $\mathbf{x}^i = (x^i_1, x^i_2, x^i_3, x^i_4) \in \{0, \ldots, 255\}^4$ and $\mathbf{x}^j = (x^j_1, x^j_2, x^j_3, x^j_4) \in \{0, \ldots, 255\}^4$ are two IPv4 addresses, the *distance* between them is defined as: $d(\mathbf{x}^i, \mathbf{x}^j) = |x^i_1 - x^j_1|256^3 + |x^i_2 - x^j_2|256^2 + |x^i_3 - x^j_3|256 + |x^i_4 - x^j_4|$. This metric can be easily extended to IPv6 addresses.

Suppose $\mathcal{X}$ is the set of unique IP addresses in $\mathcal{F}$. We apply typical K-means clustering on $\mathcal{X}$. For each $\mathbf{x} \in \mathcal{X}$, we thus obtain a cluster label $k(\mathbf{x})$. Suppose the cluster center for cluster k is $\bar{\mathbf{x}}^k$; then the distance of $\mathbf{x}$ to the corresponding cluster center is $d_a(\mathbf{x}) = d(\mathbf{x}, \bar{\mathbf{x}}^{k(\mathbf{x})})$. The cluster label $k(\mathbf{x})$ and distance to cluster center $d_a(\mathbf{x})$ are used to identify a user IP address $\mathbf{x}$, leading to our final representation of a flow as:

$$\mathbf{f} = (k(\mathbf{x}), d_a(\mathbf{x}), b, d_t, t). \tag{5}$$

B. Window aggregation

In our methods, flows in $\mathcal{F}$ are further aggregated into windows based on their *flow transmission time*s. A window is a detection unit that consists of flows in a continuous time range, which means the flows in the same window are evaluated together. Let h be the interval between the start points of two consecutive time windows and w_s be an appropriate window size; then the total number of windows is $n_w = \lceil (t^{|\mathcal{F}|} - t^1 - w_s)/h \rceil$. Flow $\mathbf{f}^i = (k(\mathbf{x}^i), d_a(\mathbf{x}^i), b^i, d^i_t, t^i)$ belongs to window j if $t^1 + (j-1)h \le t^i < t^1 + (j-1)h + w_s$, $j = 1, \ldots, n_w$. Let $\mathcal{F}_j$ be the set of all flows in window j, then $\mathcal{F} = \cup_{j=1}^{n_w} \mathcal{F}_j$ if $h \le w_s$. Note that there will be overlap between two consecutive time windows if $h < w_s$ and $\{\mathcal{F}_j : j = 1, \ldots, n_w\}$ is a partition of $\mathcal{F}$ if $h = w_s$.

For each $\mathbf{f}^i$, we first quantize $d_a(\mathbf{x}^i)$, b^i, and d^i_t to discrete values. For any collection of flows $\mathcal{F}$, let the range of $d_a(\mathbf{x}^i)$, $i = 1, \ldots, |\mathcal{F}|$ be $[d^{min}_a, d^{max}_a]$. We define a discrete alphabet $\Sigma_{d_a} = \left\{ d^{min}_a + (m - \frac{1}{2})(d^{max}_a - d^{min}_a)/|\Sigma_{d_a}| \right\}_{m=1,\ldots,|\Sigma_{d_a}|}$ for $d_a(\mathbf{x}^i)$, where $|\Sigma_{d_a}|$ is the total number of quantization levels for $d_a(\mathbf{x}^i)$. For features b^i and d^i_t, Σ_b and Σ_{d_t} are defined similarly. We quantize $d_a(\mathbf{x}^i)$, b^i, and d^i_t in $\mathbf{f}^i$ to the closest symbol in the discrete alphabets Σ_{d_a}, Σ_b, and Σ_{d_t}, respectively. Suppose also the total number of user clusters is K.

After quantization, each tuple of $(k(\mathbf{x}), d_a(\mathbf{x}), b, d_t)$ corresponds to a symbol in $\Sigma = \{1, \ldots, K\} \times \Sigma_{d_a} \times \Sigma_b \times \Sigma_{d_t}$. Denoting by $\mathbf{g}^i$ the corresponding symbol of $\mathbf{f}^i$ and by $\mathcal{G}_j$ the counterpart of $\mathcal{F}_j$, we number the symbols in $\mathbf{g}^i$ corresponding to $k(\mathbf{x}^i)$, $d_a(\mathbf{x}^i)$, b^i, and d^i_t as features 1, 2, 3, 4.

For each window j, an empirical measure of $\mathcal{G}_j$ is calculated. Because the calculation of empirical measure is the same for each window, we refer to $\mathcal{G}_j$ as $\mathcal{G}$.

C. Model-free method

For anomaly detection of dynamic networks, this section presents a *model-free* method that uses the *model-free generalized Hoeffding test* described in Sec. II-A. In each window, the data $\mathcal{G} = \{\mathbf{g}^1, \ldots, \mathbf{g}^n\}$ are viewed as n i.i.d. observations of a random variable G that is drawn from a family of *model-free* PLs parametrized by θ. Assuming that $\Omega = \{\omega_1, \ldots, \omega_{|\Omega|}\}$ is a finite set, $\mathscr{P}^F = \{\mathbf{p}^F_\theta : \theta \in \Omega\}$ is a finite set thus and can be written as $\mathscr{P}^F = \{\mathbf{p}^F_1, \ldots, \mathbf{p}^F_{|\Omega|}\}$.

$\mathscr{P}^F$ is usually not available and needs to be estimated from a reference $\mathcal{G}_{ref}$. Each flow in $\mathcal{G}_{ref}$ is governed by one of the PLs in $\mathscr{P}^F$. Let $\mathcal{M}^F(j)$ be the set of indices of flows in $\mathcal{G}_{ref}$

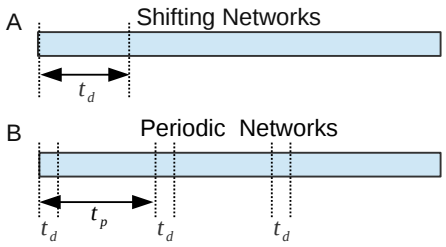

Fig. 2. The relationship between *flow transmission time* t and indices of flows $\mathcal{M}(j)$ governed by PL j for *shifting networks* (A) and *periodic networks* (B).

governed by PL j. If $\mathcal{M}^F(j)$ is known, we use (1) to estimate $\mathbf{p}_j$. We will discuss how to calculate $\mathcal{M}^F(j)$ later.

Let $\boldsymbol{\mathcal{E}}_F^{\mathcal{G}}$ be the *model-free* empirical measure of $\mathcal{G}$ calculated using (1). The *model-free generalized Hoeffding test* of Definition 1 becomes

$$\mathcal{S}_F^* = \{\mathcal{G} | \min_{j=1,\ldots,|\Omega|} D_F(\boldsymbol{\mathcal{E}}_F^{\mathcal{G}} \parallel \mathbf{p}_j^F) \geq \lambda\}. \tag{6}$$

D. Model-based method

Likewise, this section presents a *model-based* method that uses the *model-based generalized Hoeffding test* described in Sec. II-B. In each window, we consider the data $\mathcal{G} = \{\mathbf{g}^1, \ldots, \mathbf{g}^n\}$ as generated from a Markov chain with transition probabilities $p_\theta^B(\cdot|\cdot)$. Again $\theta \in \Omega$ is a parameter for the transition probabilities. Similar to the *model-free* approach, we assume that Ω is a finite set $\Omega = \{\omega_1, \ldots, \omega_{|\Omega|}\}$. Let $\mathbf{P}_\theta^B$ denote the PL under θ. Define $\mathscr{P}^B = \left\{\mathbf{P}_1^B, \ldots, \mathbf{P}_{|\Omega|}^B\right\}$ as a finite set of model-based PLs.

$\mathscr{P}^B$ needs to be estimated from the reference data $\mathcal{G}_{ref}$. Considering two consecutive flows $\mathbf{g}^l$ and $\mathbf{g}^{l+1}$, the transition from $\mathbf{g}^l$ to $\mathbf{g}^{l+1}$ must be governed by one of the PL in $\mathscr{P}^B$. We say both $\mathbf{g}^l$ and $\mathbf{g}^{l+1}$ are governed by this PL. Define $\mathcal{M}^B(j)$ as the index set of flows governed by PL j for the *model-based* method. If $\mathcal{M}^B(j)$ is known, we can apply (3) to calculate the *mode-based* empirical measure.

Let $\boldsymbol{\mathcal{E}}_B^{\mathcal{G}}$ be the empirical measure (cf. (3)). The *model-based generalized Hoeffding test* of Definition 2 becomes

$$\mathcal{S}_B^* = \{\mathcal{G} | \min_{j=1,\ldots,|\Omega|} D_B(\boldsymbol{\mathcal{E}}_B^{\mathcal{G}} \parallel \mathbf{P}_j^B) \geq \lambda\}. \tag{7}$$

E. PLs for different dynamic networks

As described in Secs. III-C and III-D, if $\mathcal{M}^F(j)$ and $\mathcal{M}^B(j)$ are known, we can use (1) and (3) to estimate the *model-free* and *model-based* PL j, respectively. Set $\mathcal{M}(j) = \mathcal{M}^F(j) = \mathcal{M}^B(j)$ for $j = 1, \ldots, |\Omega|$, which is motivated by the fact that $\mathcal{M}(j)$ reflects a stationary mode of the network and is independent of the method selected to represent the flows. Let $\mathbf{t} = \{t^1, \ldots, t^n\}$ be the set of *flow transmission times* of flows in $\mathcal{G}_{ref}$, where t^i corresponds to the flow transmission time of $\mathbf{g}^i$. Based on the information in $\mathbf{t}$, we can propose candidate $\mathcal{M}(j)$. This section describes methods to identify $\mathcal{M}(j)$ for two common types of dynamic networks.

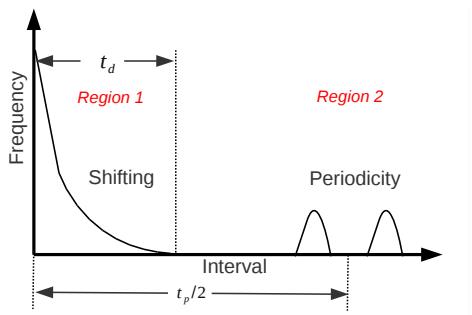

Fig. 3. Histogram of intervals between two consecutive flows with a specific feature quantized to the same discrete value.

1) Shifting networks: For the first type of dynamic network, properties of normal traffic shift with respect to time. This means that the flows close in time are more likely to be governed by the same PL. We divide $\mathcal{G}_{ref}$ into segments, each with a duration of a prescribed value t_d. The flows in each segment are used to estimate one PL (Fig. 2A). The flow indices in segment j are

$$M(j) = \{i : t^1 + (j-1)t_d \leq t^i < t^1 + jt_d\}, \tag{8}$$

where $j = 1, \ldots, \lfloor (t^n - t^1)/t_d \rfloor$. t_d characterizes how quickly we expect the statistical properties of the traffic to shift. Larger t_d indicates a slower shifting of traffic properties in the network. One can choose a variety of t_d, and for each t_d's generate the corresponding $\mathcal{M}(j)$ and the resulting PLs.

2) Periodic networks: In the second type of dynamic networks, the properties of the normal traffic change periodically. In addition to having close *flow transmission times*, two flows can be governed by the same PL when the difference of their *flow transmission times* equals the period (Fig. 2B). Let t_d characterize shifts within the period and let t_p be the period. For $j = 1, \ldots, \lfloor t_p/t_d \rfloor$, let

$$M(j) = \cup_{k \in \mathcal{K}_j} \left\{i : kt_p + (j-1)t_d \leq t^i < kt_p + jt_d\right\}, \tag{9}$$

where $\mathcal{K}_j = \left\{k : kt_p + (j-1)t_d > t^1 \text{ and } kt_p + jt_d < t^n\right\}$. Again, we can choose a variety of t_p's and t_d's with each combination contributing to $\lfloor t_p/t_d \rfloor$ PLs.

Practical networks can exhibit both types of nonstationary behavior described above. Moreover, the periodicity and the degree of shift may change over time, too. To increase the robustness of the set of estimated PLs to these non-stationarities, we first propose a large collection of candidates and then refine it using integer programming.

F. Estimation of candidate PLs

We now present a procedure to generate a set of candidate PLs by inspecting each feature separately. We have a sequence of reference quantized flows $\mathcal{G}_{ref} = \left\{\mathbf{g}^1, \ldots, \mathbf{g}^{|\mathcal{G}_{ref}|}\right\}$ and the corresponding *flow transmission times* are $\mathbf{t} = \left\{t^1, \ldots, t^{|\mathcal{G}_{ref}|}\right\}$. Recall that each quantized flow $\mathbf{g}^i$ consists of quantized values of a cluster label $k(\mathbf{x}^i)$, a distance to cluster center $d_a(\mathbf{x}^i)$, a flow size b^i and a flow duration d_t^i, which are called features $1, \ldots, 4$, respectively. For all $a = 1, 2, 3, 4$, let $\mathscr{G}_a = \left\{g_a^1, \ldots, g_a^{|\mathcal{G}_{ref}|}\right\}$ be the sequence of quantized feature a for each flow in $\mathcal{G}_{ref}$. For $a = 1, 2, 3, 4$ and $b = 1, \ldots, |\Sigma_a|$, we say a flow $\mathbf{g}^i$ belongs to *channel* a–b if g_a^i equals σ_b^a. We

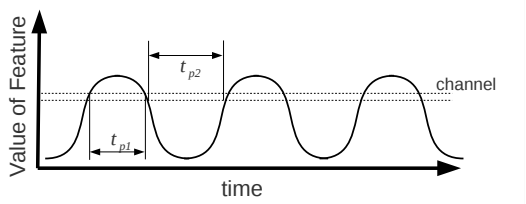

Fig. 4. Illustration of the peaks in Region 2 of Fig. 3.

first analyze each channel separately to get a rough estimate of t_d and t_p. Then, channels corresponding to the same feature are aggregated to generate a combined estimate for this feature.

We define $\mathscr{I}_{ab} = \left\{t^i : g_a^i = \sigma_b^a\right\}$ as the sorted sequence of *flow transmission times* for flows in *channel* a–b. The interval between two consecutive flows in *channel* a–b is $\tau_{ab}^k = t_{ab}^k - t_{ab}^{k-1}, k = 2, \ldots, |\mathscr{I}_{ab}|$, where t_{ab}^k is the kth element in $\mathscr{I}_{ab}$.

For shifting networks, since the majority of flows in each channel belong to a continuous time range, the intervals between two consecutive flows are small. The histogram of the intervals $\{\tau_{ab}^k : k = 2, \ldots, |\mathscr{I}_{ab}|\}$ will have a heavy head and a small tail (Region 1 in Fig. 3). The end of the tail can be used as an upper bound on the interval between two consecutive flows and is a good option for t_d.

For periodic networks, the histogram for intervals in $\{\tau_{ab}^k : k = 2, \ldots, |\mathscr{I}_{ab}|\}$ is also heavily skewed to small intervals, thus the t_d can be estimated in the same way as with shifting networks. However, the intervals between two consecutive flows can be large. Fig. 4 shows an example of a feature that exhibits periodicity. There will be two peaks around t_{p1} and t_{p2} in the histogram of intervals for flows whose values are between the two dashed lines. We can select t_p such that $(t_{p1} + t_{p2})/2 \approx t_p/2$. There can be a single or more than two peaks because of the randomness in the network; in either case, we can choose the mean of all peaks as an approximation of $t_p/2$ (cf. Fig. 3).

Denote the estimate of t_d and t_p based on channel a–b as t_d^{ab} and t_p^{ab}, respectively. We use the subscript $\{d, p\}$ to unify the notations for both estimates. $t_p^{ab} = 0$ if no periodicity is found in channel a–b. For $a = 1, 2, 3, 4$, let $\mathscr{T}_{\{d,p\}}^a = \{t_{\{d,p\}}^{ab} : b = 1, \ldots, |\Sigma_a|, \text{ and } t_{\{d,p\}}^{ab} > 0\}$ be the collection of estimates for all channels of feature a. We define the combined estimate of t_d and t_p for feature a as $t_{\{d,p\}}^a = \textsc{Mean}\left(\mathscr{T}_{\{d,p\}}^a\right)$, where $\textsc{Mean}(\cdot)$ calculates the sample mean of a set.

If $\mathscr{T}_p^a$ is empty, the network is non-periodic according to feature a, thus, a family of candidate PLs can be generated using t_d^a and (8). Otherwise, the network is periodic according to feature a, and a family of candidate PLs can be generated using t_d^a, t_p^a, and (9). In addition, in case that some prior knowledge of t_d and t_p is available, the family of candidate PLs can include the PLs calculated based on this prior knowledge.

G. PL refinement with integer programming

Since PLs are estimated from a reference trace $\mathcal{G}_{ref}$, we need to avoid "overfitting" the data. The larger the family of PLs is, the more likely it is to overfit $\mathcal{G}_{ref}$. Furthermore, a smaller family of PLs reduces the computation cost. With this motivation, this section introduces a method to refine the family of candidate PLs.

For simplicity, we only describe the procedure for the *model-free* method. The procedure for the *model-based* method is similar. In the rest of the paper, the divergence between a collection of flows and a PL is equivalent to the divergence between the empirical measure of these flows and the PL.

Suppose the family (namely the set) of candidate PLs is the set $\mathcal{P} = \{\mathbf{p}_1^F, \ldots, \mathbf{p}_N^F\}$ of cardinality N. Because no alarm should be reported for $\mathcal{G}_{ref}$, or segments of $\mathcal{G}_{ref}$, our *primary objective* is to choose the smallest set $\mathscr{P}^F \subseteq \mathcal{P}$ such that there is no alarm for $\mathcal{G}_{ref}$. We aggregate $\mathcal{G}_{ref}$ into M windows using the techniques of Sec. III-B and denote the data in window i as $\mathcal{G}_{ref}^i$. Let $D_{ij} = D_F(\boldsymbol{\mathcal{E}}^{\mathcal{G}_{ref}^i} \parallel \mathbf{p}_j^F)$ be the divergence between flows in window i and PL j for $i = 1, \ldots, M$ and $j = 1, \ldots, N$. We say window i is covered (namely, reported as normal) by PL j if $D_{ij} \leq \lambda$. With this definition, the primary objective becomes to select the minimum number of PLs to cover all the windows.

There may be more than one subsets of $\mathcal{P}$ having the same cardinality and covering all windows. We propose a *secondary objective* characterizing the variation of a set of PLs. Let $N_j = \{i : \ D_{ij} \leq \lambda\}$ be the index set of windows covered by PL j and denote by $N_j^{(1)}, \ldots, N_j^{(|N_j|)}$ the ordered elements of N_j. Define $\mathscr{D}_j = \{N_j^{(i)} - N_j^{(i-1)} : \ i = 2, \ldots, |N_j|\}$ the set of differences between consecutive window indices covered by PL j. The *coefficient of variation* for PL j is defined as $c_v^j = \textsc{Std}(\mathscr{D}_j)/\textsc{Mean}(\mathscr{D}_j)$, where $\textsc{Std}(\mathscr{D}_j)$ and $\textsc{Mean}(\mathscr{D}_j)$ are the sample standard deviation and mean of set $\mathscr{D}_j$, respectively. A smaller *coefficient of variation* means that the PL is more "regular." The *secondary objective* to minimize the sum of *coefficients of variation* for selected PLs. We formulate PL selection as a *weighted set cover problem* in which the weight of PL j is $1 + \gamma c_v^j$, where γ is a small weight for the secondary objective. Let x_i be the 0–1 variable indicating whether PL i is selected or not; let $\mathbf{x} = (x_1, \ldots, x_N)$. Let $\mathbf{A} = \{a_{ij}\}$ be an $M \times N$ matrix whose (i, j)th element a_{ij} is set to 1 if $D_{ij} \leq \lambda$ and to 0 otherwise. Here, λ is the same threshold we used in (6). Let $\mathbf{c}_v = (c_v^1, \ldots, c_v^N)$. The selection of PLs can be formulated as the following integer programming problem:

$$\begin{aligned} \min \quad & \mathbf{1}'\mathbf{x} + \gamma \mathbf{c}_v'\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{1}, \\ & x_j \in \{0, 1\}, \ j = 1, \ldots, N, \end{aligned} \tag{10}$$

where $\mathbf{1}$ is a vector of ones. The cost function equals a weighted sum of the *primary cost* $\mathbf{1}'\mathbf{x}$ and the *secondary cost* $\mathbf{c}_v'\mathbf{x}$. The first constraint enforces there is no alarm for $\mathcal{G}_{ref}^i$, $i = 1, \ldots, M$.

Because (10) is NP-hard, we propose a *greedy algorithm* to solve it (Algorithm 1). HEURISTICREFINEPL is the main procedure whose parameters are $\mathbf{A}$, $\mathbf{c}_v$, a discount ratio $r < 1$, and a termination threshold γ_{th}. In each iteration, the algorithm decreases γ by a ratio r and calls the GREEDYSETCOVER procedure to solve (10). The algorithm terminates when $\gamma < \gamma_{th}$. In the initial iterations, the weight γ for the secondary cost is large so that the algorithm explores solutions which select PLs

function HEURISTICREFINEPL($\mathbf{A}$, $\mathbf{c}_v$, r, γ_{th})
  Init: bestCost := ∞, $\gamma := 1$, $\mathbf{x}^* := 0$
  while $\gamma > \gamma_{th}$ **do**
    $\mathbf{x}$:= GREEDYSOLVE($\mathbf{A}$, γ, $\mathbf{c}_v$), $\gamma := r\gamma$
    if $\mathbf{1}'\mathbf{x} + \gamma_{th}\mathbf{c}_v'\mathbf{x} <$ bestCost **then**
      bestCost := $\mathbf{1}'\mathbf{x} + \gamma_{th}\mathbf{c}_v'\mathbf{x}$
      $\mathbf{x}^* := \mathbf{x}$
    end if
  end while
  return $\mathbf{x}^*$
end function
function GREEDYSETCOVER($\mathbf{A}$, γ, $\mathbf{c}_v$)
  Init: $\mathbf{x}^0 := \mathbf{0}$, $C := \varnothing$
  while $|C| < M$ **do**
    $j^+ := \arg\max_{j:\mathbf{x}[j]=0} \frac{\sum_{i \notin C} a_{ij}}{1+\gamma \mathbf{c}_v[j]}$
    $\mathbf{x}[j^+] := 1$, $C := C \cup \{i : a_{ij^+} = 1\}$
  end while
  return $\mathbf{x}$
end function

Algorithm 1: Greedy algorithm for PL refinement.

with less variation. Later, the weight γ decreases to insure that the primary objective plays the main role. GREEDYSETCOVER uses the ratio of the number of uncovered windows a PL can cover and the cost $1 + \gamma c_v$ as heuristics, where c_v is the corresponding *coefficient of variation*. GREEDYSETCOVER will add the PL with the maximum heuristic value to $\mathscr{P}^F$ until all windows are covered by the PLs in $\mathscr{P}^F$. Suppose the return value of HEURISTICREFINEPL is $\mathbf{x}^*$. Then, the refined family of PLs is $\mathscr{P}^F = \{\mathbf{p}_j^F : x_j^* > 0, j = 1, \ldots, N\}$.

IV. SIMULATION RESULTS

Lacking data with annotated anomalies is a common problem for validation of network anomaly methods. We developed an open source software package SADIT [10] to provide flow-level datasets with annotated anomalies. Based on the *fs*-simulator [11], SADIT simulates the normal and abnormal flows in networks efficiently.

We simulated the network of a small organization (Fig. 5). The network is clearly partitioned into an internal network and several Internet nodes. The internal network consists of 8 normal nodes *CT1-CT8* and 1 server *SRV* containing some sensitive information. There are also three Internet nodes *INT1-INT3* that access the internal network through a gateway (*GATEWAY*). For all links, the link capacity is 10 Mb/s and the delay is 0.01 s.

All internal and Internet nodes communicate with the *SRV* and there is no communication between other nodes. The normal flows from all nodes to *SRV* have the same characteristics. The size of the normal flows follows a Gaussian distribution $N(m(t), \sigma^2)$. The arrival process of flows is a Poisson process with arrival rate $\lambda(t)$. Both $m(t)$ and $\lambda(t)$ change with time t. We consider two types of changing patterns for normal traffic: *shifting pattern* and *day-night pattern*. For both patterns, we monitor the traffic on the server and evaluate the performance of

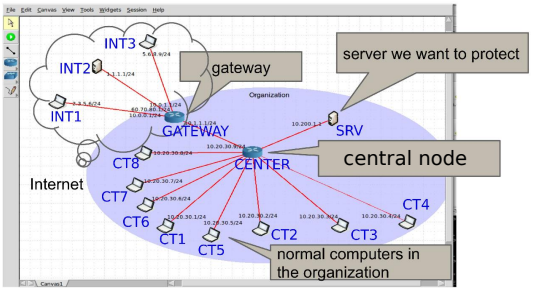

Fig. 5. Simulation settings.

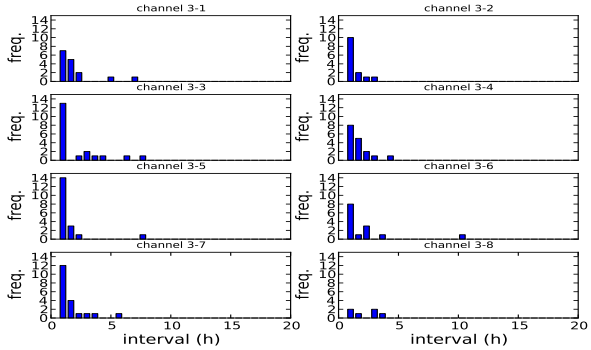

Fig. 6. Histogram of intervals for a network in which flow size exhibits a *shifting pattern*. Each plot corresponds to a channel. There are 30 bins and all plots share the x-axis. For all plots, the first bin is not plotted because it is significantly higher than the rest of the bins.

the robust *model-free* and *model-based* methods for an anomaly in which a user downloads substantially large files.

A. Shifting pattern

Since the average Internet speed is increasing, user applications become more and more bandwidth-consuming. From the flow perspective, this means that the average flow size is shifting to higher values.

As a simple model, we assume this shift is linear with respect to time and the flow arrival rate is still stationary. We further assume that all users exhibit the same shift pattern. As a result, $m(t)$ is a linear function of time as $m(t) = at + b$, where a and b are two parameters characterizing the shift of the traffic and $\lambda(t)$ is a constant. In our simulation, we set $b = 4$ Mb, $a = 3.6$ Kb/h, and $\sigma^2 = 0.01$ for all users. The flow arrival rate is constant and $\lambda(t) = 0.1$ fps (flows per second). Using this *shifting pattern*, we generate reference traffic $\mathcal{G}_{ref}$ for one week (168 hours).

To obtain the traffic patterns in $\mathcal{G}_{ref}$, the procedure of Sec. III-F is applied to identify t_d and t_p. For window aggregation, both the window size w_s and the interval h between two consecutive windows are 2000 s. The number of user clusters is $K = 2$. For the quantization, the number of quantization levels for the distance to cluster center, the flow size, and the flow duration (features 2, 3, 4) are 2, 2, and 8, respectively.

The values of t_p and t_d can be estimated by inspecting the histograms of the 8 channels of feature 3 (cf. Fig. 6). Most channels have tails for small interval values but no peak caused by periodicity, which clearly indicates that the normal pattern is shifting and non-periodic, thus, t_p is unnecessary.

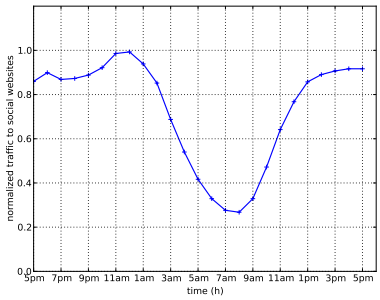

Fig. 11. Traffic pattern of social network websites.

The combined estimate of t_d based on flow size is $t_d^3 = 3.89h$. There are 43 candidate *model-free* and *model-based* PLs, each PL being calculated using a segment of $\mathcal{G}_{ref}$.

For the *model-free* method, because $m(t)$ and $\lambda(t)$ shift with time, the PL calculated based on the flows in one interval has very small divergence during this interval, but the divergence becomes larger for times away from this interval (cf. Fig. 7A). There are 4 PLs selected by the PL refinement procedure when the detection threshold is set to $\lambda = 2$ (Fig. 7B).

We say PL j^* is *active* during window i if its divergence with traffic in this window is the smallest among all selected PLs, namely $j^* = \arg\min_j D_{ij}$. Each selected PL is active for a continuous range of time, which is consistent with the fact that the traffic pattern is shifting (Fig. 7C). The active PL oscillates before it switches to a new PL. Although the number of PLs is reduced after refinement, the *generalized model-free divergence* between each $\mathcal{G}_{ref}^i$ and the set of selected PLs is very close to the corresponding divergence between $\mathcal{G}_{ref}^i$ and the set of all candidate PLs (Fig. 7D). This implies that the reduced set represents $\mathcal{G}_{ref}$ well.

For the *model-based* method, 6 PLs are selected by the PL refinement procedure when $\lambda = 2$ (Fig. 8A,B). Each PL is active for a continuous range of time and we can observe similar oscillations as in the *model-free* method during the transition between two active PLs (Fig. 8C). Again, the *model-based generalized divergence* between each $\mathcal{G}_{ref}^i$ and either the set of selected PLs or the set of all candidate PLs is very similar (Fig. 8D).

B. Day-night pattern

Another type of dynamic pattern is *a day-night pattern*. One example of *day-night pattern* is the traffic of web servers. People browse websites more frequently during the day than the night, thus, the normal traffic to web servers exhibits *diurnal variation*. Fig. 11 shows the normalized average traffic to American social websites over a day [12].

We assume that the flow arrival rate and the mean flow size have the same *day-night pattern*. Let $p(t)$ be the function shown in Fig. 11, and assume $\lambda(t) = \Lambda p(t)$ and $m(t) = M_p p(t)$, where Λ and M_p are the peak arrival rate and the peak mean flow size. In our simulation, we set $M_p = 4$ Mb, $\sigma^2 = 0.01$, and $\Lambda = 0.1$ fps for all users. Using this *day-night pattern*, we generate reference traffic $\mathcal{G}_{ref}$ for one week (168 hours) whose start time is 5 pm. Again, an estimation procedure is applied to estimate t_d and t_p. The parameters for window aggregation and quantization are the same as in Sec. IV-A.

The period can be estimated by inspecting the histograms of the 8 channels of feature 3, namely the flow size feature (Fig. 12). Peaks can be observed in all channels except for channel 3-4 and 3-5. The combined estimate of the period based on flow size is $t_p^3 = 24.56$ h, which has only 2.3% error with the real value of 24 h.

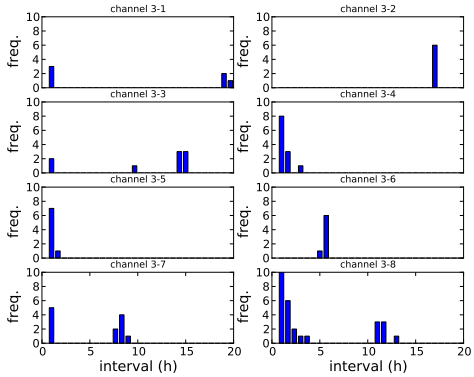

Fig. 12. Histogram of intervals for the *day-night pattern*. Each subfigure corresponds to one channel. All subfigures share the x-axis and the total number of bins is $q = 30$. The first bin is not plotted in the histogram because it is significantly higher than the rest of the bins.

For the *model-free* method, there are 64 candidate *model-free* PLs proposed in the estimation stage. The *model-free divergence* between each window and each candidate PL is a periodic function of time, too. Some PLs have smaller divergence during the day and some others have smaller divergence during the night (cf. Fig. 9A). However, no PL has small divergence for all windows. 3 PLs out of the 64 candidates are selected when the detection threshold is $\lambda = 0.6$ (cf. Fig. 9B). The 3 selected PLs are active during day, night, and the *transitional time*, respectively (cf. Fig. 9C for the active PLs of all windows). For all windows, the *model-free generalized divergence* between $\mathcal{G}_{ref}$ and all candidate PLs is very close to the divergence between $\mathcal{G}_{ref}$ and only the selected PLs (Fig. 9D). The difference is relatively larger during the *transitional time* between day and night. This is because the network is more dynamic during this *transitional time*, thus, more PLs are required to represent the network accurately.

For the *model-based* method, there are 64 candidate *model-based* PLs, too. Similar to the *model-free* method, the *model-based* divergence between all candidate PLs and flows in each window in $\mathcal{G}_{ref}$ is periodic (Fig. 10A). There is no PL that can represent all the reference data $\mathcal{G}_{ref}$. 2 PLs are selected when $\lambda = 0.4$ (Fig. 10B). One PL is active during the *transitional time* and the other is active during the *stationary time*, which consists of both day and night (Fig. 10C). As before, the divergence between each $\mathcal{G}_{ref}^i$ and all candidate PLs is similar to the divergence between $\mathcal{G}_{ref}^i$ and just the selected PLs (Fig. 10D).

The results show that the PL refinement procedure is effective and the refined family of PLs is meaningful. Each PL in the

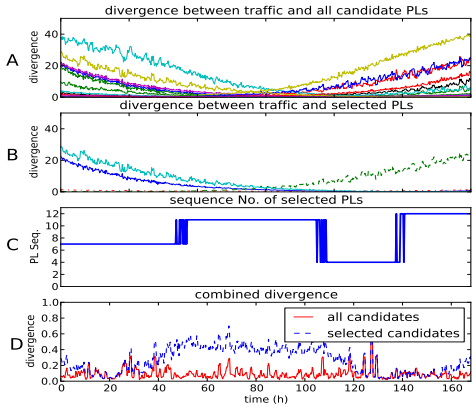

Fig. 7. Results of PL refinement for *model-free* PLs in a network with *shifting pattern*. (A) and (B) plot the *model-free divergence* between $\mathcal{G}^i_{ref}$ and all candidate PLs or just selected PLs, respectively. (C) depicts the active PL at each time. (D) plots the *model-free generalized divergence* between $\mathcal{G}^i_{ref}$ and all candidate PLs or selected PLs. The x-axis corresponds to the start time of the various windows i.

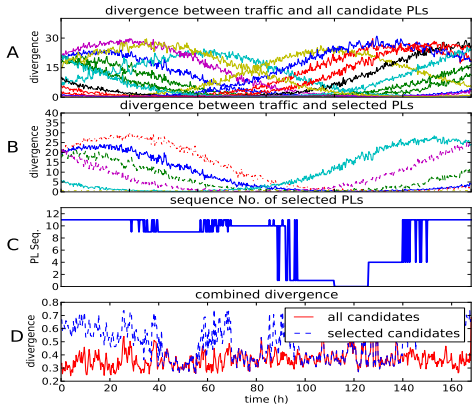

Fig. 8. Results of PL refinement for *model-based* PLs in a network with *shifting pattern*. (A) and (B) plot the *model-based* divergence between $\mathcal{G}^i_{ref}$ and all candidate PLs or just selected PLs, respectively. (C) plots the *active* PL for each window. (D) plots the *model-based generalized divergence* between $\mathcal{G}^i_{ref}$ and all candidate PLs/selected PLs.

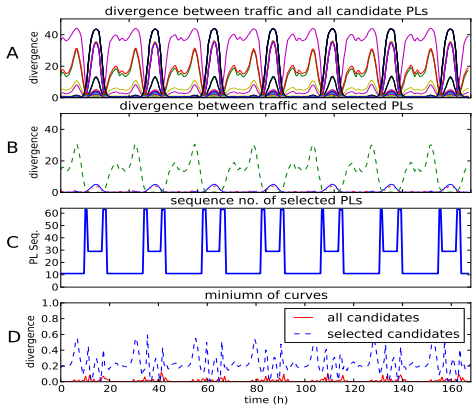

Fig. 9. Results of PL refinement for *the model-free* method in a network with *day-night pattern*. All figures share the x-axis. (A) and (B) plot the divergence of traffic in each window with all candidate PLs and with selected PLs, respectively. (C) shows the *active* PL for each window. (D) plots the *generalized divergence* of traffic in each window with all candidate PLs and selected PLs.

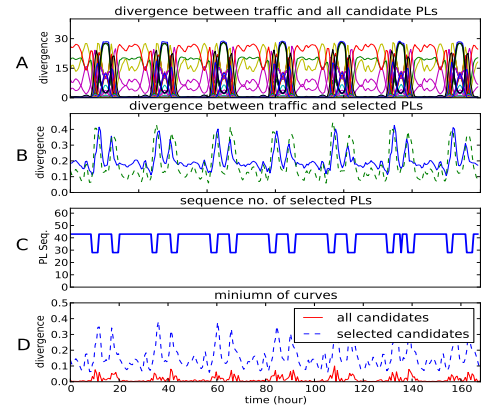

Fig. 10. Results of PL refinement for *the model-based* method in a network with *day-night pattern*. All figures share the x-axis. (A) and (B) plot the divergence of traffic in each window with all candidate PLs and with selected PLs, respectively. (C) shows the *active* PL for each window. (D) plots the *generalized divergence* of traffic in each window with all candidate PLs and selected PLs.

refined family of the *model-free* method corresponds to a "pattern of normal behavior," whereas, each PL in the refined family of the *model-based* method describes the transition among the "patterns". This information is useful not only for anomaly detection but also for understanding the normal traffic in dynamic networks.

C. Comparison with vanilla stochastic methods

As we have argued, anomaly detection of dynamic networks is very challenging. Because the properties of normal traffic are time-varying, it is much harder for algorithms to distinguish malicious traffic from normal traffic. The vanilla *model-free* and *model-based* methods ([4, 13]), which are the foundation of our robust methods, are designed for networks with stationary traffic. In the vanilla methods, all reference traffic $\mathcal{G}_{ref}$ is used to estimate a single PL.

For both types of normal patterns in Sec. IV-A and Sec. IV-B, we compared the performance of our robust *model-free* and *model-based* method with their vanilla counterparts in detecting anomalies. We used all methods to monitor the server *SRV* for one week (168 hours) under the two patterns.

We considered an anomaly in which node $CT2$ increases the mean flow size by 30% at 59h and the increase lasts for 80 minutes before the mean returns to its normal value. This type of anomaly is associated with a situation when some users try to download large files from the server, which can happen when the attacker tries to download sensitive information packed into a large file.

For all methods, the window size is $w_s = 2000s$ and the interval $h = 2000s$. The quantization parameters are equal to those in the procedure for analyzing the reference traffic $\mathcal{G}_{ref}$. The simulation results show that the robust *model-free* and *model-based* methods perform better than their vanilla counterparts for both types of normal traffic patterns (Fig. 13).

For the case when normal traffic exhibits a *shifting pattern*,

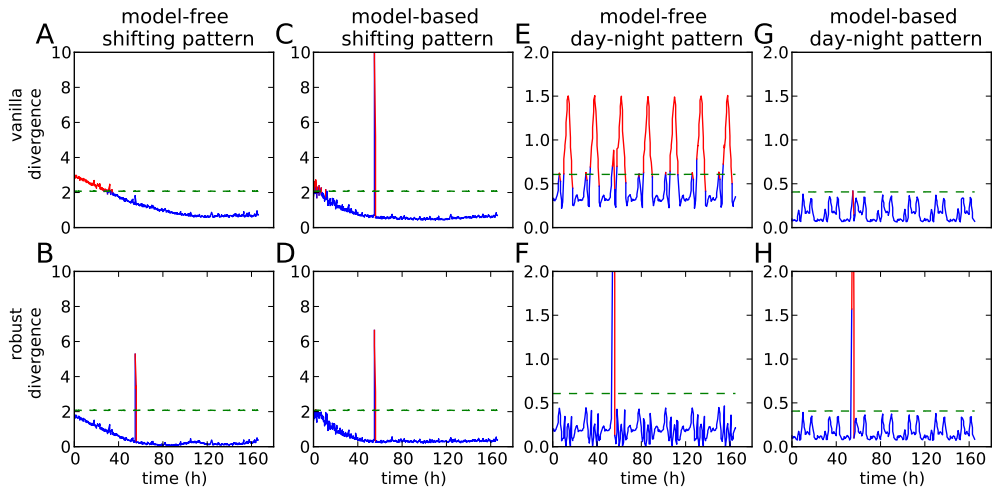

Fig. 13. Comparison of vanilla and robust methods. (A), (B), (C), (D) are for a network with a shifting pattern; (A), (B) show detection results of vanilla and robust *model-free* methods and (C), (D) show detection results of vanilla and robust *model-based* methods. (E), (F), (G), (H) correspond to a network with a day-night pattern; (E) (F) are detection results of vanilla and robust *model-free* methods and (G), (H) show detection results of vanilla and robust *model-based* methods. The horizontal lines indicate the detection threshold.

the detection threshold λ equals 2.0 for all methods. The vanilla *model-free* method misses the anomaly when the normal traffic shows a shifting pattern (Fig. 13A). To make it worse, it generates false alarms for the first 30 hours. In contrast, the robust *mode-free* method detects the anomaly successfully without false alarms (Fig. 13B). False alarms also appear in the detection results of the vanilla *model-based* method, though it detects the anomaly successfully (Fig. 13C). Again, the robust *model-based* method is superior as it reports no false alarm (Fig. 13D).

Compared with the *shifting pattern*, the *day-night pattern* has more influence on the results from the vanilla methods. For both the vanilla and the robust *model-free* methods, the detection threshold λ equals 0.6. The vanilla *model-free* method reports all night traffic (between 3 am to 11 am) as anomalies (Fig. 13E). The reason is that the night traffic is lighter than the day traffic, so the PL calculated using all of $\mathcal{G}_{ref}$ is dominated by the *day pattern*, whereas the *night pattern* is underrepresented. In contrast, because both the *day* and the *night pattern* is represented in the refined family of PLs (Fig. 9B), the robust *model-free* method is not influenced by the fluctuation of normal traffic and successfully detects the anomaly (Fig. 13F).

The *day-night pattern* has similar effects on the *model-based* methods. When the detection threshold λ equals 0.4, the anomaly is barely detectable using the vanilla *model-based* method (Fig. 13G). Similar to the vanilla *model-free* method, the divergence is higher during the *transitional time* because the *transition* pattern is underrepresented in the PL calculated using all of $\mathcal{G}_{ref}$. Again, the robust *model-based* method is superior because both the *transition pattern* and the *stationary pattern* are well represented in the refined family of PLs (Fig. 13H).

V. Conclusions

The statistical properties of normal traffic are time-varying for many networks that are dynamic. We propose a robust *model-free* and a robust *model-based* method to perform host-based anomaly detection in dynamic networks. Our methods can generate a more complete representation of the normal traffic and are robust to the non-stationarity in networks.

References

[1] M. Roesch *et al.*, "Snort-lightweight intrusion detection for networks," in *Proceedings of the 13th USENIX conference on System administration*. Seattle, Washington, 1999, pp. 229–238.

[2] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer networks*, vol. 31, no. 23, pp. 2435–2463, 1999.

[3] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*. ACM, 2002, pp. 71–82.

[4] I. C. Paschalidis and G. Smaragdakis, "Spatio-temporal network anomaly detection by assessing deviations of empirical measures," *IEEE/ACM Trans. Networking*, vol. 17, no. 3, pp. 685–697, 2009.

[5] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham *et al.*, "Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation," in *DARPA Information Survivability Conference and Exposition, 2000. DISCEX'00. Proceedings*, vol. 2. IEEE, 2000, pp. 12–26.

[6] A. Dembo and O. Zeitouni, *Large Deviations Techniques and Applications*, 2nd ed. NY: Springer-Verlag, 1998.

[7] K. Thompson, G. J. Miller, and R. Wilder, "Wide-area Internet traffic patterns and characteristics," *Network, IEEE*, vol. 11, no. 6, pp. 10–23, 1997.

[8] W. Hoeffding, "Asymptotically optimal tests for multinomial distributions," *Ann. Math. Statist.*, vol. 36, pp. 369–401, 1965.

[9] I. C. Paschalidis and D. Guo, "Robust and distributed stochastic localization in sensor networks: Theory and experimental results," *ACM Transactions on Sensor Networks*, vol. 5, no. 4, 2009.

[10] J. Wang, "SADIT: Systematic Anomaly Detection of Internet Traffic," http://people.bu.edu/wangjing/open-source/sadit/html/index.html, 2012.

[11] J. Sommers, R. Bowden, B. Eriksson, P. Barford, M. Roughan, and N. Duffield, "Efficient network-wide flow record generation," pp. 2363–2371, 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5935055

[12] A. Technologies, "The Net Usage Index by Industry," http://www.akamai.com/html/technology/nui/industry/index.html, 2013.

[13] R. Locke, J. Wang, and I. Paschalidis, "Anomaly detection techniques for data exfiltration attempts," Center for Information & Systems Engineering, Boston University, 8 Saint Mary's Street, Brookline, MA, Tech. Rep. 2012-JA-0001, June 2012.