Information Sciences 420 (2017) 313–328

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Anomaly detection using the correlational paraconsistent machine with digital signatures of network segment

Eduardo H.M. Pena[a,*], Luiz F. Carvalho[b], Sylvio Barbon Jr.[b], Joel J.P.C. Rodrigues[c,d,e,f], Mario Lemes Proença Jr.[b]

[a] Federal University of Technology Paraná, Toledo, Brazil
[b] Computer Science Department, State University of Londrina, Londrina, Brazil
[c] National Institute of Telecommunications (Inatel), Brazil
[d] Instituto de Telecomunicações, Universidade da Beira Interior, Portugal
[e] ITMO University, Russia
[f] University of Fortaleza (UNIFOR), Brazil

ARTICLE INFO

Article history:
Received 13 January 2015
Revised 18 August 2017
Accepted 23 August 2017
Available online 24 August 2017

Keywords:
Traffic characterization
Anomaly detection
Network management
Paraconsistent logic
ARIMA
ACO

ABSTRACT

This study presents the correlational paraconsistent machine (CPM), a tool for anomaly detection that incorporates unsupervised models for traffic characterization and principles of paraconsistency, to inspect irregularities at the network traffic flow level. The CPM is applied for the mathematical foundation of uncertainties that may arise when establishing normal network traffic behavior profiles, providing means to support the consistency of the information sources chosen for anomaly detection. The experimental results from a real traffic trace evaluation suggest that CPM responses could improve anomaly detection rates.

1. Introduction

Considered to be some of the most important resources in a modern environment, computer networks must provide means to satisfy demanding requirements. Moreover, along with their rapid growth comes a need for automating management functions to prevent network abuse and reduce the cost of ordinary operations. In this context, a network anomaly detection system is an important component of a security management infrastructure for computers and networks.

Network traffic anomalies have become a troubling issue for both network administrators and end users because they typically change the normal behavior of a network traffic in a malicious or unintentional manner, resulting in the congestion and depletion of available resources. Apart from reducing performance, abnormal activities may also interrupt the operation of services on a network, incurring substantial losses for universities, government agencies, and companies in general [25].

Many challenges restrict the widespread setting of anomaly detection techniques. For example, defining the normal behavior is very ambitious because it can evolve over time and domains [21]. Moreover, filtering techniques may not be able to remove only the actual noise from training data [40]. In addition, the complexity in the tuning, configuration, and de-

[*] Corresponding author.
E-mail addresses: eduardohmpena@gmail.com (E.H.M. Pena), luizfcarvalhoo@gmail.com (L.F. Carvalho), barbon@uel.br (S. Barbon Jr.), joeljr@ieee.org (J.J.P.C. Rodrigues), proenca@uel.br (M.L. Proença Jr.).

http://dx.doi.org/10.1016/j.ins.2017.08.074
0020-0255/© 2017 Elsevier Inc. All rights reserved.

ployment of available solutions may lead to rough requirements and constraints [39]. Another major difficulty is the current low detection efficiency of available systems, for example, high false-positive rates [7]. However, despite all these challenges, anomaly detection techniques have still been widely investigated because they consider several interesting research problems.

As practical measures that mitigate the trespassing of networks, anomaly detection techniques are typically divided into two main categories: signature- and anomaly-based [7]. Signature-based techniques rely on templates of well-known attacks to match and identify intrusions, require a regular update of their signature rules, and are not generally efficient against novel intrusions. In contrast, anomaly-based techniques can detect unknown attacks as a result of a strategy that analyzes deviations in the real traffic behavior from normal patterns. Anomaly-based techniques are generally related to a pair (M, λ), where M is the model of a normal network operation, and λ is a defined rule to estimate the deviation from M used to detect anomalous activities [7].

The nature of input data remains a common issue, regardless of the technical perspective supported by anomaly detection solutions [7]. In this context, the traffic flow analysis has drawn the interest of the research community because of the increasing support of flow tools from network equipment manufacturers [31]. Abnormal behavior detection can be based on the features extracted from different metrics of network traffic flows. Moreover, a proper correlation between them can be established to provide a better perspective of an event. These procedures have shown great potential in reducing unwanted notifications while also helping establish the underlying problem or condition that produces the anomalous events [47].

One of the foundations of our proposal for anomaly detection is the digital signature of the network segment flow analysis (DSNSF) used as normal traffic behavior profiles for servers or segments of the network [6,35]. In this study, DSNSFs are arranged under the rules of two models, namely the autoregressive integrated moving average (ARIMA) [34] and the ant colony optimization for digital signature (ACODS) [12]. These models have distinctive features. ARIMA is a traditional time series forecasting model, while ACODS is a metaheuristic typically used for optimization. Through the analysis of flow records, each of them can structure different DSNSFs used as a basic standard or level for selected traffic features. Aiming to assimilate DSNSFs from both models and common disturbance of traffic features caused by network-wide anomalies, we take advantage of the paraconsistent criteria to frame a tool for anomaly detection, called the correlational paraconsistent machine (CPM).

The CPM design is based on a nonclassical logic known as the paraconsistent logic (PL) [19], which is applied for the mathematical foundation and interpretation of the uncertainties associated with normal traffic behavior profiles and real measurement evaluation for anomaly classification. The interpretation of uncertainties is inspired by the usual expertise of network administrators that benefits from the historical knowledge of different parameters extracted from network segments to handle events harmful to the network infrastructure. It parallels the behavior of traffic features from DSNSFs and real-time measurements to assess evidence of the following proposition or hypothesis:

P_1: Presence of an anomaly in traffic at a time interval t.

These pieces of evidence determine the levels of certainty and contradiction of the presence of anomalies. If the real-time measurements are behaving properly, the levels of certainty and contradiction are the lowest. If not (i.e., the target of network anomalies), the levels of certainty are the highest, while the levels of contradiction are the lowest.

The proposed CPM accomplishes the following contributions: (a) it combines different computational intelligence approaches in a single cooperative solution; (b) it operates on aggregate traffic at network segments, which is an appealing answer for the current bandwidth transmission technologies; (c) it explores the potential of traffic volume and distribution measures together; and (d) it provides alternative reasoning metrics for the evaluation of the level of certainty for the presence of anomalies.

The rest of this paper is organized as follows: Section 2 presents the related work; Section 3 introduces some fundamentals on the paraconsistent logic for the evaluation of information signals; Section 4 describes the concepts, applications, and generation of DSNSFs; Section 5 discusses the proposed CPM design and application for anomaly detection; Section 6 shows the experimental evaluation results; and finally, Section 7 presents our conclusions and future directions.

2. Related work

Many papers have contributed to the network anomaly detection field by means of several approaches. The use of the subspace method was investigated by [29] for anomaly detection in traffic flow data. The method was applied in the flow time series of the traffic coming from randomly sampled data captured in routers from an academic Internet backbone. The subspace method designated a time interval, in which the traffic was considered anomalous. Through a manual inspection, the method then characterized network-wide anomalies, showing the wealth of information that can be extracted from network traffic flows. More recently, the methodology for applying the subspace method was reviewed in [11]. The study also discussed the limitations of existing solutions based on principal component analysis and investigated the use of statistical process control to overcome their main drawbacks (e.g., to best select the number of principal components and how to incorporate dynamics into the model).

To some extent, many approaches for anomaly detection rely on traffic–feature distributions and correlations [26,42]. Yu et al. [46] found that current DDoS attack flows are usually more similar to each other compared to the flows of flash crowds. Thus, the authors proposed a discrimination algorithm using the flow correlation coefficient as a metric to measure the similarity among suspicious flows and differentiate DDoS attacks from flash crowds. The feasibility of the proposed method was theoretically proven, and experiments confirmed its effectiveness. In this context, a hybrid probabilistic metric to detect flooding attacks was proposed by Li et al. [32], which can efficiently distinguish a DDoS attack from a flash crowd. The initial mechanics of these approaches usually involve selecting suitable features, estimating their distributions, and then defining proper correlations between them. It is important to note that different feature sets may be chosen according to the goal of the anomaly detector and the dynamic nature of network traffic. Our work particularly focuses on distinguishing between normal and anomalous traffic.

Unsupervised anomaly detection techniques have triggered interest in the academic community because of their ability to detect anomalies without using labeled training data [39,41] used common network traffic patterns as a baseline for the online detection of abnormal behavior. Meanwhile, [48] and [45] incorporated the autoregressive integrated moving average (ARIMA) in network security applications. Both studies relied on simulated data to simulate a real network environment and a synthetic anomaly generation. These studies explored the ARIMA capabilities to characterize future traffic rates based on a measured traffic history.

Although algorithms based on the ant colony are plentiful in several research fields related to data mining, their application on anomaly detection is limited [16]. A study of an ant-based clustering algorithm applied to detect anomalous events is presented in [36]. The authors demonstrated results comparable to those obtained using widely used methods, such as support vector machines and genetic algorithms. Meanwhile, Tsang and Kwong [44] used the principles of the ant colony theory to improve the clustering algorithms developed to extract high-dimensional traffic patterns.

A method for reducing false alarm rates in anomaly detection was presented in [23]. The authors presented the locally adaptive multivariate smoothing method to improve the classification of anomaly detectors based on diverse algorithms, such as abrupt change detection and fixed rules. The outputs from different detectors were combined according to the historical scores of similar events and the features of the handled detector models to produce better results.

Grana et al. [22] advocated that attackers are able to rapidly innovate for the sake of avoiding the signature schemes proposed by intrusion detection systems (IDSs). In light of this, behavioral methods were preferred to signature-based ones. The authors pointed out two approaches to address this issue. The first one was the specification improvement of the normal network conditions. The second approach was a model based on the comparison of the hypothetical behavior of a network exposed to risk and that of a normal network. In this way, the challenge was how to model the behavior of an attacked network without presupposing an attacker scheme because the penetration methods quickly evolve. The work presented by the authors proposed a likelihood ratio detector. The network was modeled as a directed graph, where each node was a host, user, or system within the network, and the edges were the communication channel between two nodes. Both nodes and edges had states associated with them. The network then evolved according to a Markov process over all the possible joint states of every node and edge. Even outperforming a simple anomaly detector, the proposed likelihood detector suffered from the noise and uncertainties generated by the low signal properties of the attacks.

Another statistical-based approach for anomaly detection was proposed by Bhuyan et al. [8]. This work applied an outlier-based anomaly detection that used generalized entropy and mutual information to create a feature selection capable of identifying a relevant nonredundant subset of features. According to the authors, the mutual information reduced the uncertainty of one random variable, and the generalized entropy measured the uncertainty in data. They made the detection faster and more accurate. Moreover, a tree-based clustering technique was proposed to generate a set of reference points and an outlier score function to rank incoming network traffic and identify anomalies. Although the approach achieved interesting results on anomaly detection, it was parameter tuning-dependent and had limited capabilities for categorical data types.

Linked to the complexity of the nature of the data, there is also the specificity of the model chosen for anomaly detection [7]. This combination can increase the degree of uncertainty of the classification of what is a harmful event and what is not. We use herein the paraconsistent logic (PL) [19], more specifically, the paraconsistent annotated logic with the annotation of two values (PAL2v), to improve the use of historical data and the evaluation of uncertainty events.

The paraconsistent annotated logics were devised by Blair and Subrahmanian [9] and applied to several fields, including artificial intelligence [1], digital circuits [2], and medicine [38].

An interesting model for pattern recognition capable of representing the concept of paraconsistency, called the discriminative paraconsistent machine (DPM), was introduced in [24]. Research partly similar to ours, and also relating to network security, can be seen in [28]. The authors handled uncertainty knowledge in classifying abnormal behavior patterns using a neutrosophic logic classifier to generate rules for intrusion classification. However, they used a supervised algorithm applied to a previously classified dataset. Our approach combines the digital signatures of network segment using flow analysis with the paraconsistent logic and offers a simple and efficient model to detect network anomalies.

3. Paraconsistent logic

Urged by the balance between computational performance and the precise formalization of the information carried out by models for traffic characterization, the paraconsistent logic (PL) is incorporated in a solution that enables assertions on the critical aspects of network traffic as well as being an appeal for validating the innate capacity of our methodology for network anomaly detection. In this section, we present the main paraconsistent aspects and measures used in our research and discuss why the PL could be used to devise a solution for network security.

3.1. Why use PL for anomaly detection?

Applications often face uncertainties and inconsistencies when required to characterize and analyze network traffic. Most of the time, the processed data may be incomplete or permeated with noise. In fact, acquiring information in an environment, which reflects ideal conditions, is rare [7]. Thus, the design of these data processing systems should result in solutions that deal with uncertainty knowledge in unfavorable situations. Such solutions should be able to gather, represent, manipulate, and interpret data considered imperfect. Next, some of the possible spots with inconsistencies for our problem setup are discussed.

Uncertainty in the evaluation of network-wide anomalies: The most critical goal of our research is to properly evaluate the network for the presence of traffic anomalies.

Uncertainty in data acquisition: We collect data from a real network and store them as a historical data resource. We do not apply any filtering technique because we are interested in their pure real behavior. Having anomaly-free stored data would be perfect, but we would be at a dead end expecting only normal data for training.

Uncertainty in the analyzed features: Some traffic features can be more reliable than others in general anomaly detection, but this may not always hold true for some events [29].

Uncertainty in the model chosen to generate the DSNSF: Many models can be chosen to define a normal behavior. Knowing which one is the best is hard. Hence, we have chosen those with a good mathematical basis. Regardless of the model chosen, using historical data to predict future ones will lead to uncertainties that the model alone is not prepared to deal with [39].

In our proposal, PL concepts are used to evaluate the uncertainty measures that arise when contrasting two sources of information. This allows our methodology to meet uncertainty, understanding through solid theoretical aspects to deal with conflicting information and provide a meaningful interpretation of uncertainty measures to support conclusions. Moreover, the PL concept is described as an extension of the paraconsistent logic to be used in our proposal. The paraconsistent annotated logic with the annotation of two values (PAL2v) provides fundamentals and terminologies that allow quantitative nonexact measures and qualitative evaluation to be computationally combined in an application.

3.2. Paraconsistent annotated logic with the annotation of two values (PAL2v)

The paraconsistent annotated logic with the annotation of two values (PAL2v) [19] enables equating pieces of evidence of a proposition P expressed by ordained pairs in the form $\{(\rho, \eta), \in [0, 1] \subset \mathbb{R}\}$. All pieces of evidence are fed from two different sources of information available for the PAL2v application. The first source of information usually produces a signal that represents favorable evidence for P, the "$\rho \rightarrow$ Level of favorable evidence." Similarly, the second source produces a signal that serves as unfavorable evidence for P, "$\eta \rightarrow$ Level of unfavorable evidence." Thus, the annotation $P(\rho, \eta)$ corresponds to the statement "The favorable evidence of a proposition P is ρ, whereas the unfavorable evidence of P is η."

Intuitively, a defined correlation can be established when the first source of information shows a high level of favorable evidence (ρ), and the second source shows a low level of unfavorable evidence (η). Another correlation can be set when the favorable source presents the lowest ρ, and the unfavorable source presents the highest η. The idea of inconsistency comes when both sources present similar levels of evidence; that is, they do not hold enough information to support the acceptance or denial of P.

Paraconsistent measures can be estimated by considering two sources of information according to evidential levels. Thus, the favorable and unfavorable pieces of evidence are related and express the level of certainty (L_c) and contradiction (L_{ct}) as in Eqs. (1) and (2), respectively:

$$L_c = \rho - \eta \tag{1}$$

$$L_{ct} = \rho + \eta - 1 \tag{2}$$

The level of certainty is located at the horizontal axis of a PAL2v lattice (Fig. 1), while the level of contradiction is located at the vertical axis. These levels belong to a set of real numbers $\mathbb{R}$ ranging from -1 to $+1$, and are used to quantify The levels of inconsistency for a paraconsistent analysis.

When associated with a PAL2v lattice, the levels of certainty and contradiction define the states that can be used for decision making. The theoretical foundation of PL and PAL2v seen in [18,19] was followed to define the output states used in this research. More details on the system of the coordinates of a paraconsistent axis, such as change in scale, rotation, and translation, can be seen in [18]. First, the levels of certainty and contradiction are used to determine logical states, which help verify the consistency of a PAL2v process. Table 1 delineates the set of eight output states, and Fig. 1 illustrates their position on the PAL2v lattice.

The evaluation of uncertainties considers all the pieces of evidence to obtain a level of certainty. These pieces of evidence may be incomplete or inconsistent; hence, they must be strengthened to the point, where the level of certainty reaches a maximum value appropriate for a concise affirmation of a proposition. A representation of an interval in values of certainty, where the level of certainty (L_c) may range without being limited by the level of contradiction (L_{ct}), is obtained through the interval of certainty (φ) in Eq. 3.

$$\varphi = 1 - |L_{ct}| \tag{3}$$

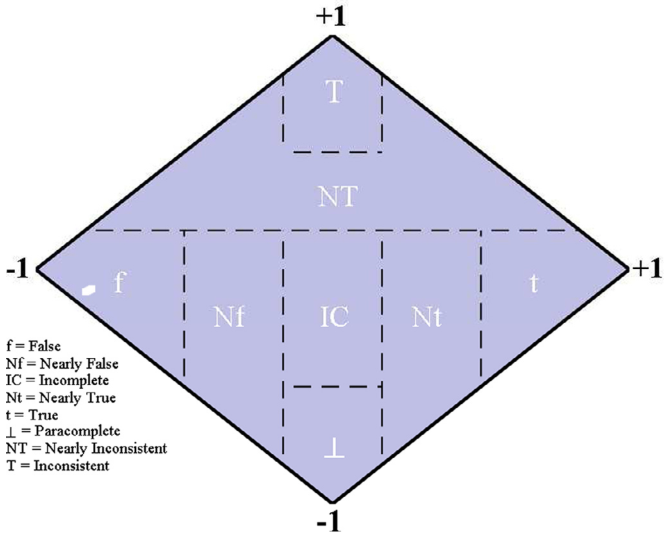


Fig. 1. PAL2v lattice represented by a diamond on the Cartesian plane. The coefficients of certainty and contradiction (L_c, L_{ct}) translate a hypothesis in a paraconsistent domain, thereby allowing one to interpret the resulting output state.

Table 1
Paraconsistent logical states.

Logical state	L_c	L_{ct}
False (*f*)	$\leq -\frac{1}{2}$	$\geq -\frac{1}{2}$ and $< \frac{1}{6}$
Nearly false (*Nf*)	$\geq -\frac{1}{2}$ and $< -\frac{1}{6}$	$\geq -\frac{5}{6}$ and $< \frac{1}{6}$
Incomplete (*IC*)	$\geq -\frac{1}{6}$ and $< \frac{1}{6}$	$\geq -\frac{1}{2}$ and $< \frac{1}{6}$
Nearly true (*Nt*)	$\geq \frac{1}{6}$ and $\leq \frac{1}{2}$	$\geq -\frac{5}{6}$ and $< \frac{1}{6}$
True (*t*)	$> \frac{1}{2}$	$\geq -\frac{1}{2}$ & $< \frac{1}{6}$
Paracomplete (⊥)	$\geq -\frac{1}{6}$ $<$ and $\frac{1}{6}$	$< -\frac{1}{2}$
Inconsistent (⊤)	$\geq -\frac{1}{6}$ and $< \frac{1}{6}$	$\geq \frac{1}{2}$
Nearly inconsistent (*N*⊤)	None of the above	None of the above

The value of the level of certainty to be considered apart from the effect caused by contradictions is called the level of real certainty (L_{cr}) and calculated from the level of certainty obtained by the analysis of the lattice of PAL2v. First, the distance between the extreme logical points *t* and *f* and *d* is estimated in Eq. (4).

$$d = \sqrt{(1 - |L_c|)^2 + {L_{ct}}^2} \tag{4}$$

The level of a real certainty value L_{cr} is obtained using Eq. (5):

$$L_{cr} = \begin{cases} 1 - d & \text{if } L_c > 0 \\ d - 1 & \text{if } L_c < 0 \\ 0 & \text{if } L_c = 0 \end{cases} \tag{5}$$

The level of real evidence of the output can be calculated by Eq. (6):

$$\mu_{er} = \frac{L_{cr} + 1}{2} \tag{6}$$

The intensity of the level of real evidence (μ_{er}) in paraconsistent analysis systems is used as the main value for decision making [18], signaling the intensity of certainty for a proposition *P*.

Section 4 presents the evidential information source used in our paraconsistent analysis, which is a network traffic behavior profile, called the digital signature of network segment using flow analysis. Meanwhile, Section 5 describes the procedures applied to extract the pieces of paraconsistent evidence in our methodology. These procedures are based on how empirical knowledge of the normal network behavior is acquired by the network manager and how he or she can identify the subtleties in the network movement for each weekday.

4. Digital signature of network segment using flow analysis

The digital signature of the network segment using flow analysis (DSNSF) is a network traffic behavior profile based on learning the normal routine consumption of network resources [35]. DSNSF summarizes the usual expectations for data flowing throughout the day in a network segment, such as performance metrics, protocol types, services, and traffic distribution. Traffic characterization is a fundamental aspect of our methodology for anomaly detection. Hence, a DSNSF should be able to efficiently forecast the traffic characteristics of segments constituting a network backbone in a desired time window. The use of DSNSFs assists network managers in identifying limitations and controlling the resource consumption of services along the backbone, thereby avoiding problems of performance and faults [5].

In this research, each DSNSF is built to forecast traffic profiles for weekdays. The model responsible for calculating the DSNSF performs a historical analysis of the traffic features and draws baseline curves for them. These traffic features are derived from the flow characteristics of the monitored network within a preprocessing process.

4.1. Data preparation

The arrangement of a DSNSF generally uses a model that analyzes a set of collected values for a desired period of the day, that is, each day of the week, thereby preserving the characteristics of the traffic based on the time variations over the day. The collected values refer to features coming from network traffic flow measurements stored over the weeks preceding the day monitored.

In this study, we have limited the set of features to bits, packets, entropy for destination IP address (H(dstIP)), entropy for source IP address (H(srcIP)), entropy for destination port (H(dstP)), and entropy for source port (H(srcP)). Each feature is represented by a quantitative value. We assume the mean values of each period for volume information and summarize their degree of concentration or dispersal using entropy for the IP addresses and port numbers.

The entropy metric can be estimated as follows: a histogram for each metric can be represented as $A = \{n_1, \dots, n_i, \dots, n_N\}$, where each element i occurs n_i times, and N is the total number of distinct observations in the histogram for each given period. The probability distribution of element i is defined in Eq. (7):

$$p_i = \frac{n_i}{N} \tag{7}$$

The entropy can then be defined as follows with Eq. (8):

$$H(A) = -\sum_{i=1}^{N} p_i \log_2 p_i \tag{8}$$

The entropy value is minimal when the sample distributions are concentrated, and zero when all samples are the same. Moreover, the degree of dispersion increases when closer to $\log_2 N$.

A wide range of traffic details can be reached through IP flow analysis, especially after an accurate profile of normal traffic behavior is established. Network traffic flows can help detect and classify anomalies because significant changes in the normal distribution and volume of their observed metrics may represent an anomalous event.

4.2. DSNSF generation using ARIMA

The autoregressive integrated moving average (ARIMA) is a time series forecasting model that captures the linear dependence of the future over past values [33]. This model is capable of characterizing future traffic trends based on the measured traffic history. This future trend and the real-time network traffic are then compared to detect divergences as an indicative of anomalies.

An ARIMA model includes three parameters: autoregressive parameter (p), number of differencing steps (d), and moving average parameter (q), and can be summarized as $ARIMA(p; d; q)$ and generalized as in Eq. (9):

$$z_t = \sum_{i=1}^{p} \phi_i z_{t-i} - \sum_{j=1}^{q} \theta_j \alpha_{t-j} \tag{9}$$

where sequence $\alpha_{t-1}, \alpha_{t-2}, \dots, \alpha_{t-q}$ represents the white noise process, and ϕ and θ are the autoregressive and moving average coefficients of finite order p, q, respectively. The original time series is differentiated for d times to obtain z_t, which is the integrated part of the model.

In practice, most time series are nonstationary. Thus, stationary autoregressive (AR) or moving average (MA) processes cannot be directly applied [33]. One possible way of handling nonstationary series is to apply differencing. This process allows us to reach a more general model, that is, the autoregressive integrated moving average (ARIMA) model. The first issue $z_t = x_t - x_{t-1}$ or $z_t = (1 - B)x_t$ can be evaluated again to give second differences and so on. The d_{th} differences may be written as $(1 - B)^d x_t$. We have an $ARIMA(p, d, q)$, where d denotes the number of differences taken, if the data series is differenced d times before fitting an autoregressive moving average $ARMA(p, q)$ model.

Searching for a model that is a mean approximation to the data is customary, which means using as few parameters as possible. The main difficulty when fitting AR and MA models is determining the order of the process rather than estimating the coefficients. Some standard techniques are available, but for practical purposes, many series analysts simply differ the series until the autocorrelation closes to zero as quickly as possible [33].

The creation of DSNSF from the ARIMA model dynamically occurs using data from the past weeks for training and the changes in every new day processed to recalibrate the model. The first step is to estimate parameter d. The differenced parameter can be evaluated using a standard estimation method [33] and tested by the autocorrelation function. A technique called grid search [48] is used to estimate p and q parameters. The Akaike for information criterion (AIC) is used to achieve a good balance between model parsimony and low prediction error [33]. The grid search method does a thorough search for all possible combinations for p and q toward obtaining the minimum AIC. Finally, the identified and estimated parameters are used to build a model to predict values for future readings in the form of a DSNSF [34].

4.3. DSNSF generation using ACODS

The ant colony optimization (ACO) metaheuristic consists of a set of computational methods and strategies based on the observations of behaviors of real ant colonies. This approach aims to simulate ant behavior to find the shortest route between a food source and their nest. Most researches on ACO are of an experimental nature because of this characteristic [14], as has also been reflected by the content of most of the papers published in the literature.

The ACO metaheuristic uses a population of concurrent and globally asynchronous agents to find solutions to complex optimization problems. These agents are guided by pheromone trails that are intensified as promising solutions are created, leading to the convergence of the entire colony. Although each ant has the capacity to build a feasible solution as a real ant can somehow find a path between the nest and the food, highest-quality solutions are achieved through a cooperation of the individuals of the entire colony.

ACO has been outperforming several algorithms in many study areas. In communication networks, for instance, several studies can be found on fault localization [20], efficient energy management in wireless sensor networks [37], and route discovery and network reconfiguration [3]. This could be accomplished because of many aspects, such as distributed collaboration of agents (ants), self-organizing, and positive feedback, that lead to a rapid discovery of good solutions.

In this study, an ACO-based algorithm called ant colony optimization for digital signature (ACODS) is used to create a normal profile of traffic behavior. ACODS creates the DSNSF using a clustering approach, which is capable of characterizing network traffic through the discovery of a cluster set from a large volume of high-dimensional input data. According to Jiang et al. [27], the ants' habit of living in groups is essentially similar to the grouping of data. Algorithms based on ant behavior have natural advantages in the application of cluster analysis, such as self-organization, flexibility, robustness, absence of need for prior information, and decentralization. Moreover, the application of clustering algorithms can benefit from the aspects of the instinct of ants searching for the optimal path because a measure indicating the progress of solutions created for groups is established. We use the objective function in Eq. (10) for this purpose.

$$J = \sum_{i=1}^{E} \sum_{j=1}^{K} \sqrt{\sum_{a=1}^{A} (x_{ia} - c_{ja})^2} \tag{10}$$

Each element i is composed of a sextuple of analyzed attributes and will be grouped to the best cluster j, in which $j = 1, \ldots, K$. Variable E represents the quantity of flows to be clustered, while A indicates data dimensionality, that is, the number of flow features to be processed. The collected elements are divided in 1 min intervals with a total of 1440 datasets throughout the day. Variable x_{ia} denotes the values of feature a of flow i, while c_{ja} stores the cluster center value j at dimension a. From right to left, the first sum is related to the Euclidean distances between x_{ia} and c_{ja}. The second and third sums apply the aforementioned comparisons between all K cluster centroids and E elements. The output value J corresponds to the objective function of our clustering approach, which should be minimized.

The search space is a graph $\mathcal{G}(\mathcal{V}, \mathcal{B})$, where V is a finite set of all nodes, and B is the set of edges. We assume that the nodes represent the elements to be clustered, while edges connect each of them to the centroids of the groups. Each group has its centroid, which is a new graph node located in the middle of the group. The centroid represents all the other cluster elements that it belongs to. The average among these centroids provides the value of the DSNSF for the six traffic features.

Ants traverse the search space following stochastic criteria, being attracted to locations most favorable to the minimization objective function. Ant decisions are taken based on a combination of a long-term learning process and an instantaneous heuristic prediction. Therefore, the sequence of random decisions is inherent in the algorithm [14]. At runtime, the ACO algorithm updates the pheromone values using previously generated solutions. The update aims to concentrate the search on regions of the search space containing high-quality solutions. In particular, the reinforcement of solution components depending on the solution quality is an important aspect of ant-based algorithms. The reinforcement implicitly assumes that good decisions are taken toward good solutions. At every association of an element to a cluster, the agent alters the whole local information by updating the pheromone trail shown in Eq. (11).

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{l=1}^{L} \Delta\tau_{ij}^{l} \tag{11}$$

The trail values can be incremented (when ants deposit pheromones on the edge or connections between the used nodes) or decreased. An increased pheromone concentration is an essential factor in the algorithm implementation because it directs ants to seek new locations more prone to acquiring a near-optimal solution. For the pheromone update equation, a constant ρ is defined, which describes the pheromone evaporation rate with value $0 < \rho < 1$. From a practical point of view, pheromone evaporation is applied to avoid a very rapid convergence of the algorithm toward a suboptimal region.

This evaporation implements a useful form of forgetting, thereby favoring the exploration of new areas in the search space. Variable t identifies the iteration running. Meanwhile, constant L is given by the number of solutions that will be chosen as the best, and only these will receive an increase in their tracks with pheromones. $\Delta\tau_{ij}^{l}$ is calculated by taking the inverse of J divided by the number of elements to be clustered, E.

As most metaheuristics, ACO may take many iterations to converge. The algorithm runs iteratively, that is, explicit criteria stops must be used during its execution. In this manner, running the ACODS can be terminated in two ways. The first one concerns the quality of the solutions generated. The iterative process is interrupted, as the ant colony converges to create a single solution, when no difference exists among all solutions built on consecutive iterations. The second method of termination occurs when the limit of 100 iterations is reached, preventing ACODS from running indefinitely. In practice, no more than 13 iterations were necessary for the convergence of the solutions used in the generation of the DSNSFs presented in Section 6.

4.4. Confidence bands

According to [10], an efficient approach for anomaly detection is the use of confidence bands or thresholds, which indicate an interval where variations are considered normal. We use this approach in comparison with our CPM proposal. The formulation of threshold ranges is founded on the adaptive threshold algorithm, which ensures accounting for traffic trends and variations based on past traffic measurements.

The thresholds are defined as $T_{up} = (\alpha + 1)\overline{\mu}_{n-1}$ and $T_{down} = (\alpha - 1)\overline{\mu}_{n-1}$. Parameter α specifies the acceptable range above the mean value that is an indication of anomalous behavior. $\overline{\mu}_{n-1}$ is the mean estimated from the measurements prior to n. As presented by [43], the mean $\overline{\mu}_n$ is computed using the exponential weighted moving average (EWMA) of previous time windows, as defined by Eq. (12):

$$\overline{\mu}_n = \beta\overline{\mu}_{n-1} + (1 - \beta)x_n, \tag{12}$$

where x is the DSNSF, and β is the EWMA factor. After numerical tests and based on the literature, the parameters considered for the adaptive threshold algorithm were $\alpha = 0.2$ and $\beta = 0.7$ [10]. Our tests show that other values for these tuning parameters resulted in a worse detection rate or did not perform better.

5. Correlational paraconsistent machine

This section describes the procedures used to integrate annotated logic and traffic characterization in a solution, called the CPM. The CPM uses DSNSFs as evidential information and PAL2v for the mathematical formalization of uncertainties. Fig. 2 shows an overview of the CPM operation.

Although the ultimate goal of the CPM is to evaluate the presence of anomalies in traffic at a given time, its design is devised to obtain a binary classifier. Moreover, using the CPM, we can represent and manipulate data that may be imperfect while meeting theoretical criteria from the paraconsistent nature. The quantitative evaluation through the PL representative lattice and the levels of real evidence provide insights into the consistency of the network traffic behavior profiles and the dataset quality. Therefore, the CPM works not only as an aggregation strategy (e.g., weighted average), but also as a decision-making tool that provides a convenient treatment to possible contradictory signals.

5.1. CPM Design

First, the DSNSFs and the real measurements are normalized to scale the range of their data in [0, 1]. Let $F \rightarrow$ be the “number of features” and $T \rightarrow$ be the “total time intervals.” A general DSNSF can be represented with a $F \times T$ matrix $\hat{\mathbf{X}}$, as for the real measurements with a $F \times T$ matrix $\hat{\mathbf{RM}}$. The DSNSF is a template or the expected behavior for network traffic. Further real measurements are from its related DSNSF farther from the expectations. This statement offers our first reasoning evidence, which is mathematically translated with Euclidean distances between $\hat{\mathbf{X}}$ and $\hat{\mathbf{RM}}$ in $\mathbf{D}$, as in Eq. (13).

$$\mathbf{D} = \left|(\hat{\mathbf{X}}, \hat{\mathbf{RM}})\right| \tag{13}$$

The variations in the normal behavior of traffic features may be caused by different anomalies. The CPM for this research correlates the features related to DoS, DDoS, and flash crowd. Table 2 relates these three anomalies and the features they typically distort [13,30]. Each type of anomaly is related to a correlational factor $\boldsymbol{\kappa}$ set with zero for features that are not affected and one for those that are affected. For example, a DDoS usually unsettles packets per second, distribution of destination IP address, and source and destination port numbers. Considering the order in which features are analyzed in our methodology (bits per second, packets per second, distribution of destination and source IP address, and destination and source port numbers), the resulting correlational factor is $\boldsymbol{\kappa} = \{0, 1, 1, 0, 1, 1\}$.

The CPM generates three outputs for each $\boldsymbol{\kappa}$, that is, a signature of the anomaly of interest. The correlational factor is used to multiply each row of $\mathbf{D}$ (Eq. (14)).

$$\mathbf{E} = \boldsymbol{\kappa}^{\intercal}\mathbf{D} \tag{14}$$

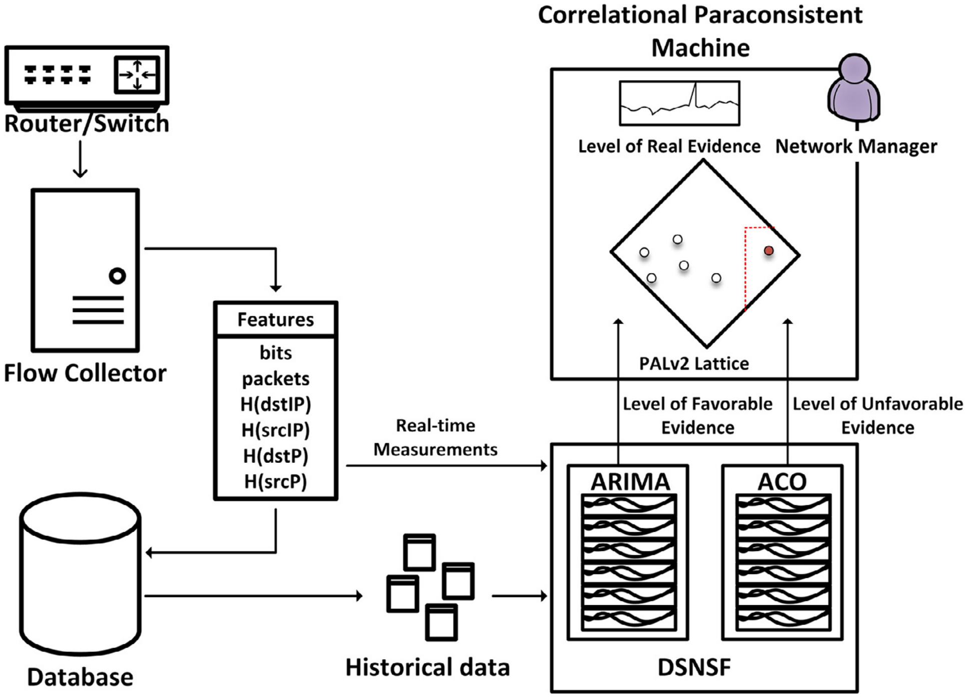


Fig. 2. Overview of the system: operating flow for the CPM.

Table 2
Traffic features.

Feature	Anomalies		
	Flash crowd	DoS	DDoS
Bits	✓	–	–
Packets	✓	✓	✓
H(dstIP)	✓	✓	✓
H(srcIP)	–	✓	–
H(dstP)	✓	✓	✓
H(srcP)	✓	✓	✓
κ	**{1, 1, 1, 0, 1, 1}**	**{0, 1, 1, 1, 1, 1}**	**{0, 1, 1, 0, 1, 1}**

Evidential terms ($\boldsymbol{\Psi}$) are then given for each $t \in T$ with a weighted average of the columns of $\mathbf{E}$ in relation to κ (Eq. (15)).

$$\boldsymbol{\Psi}_t = \frac{\sum_{j=1}^{F} \mathbf{E}_{(j,t)}}{\sum_{j=1}^{F} \kappa_j}, \quad t = 1, 2, 3, \ldots T \tag{15}$$

Recalling PAL2v concepts, we should be able to contrast the character from distinct sources of information. Therefore, the described models for the DSNSF generation, ARIMA, and ACODS are used because these information sources obtain a pair of DSNSFs $\hat{\mathbf{X}}'$ and $\hat{\mathbf{X}}''$, for each weekday:

$\hat{\mathbf{X}}'$ → DSNSF generated with the ARIMA model;

$\hat{\mathbf{X}}''$ → DSNSF generated with the ACODS model.

$\hat{\mathbf{X}}'$ and $\hat{\mathbf{X}}''$ are then used to evaluate the nominal pieces of evidence $\boldsymbol{\Psi}'$ and $\boldsymbol{\Psi}''$ using Eqs. (13), (14), and (15). Eq. (16) is used for the case of ARIMA ($\hat{\mathbf{X}}'$).

$$P = \boldsymbol{\Psi}' \tag{16}$$

As for ACODS ($\hat{\mathbf{X}}''$), Eq. (17).

$$\Lambda = |\boldsymbol{\Psi}'' - 1| \tag{17}$$

The operation in Eq. (17) is performed to transform the positive evidential levels coming from an information source into negative evidential levels because an information source rejects a certain proposition in inverse proportion to its assertion in terms of evidential levels. The CPM benefits from this feature by using different perspectives to draw conclusions. The interpolation between P and Λ follows a linear trend when a single source of information is used to extract both favorable and unfavorable pieces of evidence. Otherwise, the biases obtained by a single source are mitigated.

The $P = \{\rho_1, \rho_1, \ldots, \rho_T\}$, and $\Lambda = \{\lambda_1, \lambda_1, \ldots, \lambda_T\}$ sets are used as the favorable and unfavorable pieces of evidence, respectively, which are the main measures required for a PAL2v application. Algorithm 1 presents the procedures used to extract evidential levels for use in the CPM.

Algorithm 1 Procedures to obtain PAL2v evidence levels.

```
Use ARIMA to generate $\hat{\mathbf{X}}'$.
Use ACODS to generate $\hat{\mathbf{X}}''$.
for all $\kappa$ do
    Use $\hat{\mathbf{X}}'$ and Equations 13, 14, and 15 to obtain the nominal evidence set $\Psi'$.
    Use $\hat{\mathbf{X}}''$ and Equations 13, 14, and 15 to obtain the nominal evidence set $\Psi''$.
    Use $\Psi'$ and Equation 16 to obtain the favorable evidence set $P$.
    Use $\Psi''$ and Equation 17 to obtain the unfavorable evidence set $\Lambda$.
end for
```

5.2. Complexity analysis

Three key components are noted in our approach: data preparation, DSNSF generation, and classification.

The data preparation is achieved by applying Shannon entropy for the qualitative flow information. This calculus varies from 1 to the number of different entries (e) in the analyzed time interval and results in $\mathcal{O}(e^2)$.

The DSNSF generation consists of a training phase for ARIMA and ACODS. The computational complexity for ARIMA is related to the identification of its parameters and model estimation with general complexity in $\mathcal{O}(n^2)$, where n is the number of analyzed intervals [15]. The complexity of the ACODS algorithm is first given by partitioning a set of n intervals by the k centers of the f features, thereby resulting in $\mathcal{O}(nkf)$. Using the population of m ants to assist the search for the best centers for the collation of data, a quadratic complexity factor is added, thereby ensuring $\mathcal{O}(nkfm^2)$.

The classification process requires a normalization step that is in $\mathcal{O}(n)$. The remaining calculations are constant in time. The system analyzes a specific network segment for six different features (f). Thus, all the calculations needed by the traffic characterization process must be executed f times. The final complexity of our system is $\mathcal{O}(fne^2 + fn^2 + nkfm^2 + fn)$ if we considered a day with n intervals to be investigated.

All algorithms are implemented in C++ and MATLAB. The experiments run on a machine with Intel Core i7 CPU (2.60GHz) and 8 GB of main memory. In this test environment, the calculation performed for all the three components of our approach was finished in less than 6 s for a whole day, demonstrating the feasibility of the presented system in networks with high traffic aggregation.

6. Experimental evaluation

We validate our proposal and demonstrate the CPM functionality in this section. This analysis aims to verify the improvement in the anomaly detection rates obtained using the CPM fed with the ARIMA and ACODS models and how the correlation between the paraconsistent criteria can be useful for decision-making processes.

Meanwhile, we use the receiver operating characteristic (ROC) curve for the performance evaluation. The ROC is one of the most traditional methods applied to describe the true-positive detection rates of classification systems according to the changes in the false-positive rates. Versions of ARIMA and ACODS based on the confidence bands were implemented for comparison.

6.1. Traffic traces and anomalies

In the experiments, we used a real dataset provided by the State University of Londrina (Brazil) network, which has been well investigated in previous works dealing with network management [6] and anomaly detection [4,17]. In view of the requirement for long periods of training data, our solution should be considered for environments, where maintaining both historical and ongoing data is possible. An application, called Softflowd network analyzer, was used to export flows to the collector in standard sFlow version 5 sampled format. These exported flows were stored in binary files, which were processed by nfdump tools, resulting in the dataset used by the preprocessor. We used data from four weeks (from September 24, to October 19, 2012) to build the training data and three weeks (from October 22, to November 9, 2012) for the tests.

In addition to the inherent anomalies, the simulation of anomalies directly led into the testing data. A tool called Scorpius was used for this purpose [4]. Scorpius uses actual flow files as entry and synthesizes new ones with flows containing anomalous features in specified time intervals. The full description of the tool and more examples of its use can be seen in [4].

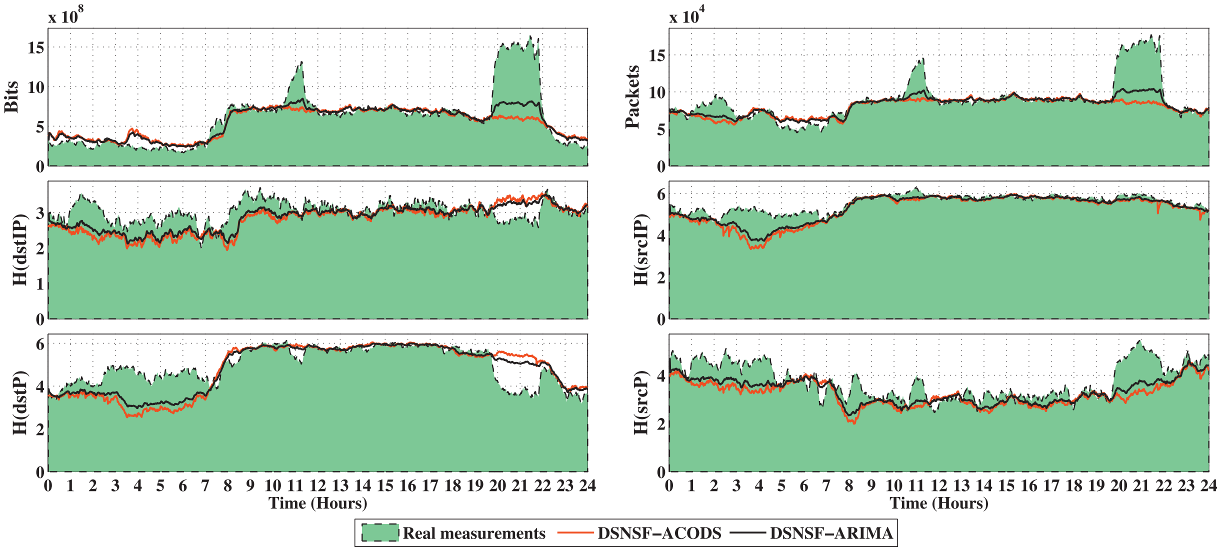


Fig. 3. Real measurements and DSNSFs.

Most of the traffic data were normal. Hence, we separated 20% of them to apply the synthesis of anomalies and fairly evaluate the performance of our proposed system. A set of fictitious IP addresses and port numbers was used to configure the DoS and DDoS anomalies with different intensities. We used the separated flows as input for the Scorpius tool, and injected a specific anomalous behavior to test the performance of the presented system.

A DoS exhaustively transmits UDP packets to a single IP address of destination from a specific origin and destination ports. In other words, a single source floods a well-known host with useless traffic. We simulated this behavior using the fictitious source IP address 11.11.11.11 through port 80 attacking the fictitious IP address 22.22.22.22 through port 8081. A DDoS usually features the incoming traffic flooding from many different sources. To synthesize its behavior, we used different fictitious source IP addresses through port 80 attacking the fictitious IP address of destination 22.22.22.22 through the 8081 port.

6.2. Experimental results

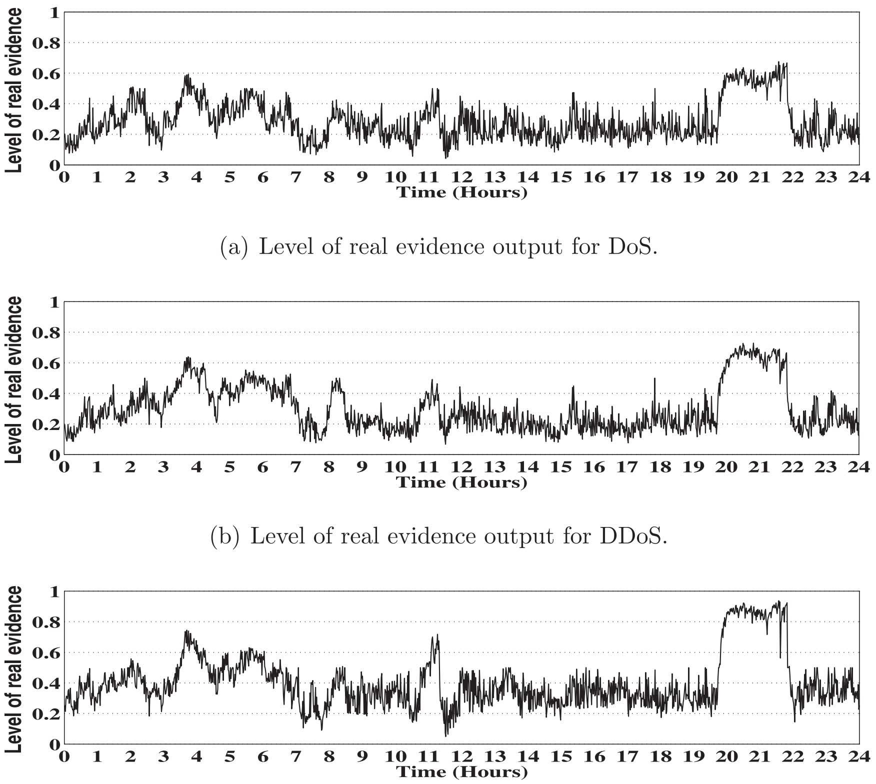
We provide the reader with an example of the CPM application, including the structuring of the sources of information chosen in this research, evidential level exposure, and anomaly identification.

First, we built DSNSFs for each day according to the ARIMA and ACODS specifications. Fig. 3 shows the behavior of real measurements and DSNSFs in the State University of Londrina network. The environment is set for October 30, 2012. The readings for each feature appear in green areas. The solid red lines represent the DSNSFs generated with ACODS, while the solid black line shows the ones generated with ARIMA. Verifying the adjustment between the real measurements and the traffic profiles found in the DSNSF is generally possible, which is a visual indication of the efficiency of the models proposed to characterize the normal traffic behavior. Noting that the changes are more pronounced in the early hours is also possible mainly because of the lack of movement and backup activities in the university network during these hours. Furthermore, the most accessed IP addresses pertained to one of the web servers of the university. At that time, a public tender was being held, and its result was scheduled to be disclosed on the aforementioned day. The increasing number of accesses to the server drastically affected the behavior of the traffic in the form of a flash crowd at around 8:00 p.m.

The used models must achieve effective results on traffic characterization because The profile-based system performance heavily depends on it. A series of efficiency tests for ARIMA and ACODS in traffic characterization are reported in [34] and [12], respectively.

The CPM can extract evidential levels by using the DSNSFs and the real measurements as sources of information, and present in its outputs the values of the real evidence levels for each kind of anomaly, in which it was set for. Figs. 4(a), 4(b), and 4(c) show the real evidence levels obtained on October 30, for DoS, DDoS, and flash crowd, respectively. From these values, the CPM provides a continuous verification of the levels of certainty and contradiction relating to the presence of anomalies.

The three outputs in Fig. 4 demonstrate similar behaviors at specific periods. Their levels are highest in the early hours as well as around 11:00 a.m. and 8:00 p.m. Small fluctuations are seen throughout the day. The similarity between these levels is caused by the correlation in the traffic features used in this research. In addition, a more fine-grained quantification of these levels could be achieved by inspecting further features. As discussed earlier, many works used the ranking and



(a) Level of real evidence output for DoS.

(b) Level of real evidence output for DDoS.

(c) Level of real evidence output for a flash crowd.

Fig. 4. Level of real evidence output from the CPM for DOS: October 30, 2012.

selection of traffic features with emphasis on different aspects, from packet arrival patterns to content-based characteristics. The CPM balances the correlation between the chosen features to determine the source and the type of anomaly.

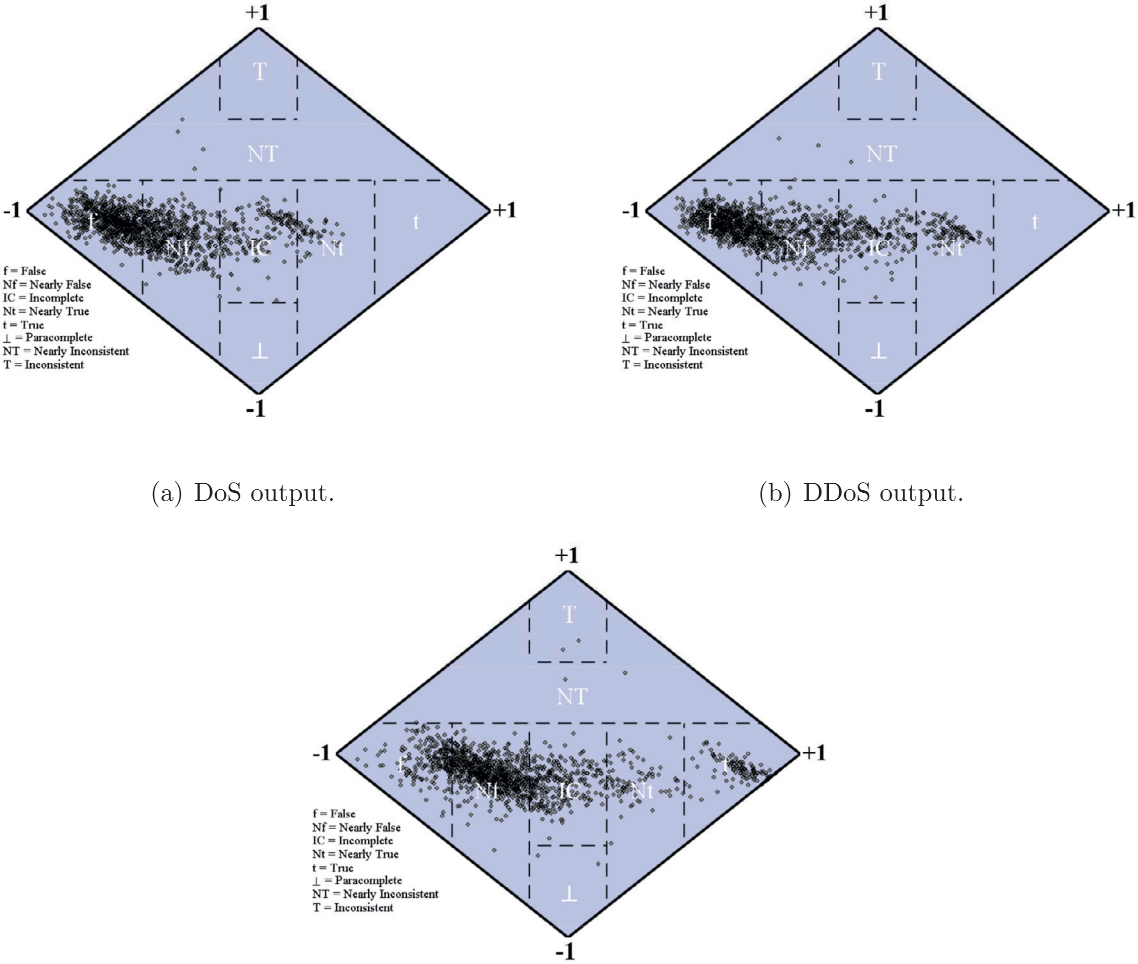
Fig. 4(c) illustrates that the real evidence levels reach a maximum value at around 8:00 p.m., which is a first indicative of a flash crowd. This result came from the evidential levels being strengthened with the lowest contradiction. Importantly, note that these measures were not used for classification. At this time, the main interest was to visualize particular regions pointing out suspicious events, such that one can more deeply examine their causes. For their part, the coordinates given by the levels of certainty and contradiction in the paraconsistent plane were used to signalize anomalies and trigger alarms to the network administrator. They were also used for the performance analysis.

Fig. 5 shows the evidential inputs in the paraconsistent plane obtained with the CPM for October 30. The coordinates of points (L_c, L_{ct}) in Fig. 5(a) present the DoS classification for each time interval. The classifications depicted in Figs. 5(b) and 5(c) are related to DDoS and flash crowd, respectively. The points in the true (t) state were considered as anomalies that happen when the favorable evidence is close to one, and the unfavorable evidence is close to zero, thereby representing a correlation between the responses of the two analyzed models. In these situations, the evidence coming from the ARIMA and the ACODS illustrated considerable distances between the DSNSF attributes and the real-time measurements. The operation done with the nominal pieces of evidence coming from the ACODS (Eq. (17)) ensured the level of uncertainty to be the lowest when a feature does not behave properly.

In practice, the majority of intervals is located at false (f), nearly false(Nf), or incomplete (IC), indicating the absence of anomalies. The states false (f) and nearly false (Nf) denote a situation, where the favorable evidence is close to zero, and the unfavorable evidence is close to one. This situation results in the lowest levels of certainty as well the lowest levels of contradiction. Although a nearly true (Nt) state has higher levels of certainty, it still does not represent enough evidence for classifying an anomaly.

The incomplete state (IC) represents a situation with insufficient evidence for the studied given proposition. Thus, an anomaly should not be assigned. This state allows the CPM to avoid possible false alarms in the presence of insufficient information.

For inconsistent states ($\top$) and ($N\top$), intervals with the highest levels of favorable and unfavorable evidence are found, resulting in levels of certainty close to zero and levels of contradiction close to one. This state reflects a discordance between



(a) DoS output.

(b) DDoS output.

(c) Flash crowd output.

Fig. 5. Responses of the CPM built with DSNSF, ACODS, and ARIMA. Each black point inside the diamonds represents an interval analyzed by the CPM with relating coordinates (L_c, L_{ct}). October 30, 2012.

ARIMA and ACODS. The first model interprets a given interval as anomalous, whereas the second one interprets it as normal. Such is the reason why the intervals located at the states are not considered anomalies.

Finally, the paracomplete state $\perp$ with neither favorable nor unfavorable evidence should also not be used to represent anomalous intervals because of the lack of evidence.

The results obtained from the three weeks of testing data regarding true-positive and false-positive rates can be visualized in Fig. 6. The solid black line represents the results using the CPM. The dotted lines represent the results obtained with ARIMA (black) and ACODS (red) based on the confidence band approach. The CPM achieved 95% of the true-positive with a 4% false-positive rate representing a good trade-off outcome. By assuming a 4% false-positive, the models based on the confidence bands achieved close to 92% of the true-positive. As observed, the detection capabilities obtained using the CPM were always superior to those approaches based on the confidence bands, thereby highlighting the influence of its use to decrease the number of false-positive alarms.

The most decisive factors that define the effectiveness of an anomaly detection system are the false alarm and undetected anomaly rates. As argued earlier, the CPM was designed to represent and manipulate inconsistencies over the models used as information sources. The correlation achieved in the extraction of evidence levels in the CPM considers the dataset as the basis of levels of truthfulness, falsehood, inconsistency, and incompleteness, and only assigns an anomaly in the absence of contradiction in the midst of the process. The results obtained illustrate that the CPM adapted along with the ARIMA and ACODS models for our experiments successfully resulted in a tool with the desired characteristics of high true detection rates and low false-positives rates.

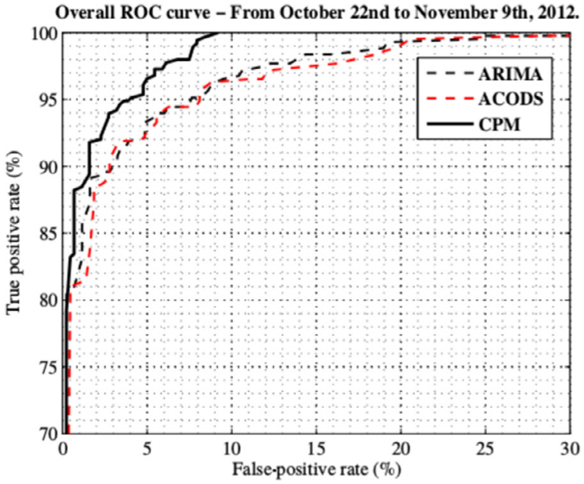


Fig. 6. Overall ROC curve for weekdays from October 22, to November 9, 2012.

7. Conclusions

This study presented a tool, called the correlational paraconsistent machine (CPM), which incorporated unsupervised models for traffic characterization and principles of paraconsistency to address the problem of network-wide anomaly detection. The parameters from the metaheuristic ACODS, and the time series forecasting model ARIMA, were considered to establish the DSNSF concisely defined as profiles for the traffic characterization of backbone segments. Through a functional algorithm based on the paraconsistent logic (PL), these profiles were used as sources of information, which helped the CPM handle uncertain and contradictory information found in the network flow analysis.

We performed tests in real network traces to validate our proposal. In addition, we compared the performance of the CPM with that of traditional anomaly detection approaches based on confidence bands. Our experimental results showed that the CPM was very effective in improving anomaly detection and decreasing false-positive rates.

We highlight four main contributions of this work: (i) implementation of a system for anomaly detection using IP flow analysis, (ii) use of different traffic features to characterize anomalies, (iii) adaptation of the ACODS and ARIMA models to generate DSNSFs, and (iv) application of PL for anomaly detection.

The CPM is the result of applying paraconsistent criteria in a solution for anomaly detection. Therefore, our future directions focus on extending its functions to identify additional anomalies. This is important because new types of anomalies will continue to appear, highlighting new features to be considered in anomaly detection. Moreover, we intend to investigate possible variations in our approach to formalize and apply PL concepts. Thus, our main goal is to improve the treatment of incompleteness and inconsistency for our problem setup and achieve better results.

Acknowledgments

This work has been partially supported by the National Council for Scientific and Technological Development (CNPq) of Brazil under Grant of Project 308348/2016-8; Instituto de Telecomunicacões, Next Generation Networks and Applications Group (NetGNA), Covilhã Delegation; the Government of the Russian Federation, Grant 074-U01; national funding from the FCT – Fundacã o para a Ciê ncia e a Tecnologia through the UID/EEA/500008/2013 Project; national funding from the FCT through the PEst-OE/EEI/LA0008/2013 Project; and FINEP, with resources from Funttel, Grant No. 01.14.0231.00, under the Radiocommunication Reference Center (CRR) project of the National Institute of Telecommunications (Inatel), Brazil.

References

[1] J.M. Abe, Paraconsistent artificial neural networks: An introduction, in: Knowledge-Based Intelligent Information and Engineering Systems, in: Lecture Notes in Computer Science, 3214, 2004, pp. 942–948, doi:10.1007/978-3-540-30133-2_124.

[2] J.M. Abe, J.I.S. Filho, Inconsistency and electronic circuits, Proc. EIS 98 (1998) 191–197.

[3] A. Amokrane, R. Langar, R. Boutaba, G. Pujolle, Flow-based management for energy efficient campus networks, IEEE Trans. Netw. Serv. Manage. 12 (4) (2015) 565–579, doi:10.1109/TNSM.2015.2501398.

[4] M.V.O. Assis, M.L. Proenča Jr., Scorpius: sflow network anomaly simulator, J. Comput. Sci. 11 (2015) 662–674, doi:10.3844/jcssp.2015.662.674.

[5] M.V.O. Assis, J.J.P.C. Rodrigues, M.L. Proença Jr., A novel anomaly detection system based on seven-dimensional flow analysis, in: 2013 IEEE Global Communications Conference (GLOBECOM), 2013, pp. 735–740, doi:10.1109/GLOCOM.2013.6831160.

[6] M.V.O. Assis, J.J.P.C. Rodrigues, M.L. Proença Jr., A seven-dimensional flow analysis to help autonomous network management, Inf. Sci. (Ny) 278 (0) (2014) 900–913, doi:10.1016/j.ins.2014.03.102.

[7] M. Bhuyan, D. Bhattacharyya, J. Kalita, Network anomaly detection: methods, systems and tools, IEEE Commun. Surv. Tutorials, 16 (1) (2014) 303–336, doi:10.1109/SURV.2013.052213.00046.

[8] M.H. Bhuyan, D. Bhattacharyya, J. Kalita, A multi-step outlier-based anomaly detection approach to network-wide traffic, Inf. Sci. (Ny) 348 (2016) 243–271, doi:10.1016/j.ins.2016.02.023.

[9] H.A. Blair, V.S. Subrahmanian, Paraconsistent logic programming, in: Proc. of the Seventh Conference on Foundations of Software Technology and Theoretical Computer Science, Springer-Verlag, London, UK, 1987, pp. 340–360.

[10] J.D. Brutlag, Aberrant behavior detection in time series for network monitoring, in: Proceedings of the 14th Systems Administration Conference (LISA 2000), 2000, pp. 139–146.

[11] J. Camacho, A. Pérez-Villegas, P. García-Teodoro, G. Maciá-Fernández, PCA-based multivariate statistical network monitoring for anomaly detection, Comput. Secur. 59 (C) (2016) 118–137, doi:10.1016/j.cose.2016.02.008.

[12] L.F. Carvalho, S.B. Jr., L.S. Mendes, M.L. Proença Jr., Unsupervised learning clustering and self-organized agents applied to help network management, Expert Syst. Appl. 54 (2016) 29–47, doi:10.1016/j.eswa.2016.01.032.

[13] S. Chang, X. Qiu, Z. Gao, K. Liu, F. Qi, A flow-based anomaly detection method using sketch and combinations of traffic features, in: 2010 International Conference on Network and Service Management (CNSM), 2010, pp. 302–305, doi:10.1109/CNSM.2010.5691206.

[14] M. Dorigo, T. Stützle, Ant colony optimization, Bradford Company, Scituate, MA, USA, 2004.

[15] H. Feng, Y. Shu, Study on network traffic prediction techniques, in: Proc. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005, pp. 1041–1044, doi:10.1109/WCNM.2005.1544219.

[16] W. Feng, Q. Zhang, G. Hu, J.X. Huang, Mining network data for intrusion detection through combining SVMS with ant colony networks, Future Generation Computer Systems 37 (0) (2014) 127–140, doi:10.1016/j.future.2013.06.027.

[17] G. Fernandes Jr., J.J.P.C. Rodrigues, M.L. Proenča Jr., Autonomous profile-based anomaly detection system using principal component analysis and flow analysis, Appl. Soft. Comput. 34 (2015) 513–525, doi:10.1016/j.asoc.2015.05.019.

[18] J.I.S. Filho, Treatment of uncertainties with algorithms of the paraconsistent annotated logic, J. Intell. Learn. Syst. Appl. 4 (2) (2012) 144–153.

[19] J.I.S. Filho, G.T. Lambert, J.M. Abe, Uncertainty treatment using paraconsistent logic: introducing paraconsistent artificial neural networks, 211, IOS Press Inc, 2010.

[20] M.S. Garshasbi, Fault localization based on combines active and passive measurements in computer networks by ant colony optimization, Reliab. Eng. Syst. Saf. 152 (2016) 205–212, doi:10.1016/j.ress.2016.03.017.

[21] P. Gogoi, D. Bhattacharyya, B. Borah, J.K. Kalita, A survey of outlier detection methods in network anomaly identification, Comput. J. 54 (4) (2011) 570–588, doi:10.1093/comjnl/bxr026.

[22] J. Grana, D. Wolpert, J. Neil, D. Xie, T. Bhattacharya, R. Bent, A likelihood ratio anomaly detector for identifying within-perimeter computer network attacks, J. Netw. Comput. Appl. 66 (C) (2016) 166–179, doi:10.1016/j.jnca.2016.03.008.

[23] M. Grill, T. Pevný, M. Rehak, Reducing false positives of network anomaly detection by local adaptive multivariate smoothing, J. Comput. Syst. Sci. 83 (1) (2017) 43–57, doi:10.1016/j.jcss.2016.03.007.

[24] R.C. Guido, S. Barbon Jr., R.D. Solgon, K.C.S. Paulo, L.C. Rodrigues, I.N. Silva, J.A.P.L. Escola, Introducing the discriminative paraconsistent machine (dpm), Inf. Sci. (Ny) 221 (2013) 389–402, doi:10.1016/j.ins.2012.09.028.

[25] S. Hansman, R. Hunt, A taxonomy of network and computer attacks, Comput. Secur. 24 (1) (2005) 31–43.

[26] F. Iglesias, T. Zseby, Analysis of network traffic features for anomaly detection, Mach. Learn. 101 (1–3) (2015) 59–84, doi:10.1007/s10994-014-5473-9.

[27] H. Jiang, Q. Yu, Y. Gong, An improved ant colony clustering algorithm, in: 2010 3rd International Conference on Biomedical Engineering and Informatics, 6, 2010, pp. 2368–2372, doi:10.1109/BMEI.2010.5639719.

[28] B. Kavitha, S. Karthikeyan, P.S. Maybell, An ensemble design of intrusion detection system for handling uncertainty using neutrosophic logic classifier, Knowl. Based Syst. 28 (0) (2012) 88–96, doi:10.1016/j.knosys.2011.12.004.

[29] A. Lakhina, M. Crovella, C. Diot, Diagnosing network-wide traffic anomalies, in: Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, in: SIGCOMM '04, ACM, New York, NY, USA, 2004, pp. 219–230, doi:10.1145/1015467.1015492.

[30] A. Lakhina, M. Crovella, C. Diot, Mining anomalies using traffic feature distributions, in: Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, in: SIGCOMM '05, ACM, New York, NY, USA, 2005, pp. 217–228.

[31] B. Li, J. Springer, G. Bebis, M.H. Gunes, A survey of network flow applications, J. Netw. Comput. Appl. 36 (2) (2013) 567–581, doi:10.1016/j.jnca.2012.12.020.

[32] K. Li, W. Zhou, P. Li, J. Hai, J. Liu, Distinguishing DDoS attacks from flash crowds using probability metrics, in: Proceedings of the 2009 Third International Conference on Network and System Security, in: NSS '09, IEEE Computer Society, Washington, DC, USA, 2009, pp. 9–17, doi:10.1109/NSS.2009.35.

[33] G.E. Pelham, G.M. Jenkins, Time series analysis: forecasting and control, 3rd, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1994.

[34] E.H.M. Pena, S. Barbon Jr., J.J.P.C. Rodrigues, M.L. Proença Jr., Anomaly detection using digital signature of network segment with adaptive ARIMA model and paraconsistent logic, in: 2014 IEEE Symposium on Computers and Communication (ISCC), 2014, pp. 1–6, doi:10.1109/ISCC.2014.6912503.

[35] M.L. Proenča Jr., G. Fernandes Jr., L.F. Carvalho, M.V.O. Assis, J.J.P.C. Rodrigues, Digital signature to help network management using flow analysis, Int. J. Netw. Manage. 26 (2) (2016) 76–94, doi:10.1002/nem.1892.

[36] V. Ramos, A. Abraham, ANTIDS: Self organized ant-based clustering model for intrusion detection system, in: A. Abraham, Y. Dote, T. Furuhashi, M. Köppen, A. Ohuchi, Y. Ohsawa (Eds.), Soft Computing as Transdisciplinary Science and Technology, Advances in Soft Computing, 29, Springer Berlin Heidelberg, 2005, pp. 977–986.

[37] V. Sharma, A. Grover, A modified ant colony optimization algorithm (MACO) for energy efficient wireless sensor networks, Optik - Int. J. Light Electron Optics 127 (4) (2016) 2169–2172, doi:10.1016/j.ijleo.2015.11.117.

[38] H.F.L. Silva, J.M. Abe, R. Anghinah, Application of paraconsistent artificial neural networks as a method of aid in the diagnosis of Alzheimer disease, J. Med. Syst. 34 (6) (2010) 1073–1081.

[39] J. Song, H. Takakura, Y. Okabe, K. Nakao, Toward a more practical unsupervised anomaly detection system, Inf. Sci. (Ny) 231 (0) (2013) 4–14, doi:10.1016/j.ins.2011.08.011. Data Mining for Information Security.

[40] A. Soule, K. Salamatian, N. Taft, Combining filtering and statistical methods for anomaly detection, in: Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement, in: IMC '05, USENIX Association, Berkeley, CA, USA, 2005. 31–31.

[41] A. Sultana, A. Hamou-Lhadj, M. Couture, An improved hidden Markov model for anomaly detection using frequent common patterns, in: 2012 IEEE International Conference on Communications (ICC), 2012, pp. 1113–1117, doi:10.1109/ICC.2012.6364527.

[42] B. Tellenbach, M. Burkhart, D. Schatzmann, D. Gugelmann, D. Sornette, Accurate network anomaly classification with generalized entropy metrics, Comput. Netw. 55 (15) (2011) 3485–3502, doi:10.1016/j.comnet.2011.07.008.

[43] P. Truong, F. Guillemin, Dynamic binary tree for hierarchical clustering of IP traffic, in: 2007 IEEE Global Telecommunications Conference, GLOBECOM '07, 2007, pp. 6–10.

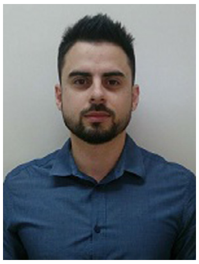
[44] C.-H. Tsang, S. Kwong, Ant colony clustering and feature extraction for anomaly intrusion detection, in: A. Abraham, C. Grosan, V. Ramos (Eds.), Swarm Intelligence in Data Mining, Studies in Computational Intelligence, 34, Springer Berlin Heidelberg, 2006, pp. 101–123.

[45] A. Yaacob, I.K.T. Tan, S.F. Chien, H.K. Tan, ARIMA based network anomaly detection, in: ICCSN '10 Second International Conference on Communication Software and Networks, 2010, pp. 205–209, doi:10.1109/ICCSN.2010.55.

[46] S. Yu, W. Zhou, W. Jia, S. Guo, Y. Xiang, F. Tang, Discriminating DDOS attacks from flash crowds using flow correlation coefficient, IEEE Trans. Parallel Distrib. Syst. 23 (6) (2012) 1073–1080, doi:10.1109/TPDS.2011.262.

[47] H. Zhang, G. Lu, M.T. Qassrawi, Y. Zhang, X. Yu, Feature selection for optimizing traffic classification, Comput. Commun. 35 (12) (2012) 1457–1471, doi:10.1016/j.comcom.2012.04.012.

[48] B. Zhu, S. Sastry, Revisit dynamic ARIMA based anomaly detection, in: 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust (PASST), and 2011 IEEE Third International Conference on Social Computing (SocialCom), 2011, pp. 1263–1268, doi:10.1109/PASSAT/SocialCom.2011.84.



Eduardo H. M. Pena is a Ph.D. candidate on informatics at Federal University of Paraná, Curitiba, Brazil. He received a M.Sc degree in Computer Science from the Computer Science Department at State University of Londrina (UEL), Brazil, in 2014. He has experience in Computer Science with emphasis in Computer Networks and is part of the research group "Computer Networks and Data Communication". He is currently an assistant professor at Federal University of Technology-Paraná in undergraduate programs. His main research interests are Computer Networks, Network Operation, Management, and Security.

Luiz F. Carvalho is a Ph.D. candidate in Electrical Engineering and Telecommunications at State University of Campinas, Brazil. He completed his master degree in Computer Science at State University of Londrina (UEL) in 2014. Currently, he is a lecturer and member of the "Computer Networks and Data Communication" research group at UEL. His main research interests are management and security of computer networks and software-defined networking.

Sylvio Barbon Jr. Ph.D. Received his B.S degree in Computer Science from Centro Universitário do Norte Paulista (2005), and master degree in Computational Physics from University of Sã o Paulo (2007), degree in Computational Engineering from Centro Universitário de Votuporanga (2008) and Ph.D. degree (2011) from IFSC/USP such as master degree. He is currently a professor at State University of Londrina (UEL), Brazil, in postgraduate and graduate programs. His research interests include Digital Signal Processing, Pattern Recognition and Machine Learning.

Joel J.P.C. Rodrigues (joeljr@ieee.org) [S'01, M'06, SM'06] is a professor and senior researcher at the National Institute of Telecommunications (Inatel), Brazil and senior researcher at the *Instituto de Telecomunicacões*, Portugal. He has been professor at the University of Beira Interior (UBI), Portugal and visiting professor at the University of Fortaleza (UNIFOR), Brazil. He received the *Academic Title of Aggregated Professor* in informatics engineering from UBI, the Habilitation in computer science and engineering from the University of Haute Alsace, France, a PhD degree in informatics engineering and an MSc degree from the UBI, and a five-year BSc degree (licentiate) in informatics engineering from the University of Coimbra, Portugal. His main research interests include e-health, sensor networks and IoT, vehicular communications, and mobile and ubiquitous computing. Prof. Joel is the leader of NetGNA Research Group, the President of the scientific council at ParkUrbis – Covilhã Science and Technology Park, the Past-Chair of the IEEE ComSoc Technical Committee on eHealth, the Past-chair of the IEEE ComSoc Technical Committee on Communications Software, Steering Committee member of the IEEE Life Sciences Technical Community and Publications co-Chair, and Member Representative of the IEEE Communications Society on the IEEE Biometrics Council. He is the editor-in-chief of the International Journal on E-Health and Medical Communications, the editor-in-chief of the Recent Advances on Communications and Networking Technology, the editor-in-chief of the Journal of Multimedia Information Systems, and editorial board member of several high-reputed journals. He has been general chair and TPC Chair of many international conferences, including IEEE ICC, GLOBECOM, and HEALTHCOM. He is a member of many international TPCs and participated in several international conferences organization. He has authored or coauthored over 500 papers in refereed international journals and conferences, 3 books, and 2 patents. He had been awarded several Outstanding Leadership and Outstanding Service Awards by IEEE Communications Society and several best papers awards. Prof. Rodrigues is a licensed professional engineer (as senior member), member of the Internet Society, an IARIA fellow, and a senior member ACM and IEEE.

Mario Lemes Proença Jr. is an Associate Professor and leader of the research group that study computer's network in the Computer Science Department at State University of Londrina (UEL), Brazil. He received the Ph.D. degree in Electrical Engineering and Telecommunications from State University of Campinas (UNICAMP) in 2005. He received the title of M.Sc degree in Computer Science from the Informatics Institute of Federal University of Rio Grande do Sul (UFRGS), in 1998. He has authored or coauthored over 90 papers in refereed international journals and conferences, books chapters, and 1 software register patent. His research interests include Computer Network, IT Governance, Network Operations, Management and Security. He has supervised 12 M.Sc. and 2 Ph.D students. He has been a Master's supervisor at computer science in State University of Londrina and Ph.D. supervisor in Department of Electrical Engineering at UEL.