SENATUS: An Approach to Joint Traffic Anomaly Detection and Root Cause Analysis

Atef Abdelkefi[1], Yuming Jiang[1], Sachin Sharma[2]
[1]NTNU, Norwegian University of Science and Technology, Norway. [2]National College of Ireland, Ireland

***Abstract*—In this paper, we propose a novel approach, called SENATUS, for joint anomaly detection and root-cause analysis. Inspired from the concept of a senate, the key idea of the proposed approach is divided into three stages: election, voting and decision. At the election stage, a small number of traffic flow sets (termed as senator flows) are chosen based on the *K-sparse approximation* technique, which can be used to represent approximately the total (usually huge) set of traffic flows. In the voting stage, Principal Component Pursuit (PCP) analysis is used for anomaly detection on the senator flows. In addition, the detected anomalies are correlated across traffic features to identify the most possible anomalous time bins. Finally, in the decision stage, a machine learning (ML) technique is applied to the senator flows of anomalous time bins to find the root cause of the anomalies. The performance of SENATUS is evaluated using real traffic traces collected from a Pan European network, GEANT, and compared against another approach which detects anomalies using lossless compression of traffic histograms. The evaluation shows that SENATUS has higher effectiveness in diagnosing traffic anomalies.**

***Keywords*—Principal Component Pursuit; K-sparse Approximation; Random Decision Tree; Network Traffic Anomaly Detection**

I. Introduction

New applications, emerging every year or even every day, have made it imperative to investigate effective techniques that can extract communication patterns from Internet traffic for security management. Among others, identifying anomalous events such as denial-of-service (DoS) attacks, distributed DoS (DDoS) attacks and network scans is a crucial task.

A key challenge in traffic anomaly detection is the curse of dimensionality, which refers to the problems that arise when analyzing and organizing data in a high-dimensional space. For example, in a pan-European network, the GEANT network, it was recorded (even after traffic sampling) that there were around 10^9 flows distributed over 2^{16} ports and 2^{32} IP addresses over a 15-minute time interval on a link. In addition, in order to identify the possible root cause of traffic anomaly on a time interval, correlating analysis on different traffic features, e.g. source Autonomous System (AS) and destination port number, is often necessary. This implies that the analysis will have to even deal with those numbers in a combinatorial manner, which further complicates the analysis making it hardly implementable.

In the literature, an extensive body of prior investigation on traffic anomaly detection exists (e.g. see [1],[2]). In these works, unusual abrupt variations in traffic time series, defined as traffic anomalies, raise alarms. Unfortunately, the practical usefulness of the reported alarms is often limited, mainly due to the tremendous amount of time and effort additionally required to analyze the root cause of the reported alarms [3]. This results in another challenge and a pressing need for approaches that perform anomaly detection and root cause analysis jointly.

To address these challenges, we propose a novel approach, called SENATUS, in this paper. SENATUS conducts root-cause analysis jointly with traffic anomaly detection through traffic aggregation and lossy compression. Specifically, SENATUS detects time intervals where anomalous events are suspicious to occur, identifies suspicious aggregate flows, and diagnoses the type of anomaly. In brief, SENATUS operates in three stages: *election*, *voting*, and *decision*. Inspired from the concept of senate, a key idea and the starting point of SENATUS is to choose or elect a small number of traffic flow sets (termed as senator flows) to represent the total (usually huge) set of traffic flows. Then, on the elected senator flows, traffic anomaly detection is performed and the results for each time bin are used together to decide or vote if the time bin may be considered as anomalous. Finally, root cause analysis is conducted on every anomalous time bin to identify the root cause of anomaly type for that time bin.

The techniques proposed to use in the three stages are, respectively, K-sparse approximation of traffic histogram [4], Principal Component Pursuit (PCP) [5], and random decision tree (RDT) machine learning (ML) classification [6]. These techniques, together with the ideas and heuristics proposed in the paper when using them, e.g. in senator flow election and in anomalous time bin voting, form the novelty of SENATUS.

The specific contributions of this paper are as follows:

- Instead of performing analysis directly on the original traffic histograms that suffer from the curse of dimensionality, we propose to use K-sparse approximation to select only the flow sets whose feature values are among the top K on the traffic histogram. This forms the core of the SENATUS election stage to choose senator flows.
- At the voting stage, SENATUS performs PCP analysis on the time series of each senator flow to detect time bins with abrupt changes. The detected time bins with abrupt changes are correlated across senator flows to flag the most possible anomalous bins.
- SENATUS, at the decision stage, performs root-cause analysis for each anomalous time bin, based on the joint application of two intuitive heuristics and a linear time signature and machine learning-based technique.
- The performance of SENATUS is evaluated and compared using real traffic traces, which shows appealing effectiveness of SENATUS, outperforming the compared approach.

The rest of this paper is structured as follows. Sec. II provides background basics and Sec. III gives a detailed introduction of SENATUS. Sec. IV provides evaluation and results. Sec. V discusses the related work. Finally, the paper is concluded in Sec. VI.

II. BASICS OF SENATUS

A. Targeted Anomalies

Table I lists the categories of anomalies considered in this paper and their traffic characteristics, though the central ideas of SENATUS are not limited to them.

TABLE I. TARGETED ANOMALY TYPES

Anomaly Type	Traffic Characteristics
DoS	Small or large-sized flows sent from one source AS (Autonomous System) via one or multiple source ports to one destination AS on one or multiple destination ports
DDoS	Many small or large-sized flows sent from one or many source AS via one or multiple source ports to one destination AS on one or multiple destination ports
Network Scan	*Many small sized flows from one source AS via one source port to one or many destination AS on one destination port*

Table I implies that the targeted anomalies are often carried by *small-sized flows*. It is worth highlighting that while applying this implication may lead to under-detection of anomalies present in large-sized flows, focusing on small-sized flows can reduce the risk of false positives in anomaly detection. This is because large-sized flows frequently involve benign activities such aslarge file transfers, and high-volume streaming [7].

B. Traffic Histogram and K-sparse Approximation

The proposed SENATUS approach is a traffic histogram-based approach. A *traffic histogram* is the distribution of the amount of traffic (in number of flows, packets or bytes) over all possible values of a traffic *feature*. A feature is a field in the header of a packet, such as source port, or a *function* of some header field values, such as AS numbers [8].

While there are many traffic features that may be analyzed, we focus on four of them, which are source AS (*srcAS*), destination AS (*dstAS*), source port (*srcPort*) and destination port (*dstPort*). This is motivated by the traffic characteristics of the targeted anomaly types as discussed in Table I.

K-sparse approximation is a technique proposed for traffic histogram compression [4]. It relies on the fact that a traffic histogram may be highly *compressible* if it exhibits a *power-law* decay when sorted, and consequently, one may use the top K-feature values to approximate the original traffic histogram.

More formally, consider a traffic histogram X with n possible distinct feature values (e.g. port 1, port 2, ..., port n). Let $X' \equiv (x'_{(1)}, x'_{(2)}, ...x'_{(n)})$ denote the sorted histogram, where the coefficients are in the non-increasing order, i.e. $x'_{(1)} \geq x'_{(2)} \geq ...x'_{(n)}$. Suppose that the sorted histogram decays according to a power law as, for all $i = 1, 2, \ldots, n$,

$$x'_{(i)} \leq R \cdot i^{(-\frac{1}{p})} \tag{1}$$

where R is a normalization constant and $0 < p \leq 1$ is a scaling parameter. Then, X can be approximated by the first few *"top"*-K coefficients, i.e. $x'_{(1)}, \ldots, x'_{(K)}$, with approximation error σ_K upper-bounded by [4]:

$$\sigma_K = ||X' - X'_K||_2 \leq (ps)^{-\frac{1}{2}} \cdot R \cdot K^{(-s)} \tag{2}$$

where X'_K has in total n elements defined as

$$X'_K \equiv (x'_{(1)}, \ldots, x'_{(K)}, 0, \ldots, 0)$$

and $s = \frac{1}{p} - \frac{1}{2}$. If the decay of the coefficients is rapid, a small value of $K (<< n)$ can lead to close approximation.

C. Principle Component Pursuit (PCP)

Principle Component Analysis (PCA) is a statistical tool for high-dimensional data analysis and dimensionality reduction. It basically assumes that the data approximately lies on a low-dimensional linear subspace. Let $X \in \mathbb{R}^{n_1 \times n_2}$ be a matrix of interest. A foundation of PCA is that it seeks the best rank-k estimate A of data matrix X by solving:

$$\min \ \|X - A\|_2, \ subject \ to \ \ rank(A) \leqslant k.$$

When applying PCA to structural analysis of network traffic [9][10] and traffic anomaly detection [11], the essential idea is to decompose X into two components, normal component N and anomalous component A, i.e. $X = N + A$. In the decomposition, the PCA technique attempts to find the matrix A such that the matrix $N = X - A$ has the lowest possible rank. More formally, the structural analysis tries to solve the following optimization problem:

$$\min_{N,A} \|A\|_0, \ \text{subject to } X = N + A \text{ and } rank(N) \leq k \tag{3}$$

where $rank(N)$ denotes the rank of a matrix N, $\| \ \|_0$ denotes the ℓ_0-norm, i.e. the cardinality of the non-zero elements.

However, the optimization problem (3) is NP-hard [5]. Fortunately, based on recent advances in optimization theory, it has been proved that the nuclear norm, i.e, the sum of singular values, recovers the low rank component N [5]. In addition, the ℓ_1 norm, i.e, the sum of absolute values, recovers component A in terms of sign and support with remarkable robustness to the outliers in comparison to the ℓ_2 norm.

Accordingly, Eq. (3) can be solved using a convex optimization problem called *Principal Component Pursuit* [5] as:

$$\min_{N,A} \ \|N\|_* + \lambda \|A\|_1, \ subject \ to \ X = N + A, \tag{4}$$

where $\| \ \|_*$ denotes the nuclear norm, i.e., the sum of the singular values of the normal matrix, $\| \ \|_1$ denotes the ℓ_1-norm of the anomalous events matrix A, and $\lambda > 0$ is a weighting parameter.

We highlight that, while (3)-based PCA has been widely used in anomaly detection, the study in [2] indicates that (4)-based PCP provides more stable performance, in addition to its reduced complexity in finding A, and has seen increased applications, e.g., [12].

D. Random Decision Tree

SENATUS uses a machine learning (ML) technique, random decision tree (RDT) [13], to find the root cause of anomalies. In fact, RDT is an ensemble of decision trees. The process for generating a tree is as follows.

First, it starts with a list of features or attributes from the data set. Then, it generates a tree by randomly choosing one of the features *without using any training data*. The tree stops growing once the height limit is reached. Then, it uses the training data to update the statistics of each node. Note that only the leaf nodes need to record the number of examples of different classes that are classified through the nodes in the tree. The training data is scanned exactly once to update the statistics in multiple random trees. A further explanation of RDT can be found in Sec. III-D.

III. Detailed SENATUS Approach

In this section, an overview of the proposed three-stage SENATUS approach is first presented, followed by an introduction to each of the three stages in detail.

A. Overview

SENATUS adopts the idea of senate, where senators elected from a population make decision for the population based on voting. Similarly, SENATUS involves three stages, which are the *election* stage, the *voting* stage and the *decision* stage.

Specifically, SENATUS first pre-filters the traffic to construct the base population, from which the top-K traffic feature values, namely *senators*, are extracted. For each feature value, a time series of the number of flows is constructed. Such time series are organized in a subspace called *senator subspace*, and analyzed using PCP to detect abrupt changes. The detected abrupt variations form the votes for each time bin to decide if the time bin is anomalous, and to flag the set of suspicious flows for each anomalous time bin. Finally, this set of flagged flows is further processed using RDT to diagnose the root-cause of anomalous behavior in the time bin.

B. Election Stage

In this stage, the traffic traces are pre-filtered using two heuristic H1 and H2 defined in Table II. In H1, *small size flows* are defined as those flows whose packet counts are not larger than threshold value α and in H2, *small size flows* are defined as those flows whose byte counts are not larger than threshold value β, in a time bin. A detailed investigation of the effect of the threshold values will be presented in Sec. IV.

TABLE II. Heuristics

Heuristic	Definition
H1	Small packet count per flow: # of packets $\leq \alpha$
H2	Small byte count per flow: # of bytes $\leq \beta$

After filtering, for every measurement time bin, K-sparse approximation is applied to the flow number histogram of each of the four traffic features (srcAS, dstAS, srcPort, dstPort) on the time bin. These traffic feature are most common feature considered in networking. Here, the idea behind pre-filtering traffic before applying K-sparse approximation is that, with pre-filtering, it is more likely that an anomalous feature value is included in the selected top K components.

After K-sparse approximation is applied for a traffic feature (e.g. srcAS) on a time bin t, K top values of this feature j for this time bin are obtained. Let $I_j(t)$ denote the set of these K top values of the feature j on the time bin t. Suppose there are N time bins in the traffic trace. Let I_j be the consolidation of these N sets of such feature values, i.e. $I_j = I_j(1) \cup \cdots \cup I_j(N)$. Then, for every feature $j \in \{$*srcAS, dstAS, srcPort, dstPort*$\}$, if it has a value i in I_j, i.e. $i \in I_j$, this feature value defines a *senator flow* or simply *senator* for the feature j.

We remark that each senator is indeed a flow aggregate where all flows have the same feature value. Accordingly, we have a three-dimensional flow count matrix $Y(t, i_j, j)$ that is defined on time $t(= 1, \ldots, N)$ with feature value $i_j \in I_j$ across all features $j \in \{$*srcAs, dstAs, srcPort, dstPort*$\}$, which we call *the senator subspace*.

C. Voting Stage

At this stage, SENATUS analyzes each senator's time series using PCP to detect abrupt changes on the time series. The detected abrupt variations, called votes, are correlated to identify or vote the most likely anomalous time bins.

For every feature $j \in \{$*srcAS, dstAS, srcPort, dstPort*$\}$, let $X(t, i_j)$ be its traffic amount time series matrix. Specifically, the element at (t, i_j) of X records the number of flows that have the same feature value i_j (e.g. srcPort 80) at time bin t in the measurement period. Essentially, $X(t, i_j) = Y(t, i_j, j)$ with j fixed to be the considered feature.

Applying the PCP technique described in Sec. II.C to the time series matrix X, with $n_1(= N)$ being the number of time bins in the measurement period and $n_2(= I_j)$ being the number of senators from feature j, the corresponding anomalous events matrix A is obtained. The positive-value elements in the obtained anomalous events matrices are referred to as *votes*.

For every time bin t, a feature j (e.g. srcPort) is flagged anomalous if (at least) one of its values in I_j makes a vote on this time bin. In other words, at least one senator time series of this feature has abrupt variation at t. For a time bin t, if all features $\{$*srcAS, dstAS, srcPort, dstPort* $\}$ give their vote on it, this time bin is considered to be an anomalous time bin.

D. Decision Stage

In this stage, SENATUS diagnoses the root-cause for every anomalous time bin. In particular, the objective is to investigate if the traffic behavior on an anomalous time bin is due to one of the focused anomaly types listed in Table I. To this aim, the following actions are performed.

1) Identifying Suspicious Flow Aggregate: Let m_j denote the cardinality or the number of senator members of I_j. In addition, define a flow aggregate using values of $srcAS$, $dstAS$, $srcPort$ and $dstPort$. Then, for the time bin, there are $M = m_{srcAS} \times m_{dstAS} \times m_{srcPort} \times m_{dstPort}$ flow aggregates, which we call *suspicious flow aggregates*. These combinations essentially tell that we consider flows that might be originated from any suspicious source AS at any suspicious port and target at any suspicious destination AS at any suspicious port. Note that, for some of these combinations, the number of flows in the flow aggregate is zero. Such flow aggregates will be skipped in later analysis. We call the remaining ones the effective suspicious flow aggregates.

2) Root Cause Analysis: After identifying the set of suspicious aggregate flows, we aim to infer the event that has caused flagging the time bin as anomalous. To this aim, a simple (with linear time complexity) threshold-based classification algorithm is performed, which is introduced below.

(i) For every $dstIP$ that is included in the $dstAS$ of the suspicious flow aggregate, find the number of flows that are destined to this dstIP, regardless of their $srcAS$, $srcPort$ or $dstPort$. Take the maximum of all such numbers and call it the anomaly intensity. Compare this intensity with a threshold, denoted as θ_1. If the former is greater, output is DDoS. Otherwise, perform the next.

(ii) For every $\{srcIP, dstIP\}$ pair included in the $\{srcAS, dstAS\}$ pair of the suspicious flow aggregate, perform similarly as above: Find the number of flows that have

the same $srcIP$ and $dstIP$, regardless of their $srcPort$ or $dstPort$. Take the maximum of all such numbers, and compare with another threshold, denoted as θ_2. If the former is greater, output is DoS. Otherwise, perform the next.

(iii) For the $dstPort$ of the suspicious flow aggregate and every $srcIP$ that is included in the $srcAS$ of the aggregate, find the number of flows that are originated from this $srcIP$ and destined to this $dstPort$, regardless of their $srcPort$ or $dstAS$. Take the maximum of all such numbers and compare the intensity value with another threshold, denoted as θ_3. If the former is greater, output is Network Scan. Otherwise, repeat these steps for the next effective suspicious flow aggregate.

(iv) The above procedure is repeated until an attack signature comparison is successful. Or, in the end, the anomaly type cannot be identified and in this case the alarm is reported as false positive.

3) Threshold Values: As described above, there are three threshold parameters, $\theta_i, (i = 1, 2, 3)$, used in the classification algorithm. We highlight that they are key parameters in the root cause analysis. To decide them, the RDT ML algorithm [13] is used: In addition to being fast and easy to interpret, RDT exhibits optimality in probability estimation [14].

In our RDT algorithm, each anomaly is mapped as a point into a space where anomalies are classified based on their intensities. Under this taxonomy, we create a set of labeled instances with one intensity attribute: the number of flows. This set is mapped to one of the three anomaly classes: DoS, DDoS and Scans. The labeled instances serve as input to the RDT learning algorithm that outputs a tree which indicates the range of intensities per anomaly class.

Specifically, our RDT algorithm works as follows. It is an iterative algorithm. For time $T = 1, 2, \ldots, N$, the inputs are the set of unknown anomalies at this time and the set of previously labeled anomalies for times $1, \ldots, T-1$. The algorithm first applies the decision tree technique on the labeled items of anomalies and their corresponding intensity. The output is a tree $DT_{[1,T-1]} = (Br(i,j), Class(j)), i \leq size(DT), j \leq size(Br)$ where each path constitutes a set of branches from the root to a leaf. A branch j of a path i, $Br(i,j)$, introduces an upper or a lower bound of an anomaly intensity while a leaf of a path i: $Class(i)$ corresponds to a class of anomaly.

We then explore the output tree and map it into a set of association rules to enable classification of anomalies based on their intensity. A rule is an antecedent $\{Br(1,i), ...Br(n,i)\}$ which represents the i_{th} path of the tree and a consequent, i.e, a class of anomalies. Since only one attribute (anomaly intensity) is adopted in the learning process, the branch $Br(n,i)$ from each leaf to its direct parent defines each association rule antecedent which is, defined as comparator, i.e, $\geq, \leq$ and a value, i.e, a threshold of anomaly intensity. The threshold values are then extracted by simple parsing of the set of rule antecedents: rule antecedents which introduce an upper bound of intensity for a class of anomaly are ignored, while those which introduce a lower bound (comparator=$\{\geq\}$) are considered. A lower bound of anomaly intensity in association rule antecedent represents a candidate threshold. The output threshold value, $\theta_i (i = 1, 2, 3)$, for a given class of anomalies is the minimum among all candidate thresholds.

IV. Evaluation

In this section, we evaluate the performance of SENATUS and compare it against that of a literature approach.

A. Dataset

The measurement dataset used in the evaluation is comprised of four traffic traces collected from the GEANT network. GEANT is a pan-Europe backbone network interconnecting European NRENs (National Research and Educational Networks) and provides them access to other NRENs and the Internet using dedicated links. The traces in the adopted dataset were collected from the following four links: (1) a peering link between the Internet and the Frankfurt router in GEANT (Trace $\mathcal{A}$); (2) a peering link between the Internet and the Vienna router in GEANT (Trace $\mathcal{B}$); (3) a peering link between the Internet and the Amsterdam router in GEANT (Trace $\mathcal{C}$); (4) a peering link between the Internet and the Copenhagen router in GEANT (Trace $\mathcal{D}$).

The four traces, which are available from GEANT on request, were collected during a 18-day measurement period in June – July 2011, and involve flow records over 15-minute measurement time bins at a sampling rate of $1/1000$. A flow record involves different information fields such as source and destination IP address and AS numbers, source and destination ports, transport protocol (TCP/UDP), the duration of a flow (in second) and the flow size in packets and bytes. We analyzed the four traces and found that they have low rank traffic metrics and sparse abrupt variations [15].

B. Apriori Approach

When evaluating the performance of SENATUS, it is compared against an approach based on [16], which we call Apriori approach in later presentation. In this approach, it first uses histogram-based detectors to identify suspicious flows and then applies association rule mining to find and summarize anomalous event flows. For the former, it uses Kullback-Leibler (KL) distance, and for the latter, it makes use of the Apriori algorithm introduced in [17].

We remark that the KL distance idea has been widely applied for anomaly detection [8]. Motivated with the basic assumption that anomalies deteriorate traffic histograms, the KL distance identifies anomalous time intervals by measuring the similarity between the current traffic histogram and a reference histogram. More formally, given a discrete distribution q and a reference distribution p, KL distance D is defined as follows:

$$D(p||q) = \sum_{i}^{m} p_i log(p_i/q_i). \quad (5)$$

To compare with SENATUS, we apply the KL distance on random projections (hash functions) of traffic histograms at each of the 4-tuple features on each of the measurement intervals. Particularly, the hash function randomly places each traffic feature value into a set of lower-dimensional bins, which represents a *lossless compression* process. In addition, the distribution from the previous time interval is used as the reference distribution p [16]. The KL distance value which exceeds a predefined threshold for any time interval serves to detect anomalies.

After the set of anomalous time bins are identified by the KL distance, root-cause analysis is performed, extracting the set of candidate anomalous flows responsible for the flagged anomalies using the Apriori flow pre-filtering algorithm [16]. This algorithm generates the meta-data that is suspicious to contain the highest amount of anomalous flows. Such flows are further extracted using a frequent item-set mining algorithm (Apriori) proposed in [17].

C. Ground-Truth Construction

Note that for the four traffic traces, there is no ground-truth anomaly data available. In addition, the process of manually inspecting anomalous time intervals, each of which may contain hundreds or thousands of anomalous flows, is an onerous process. To address this challenge, we adopt a combined method, similar to what has been used in the literature when ground-truth data is not available, to construct the ground-truth.

Specifically, we run both SENATUS (using a given heuristic) and the Apriori approach on the dataset traces. For each time bin, if both approaches flag the same anomaly type, then the anomaly is added to the ground-truth. However, if for a particular time bin, only one of them flags an anomaly or each flags a different type of anomaly, we extract the following four tuple features: srcIP, dstIP, srcPort and dstPort for each of the flagged anomalous flows and *do manual inspection* in the following way. We draw a scatter plot per each couple of traffic features in addition to a graphlet of communication pattern [18] and check whether the label suggested by either method matches visual inspection of the graph. If there is a match, the alarm is added to the ground-truth; otherwise, the alarm is considered as a false positive.

D. SENATUS Parameters

In SENATUS, several parameters are associated with its algorithms and heuristics, which are summarized in Table III. In the following, we discuss their value setting in SENATUS.

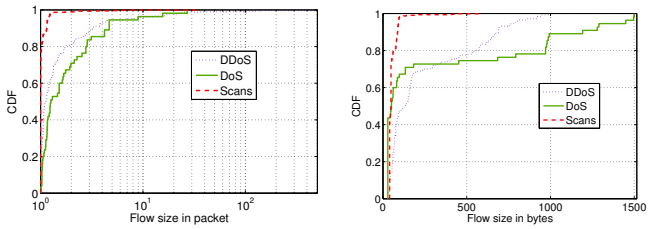
TABLE III. SENATUS PARAMETERS

Parameter	Description	Constraints
α (for H1)	flow size in packets	small
β (for H2)	flow size in bytes	small
K	number of senator	small
λ	PCP weighting parameter	≥ 2
j	flow aggregation level	$[1, 4]$

1) Traffic Filtering Heuristics H1 and H2: As previously discussed, traffic pre-filtering tends to concentrate anomalies in the top-K feature values, thus reducing the number of components required for traffic histogram approximation. Below we study the range of both parameters α and β and explain our choice of their values.

Related to heuristic H1, previous study (e.g. [19]) has reported that most of the encountered anomalies (including scans) in datasets are carried by flows with the number of packets in the range of $[1, 3]$. Authors of [20] also claim that most of the detected scans are carried by flows having a packet count number ≤ 2.

Our investigation of the flow size in the constructed ground-truth is shown in Fig. 1. The figure illustrates the flow size distribution in number of packets for the inspected



(a) Flow size in packets (b) Flow size in bytes

Fig. 1. Flow size distribution per anomaly type

anomaly types in our traces. Fig. 1(a) depicts that anomalies involving flow-size of less than or equal to 3 packets are in the order of 85% of all attacks in our dataset.

Related to heuristic H2, several previous studies have tried to investigate the validity of using H2 as a heuristic to filter out traffic carrying network abuse attacks. Authors of [19] show that most of the detected anomalies (including scans, worms, etc.) in their dataset are carried by small flows having a byte count $\in [40, 144]$. Authors of [21] give narrower range of byte count and show that over 99% of the detected DDoS attacks in CAIDA traces have a packet size falling in the range between 40 and 60 bytes.

Our inspection of the attacks in the constructed ground-truth shows similar results. The flow size distribution in number of bytes is illustrated in Figure 1(b). Although of a long tail due to variable size of the flagged anomalous flows, most frequent DoS/DDoS and scans in our traces are of a small size (≤ 64 *bytes*). For example 52% of the detected DOS and 99% of the detected scans carry flows of size 60 bytes. In the remaining, the threshold values for α and β used in the evaluation are set to respectively 3 and 64.

2) The choice of K: We choose K such that it realizes an average approximation error $\sigma_K \in [0.01, 0.3]$ depending on the measurement trace and the type of traffic under analysis. We assume that the resultant K value under such an approximation error is an acceptable "information-loss" tradeoff.

Fig. 2 illustrates the range of the K value which achieves an average approximation error in the range [0.01, 0.3] per pre-filtering heuristic, for each of the traces. Expectedly, as the approximation error decreases the required number of coefficients for traffic histogram approximation exponentially increases. The figure additionally reports that the value of K can vary from several tens to hundreds in order to achieve a targeted approximation error, depending on the measurement trace and the pre-filtering heuristic.

TABLE IV. APPROXIMATION ERRORS UNDER $K = 20$

Feature	Heuristic	$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$	$\mathcal{D}$
Srcport	H1	0.03	0.02	0.14	0.02
	H2	0.02	0.01	0.12	0.02
Dstport	H1	0.18	0.03	0.18	0.18
	H2	0.06	0.06	0.25	0.25

To avoid complex tuning of the K parameter and motivated with the observation that the value of K is stable over time [4], we choose for simplicity one value of K, i.e. $K = 20$, for all traces under both heuristics. Table IV illustrates the resultant average approximation error for the four measurement traces using each of the proposed heuristics, H1 and H2, under $K = 20$.

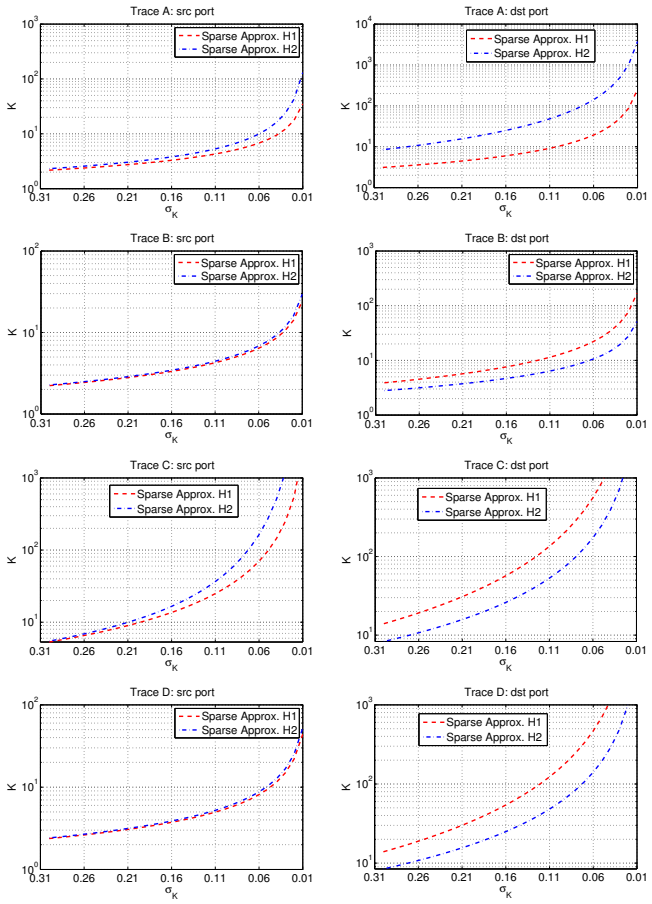


Fig. 2. K value as a function of the approximation error

3) The choice of the PCP Tuning Parameter: Since PCP aims to minimize the weighted combination of the nuclear norm and the ℓ_1-norm, one has to identify an appropriate value of the weight parameter λ such that the matrix A captures the maximum number of anomalies with the least false positive rate. In PCP, parameter λ is also expressed as [5]:

$$\lambda = \frac{C}{\sqrt{\max\{N, K\}}}, C \in \mathbb{R} \qquad (6)$$

where N and K are the dimension parameters of the senator subspace, and C is a constant that needs to be set.

It has been previously shown that $C = 2$ is appropriate for anomaly detection in traffic time series [2]. We base our analysis on this previous observation and tune the parameter λ to find an "optimal" value that achieves the "best" detection-false positive tradeoff. To illustrate this, Fig. 3 is presented, which shows the number of detected anomalies and the false positives as functions of the parameter C. The figure shows that both the detection and false positive rates decrease as the value of C increases. For example, when H2 is used, 113 anomalies are detected with 9 false positives for the value of $C = 2$, while only 57 anomalies are detected with 1 false positive for $C = 2.5$, both in trace $\mathcal{A}$. The figure additionally shows that while the number of false positives, when H1 is chosen, is higher than those when H2 is used, it remains low for all values of C. In the remaining of the evaluation we choose $C = 2$ for both heuristics H1 and H2.

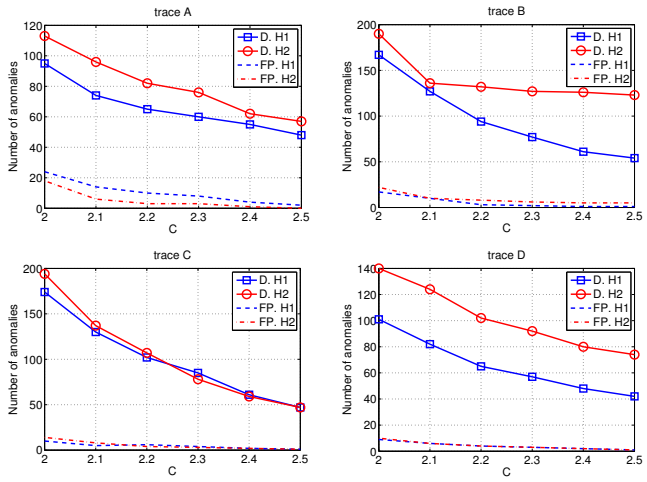


Fig. 3. Detection rate and false positive rate as functions of C

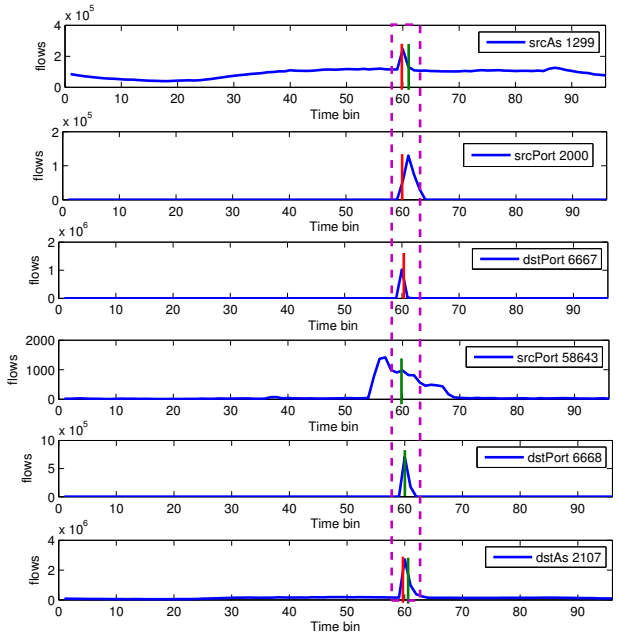


Fig. 4. Anomalous time bin detection

E. Anomalous Time Bin Detection

As an example, Fig. 4 illustrates anomalous time bin detection and suspicious aggregate flows identification based on senators' votes. It shows the flow count time series per *srcAS*, *dstAS*, *srcPort* or *dstPort* senator. These time series were extracted, organized within the senator subspace and processed during the voting stage. As previously discussed, senators' votes consist in the detection of abrupt variations in each senator's time series using PCP.

The figure shows that for every feature, i.e., *srcAS*, *dstAS*, *srcPort* or *dstPort*, PCP detects abrupt variation at time bin 60 in at least one time series of this feature. Based on the chosen decision rule, this time bin is flagged to be anomalous. However, unlike time bin 60, the previous time bin is only flagged with abrupt variation by a srcPort senator (number 58643). In this case the vote from this senator does not trigger the rule which flags the time bin as anomalous.

TABLE V. ANOMALIES FOUND BY EACH APPROACH: SENATUS H1, H2 AND APRIORI (AP)

	Anomaly type	Total	$\mathcal{H}1$	$\mathcal{H}2$	AP
Trace $\mathcal{A}$	DDoS	75	33	10	32
	DoS	12	4	7	1
	Scans	223	58	96	69
	Total	310	95	113	102
	Anomaly type	Total	$\mathcal{H}1$	$\mathcal{H}2$	AP
Trace $\mathcal{B}$	DDoS	76	34	15	27
	DoS	18	8	8	2
	Scans	335	125	167	43
	Total	429	167	190	72
	Anomaly type	Total	$\mathcal{H}1$	$\mathcal{H}2$	AP
Trace $\mathcal{C}$	DDoS	54	16	13	25
	DoS	22	5	3	14
	Scans	374	153	178	43
	Total	450	174	194	82
	Anomaly type	Total	$\mathcal{H}1$	$\mathcal{H}2$	AP
Trace $\mathcal{D}$	DDoS	101	24	18	59
	DoS	10	2	4	4
	Scans	220	75	118	27
	Total	331	101	140	90

F. Detected Anomalies per Type

Table V presents the number of anomalies per type found by each method. The table shows that SENATUS generally detects more anomalies (particularly network scans) than Apriori. However, Apriori detects more DDoS attacks for trace $\mathcal{C}$ and $\mathcal{D}$. We also observe that while SENATUS using H1 or H2 detects more DoS attacks for traces $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{D}$, the situation is reversed for trace $\mathcal{C}$. To understand this, we observed that the missed DDoS attacks are mostly originated with or targeted at a random port number. In this paper, we have adopted the rule $(srcAS \wedge dstAS \wedge srcPort \wedge dstPort)$ in flagging anomalous time bins. This rule could not be best suitable for detecting such anomalies and new rules could be tried, but we leave this for future investigation.

In addition, Table V shows that SENATUS using H1 finds more DDoS attacks than using H2. This is likely due to the fact that only one third (around 29%) of the DDoS attacks found in the collected dataset have packets size less than 64 bytes, as indicated by Fig. 1.

Furthermore, we have observed that SENATUS using H2 detects more network scans than using H1. Most of the additionally detected network scans are small intensity SYN scans using small packets, which are more easily spotted using the second heuristic.

G. Performance Comparison

We now evaluate the tradeoff between detection and false positive rates for SENATUS(H1 $\bigcup$ H2), SENATUS(H1), SENATUS(H2) and Apriori. Here, the detection rate is defined as the ratio of the detected number of anomalies using the method with respect to the total detected number of anomalies using either method. In addition, the false positive rate is defined as the ratio of the number of false positives caused by the method with respect to the number of detected anomalies using this method. For ease of notation, we refer the detection rate of the combined set of anomalies resulting from the union of H1 and H2, i.e. SENATUS(H1 $\bigcup$ H2), as SENATUS' detection rate. As in calculating the false positive rate for SENATUS(H1 $\bigcup$ H2), if any of SENATUS(H1) and SENATUS(H2) flags an anomaly but it is identified by visual inspection as a false positive, then the combined number of false positive for SENATUS(H1 $\bigcup$ H2) increments by one.

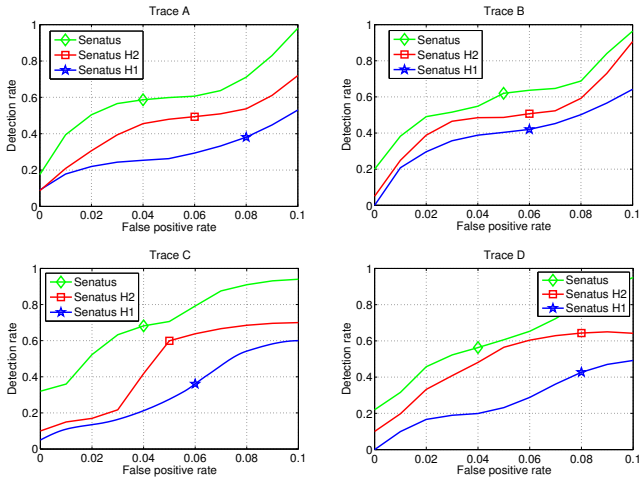


Fig. 5. Receiver Operator Characteristics (ROC) curves

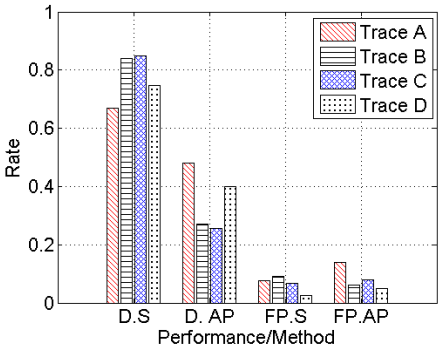


Fig. 6. D.*X*: detection rate of *X*, and FP.*X*: false positive rate of *X*, where *X*= S for SENATUS and *X*= AP for Apriori.

Fig. 5 displays the ROC (Receiver Operator Characteristics) curves that illustrate the detection rate of SENATUS as a function of the false positive rate. The figure firstly shows that while the detection rate of SENATUS using either H1 or H2 is low, the detection rate of SENATUS using the union of the two heuristics is much higher. For example, the detection rate for SENATUS(H2), SENATUS(H1), and SENATUS (H1 $\bigcup$ H2) are around 73%, around 64%, and 84% in trace $\mathcal{B}$. The figure additionally shows that the false positive rate is low for all collected traces: it does not exceed 10% for the four traces.

To illustrate more directly the detection-false positive tradeoff for SENATUS (H1 $\bigcup$ H2) and Apriori, Fig. 6 is presented. Each bar graph shows either the detection or the false positive rate for SENATUS or Apriori. The figure shows that while SENATUS experiences the best detection-false positive tradeoff, Apriori generally exhibits the lowest detection and the highest false positive rate among the three approaches for the four collected traces. For example, for trace D, while the SENATUS' detection rate is about 75% with a false positive rate about 2%, the Apriori's detection rate is around 40% with a false positive rate about 5%.

V. Related work

The problem of network anomaly detection has attracted a lot of research effort. Earlier techniques have mostly relied on volume metrics such as packets and bytes, using time series prediction [22], signal processing [23] machine learning [24] or information entropy techniques [25] to detect abrupt variations in traffic volume signals.

In addition, histogram-based approaches have also been investigated, but they often face the challenge of histogram dimensionality reduction [8][16]. The authors of [8] proposed to keep the well-known source and destination ports, remove the components that remain constant, and additionally apply the PCA technique. In [26], network anomalies are detected via sparsity and low rank property. There, the goal was to construct a map of anomalies in real time, which summarizes the network "health state" along both the flow and time dimensions. Recently, another anomaly detection scheme was proposed in [1], which focuses on the anomaly detection problem for dynamic data streams through the lens of random cut forests. While SENATUS also relies on traffic histograms, it originally resorts to a simple low-complexity lossy compression approach [4] to deal with the curse of dimensionality.

In the literature, several works have tried to conduct root-cause analysis on the traffic anomaly related alarms. Fernandes et al [27] tried to address this using a set of predefined signatures, such as traffic descriptors describing the behavior of network attacks including DoS, DDoS and scans. These predefined signatures involve a large number of empirical threshold values. In addition, authors of [28] proposed a technique for root cause analysis in component-based systems and their approach focuses on application-level anomaly correlation. Differently, SENATUS relies on an algorithm where the threshold values are automatically identified based on the RDT machine learning classification technique [6].

VI. Conclusion

In this paper, we proposed SENATUS, a novel approach for traffic anomaly detection and root-cause analysis, and evaluated it in comparison with the Apriori approach. We found that SENATUS not only uncovers a high number of anomalies but also performs better in diagnosing the root causes. This makes SENATUS an appealing approach. In addition to the novel joint treatment of root cause analysis and anomaly detection, the specific novelty and contribution of SENATUS are: (i) Instead of performing analysis directly on the original traffic histograms, SENATUS uses approximate traffic histograms with much reduced dimensionality as inputs to the analysis. (ii) SENATUS uses PCP, instead of PCA, to detect time bins with abrupt changes. Another *novel idea is to use the detected abrupt variations as votes to collectively flag if a time bin is anomalous*. (3) For root-cause analysis on each anomalous time bin, a simple linear time classification algorithm is used, where the threshold values are decided using the RDT machine learning algorithm.

Acknowledgment

This research was partly supported by the EU FP7 Marie Curie Actions Cleansky Project, Contract No. 607584.

References

[1] S. Guha and et. al., "Robust random cut forest based anomaly detection on streams," in *ICML*, 2016.

[2] A. Abdelkefi and et. al., "Robust traffic anomaly detection with principal component pursuit," in *ACM CONEXT, Student Workshop*, 2010.

[3] F. Silveira and C. Diot, "Urca: Pulling out anomalies by their root causes," in *Proc. IEEE INFOCOM*, 2010.

[4] A. Abdelkefi, Y. Jiang, and X. Dimitropoulos, "K-sparse approximation for traffic histogram dimensionality reduction," in *CNSM*, 2012.

[5] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?," *J. ACM*, vol. 58, no. 3, pp. 11:1–11:37, 2011.

[6] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*. Elsevier, 2011.

[7] M. Molina and et. al., "Operational experiences with anomaly detection in backbone networks," *Computers & Security*, 2012.

[8] A. Kind and et. al., "Histogram-based traffic anomaly detection," *IEEE Transactions on Network and Service Management*, 2009.

[9] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," in *Proc. SIGMETRICS*, 2004.

[10] A. Abdelkefi and Y. Jiang, "A structural analysis of network delay," in *Proc. Conference on Communication Networks and Services Research (CNSR)*, 2011.

[11] H. Ringberg, A. Soule, J. Rexford, and C. Diot, "Sensitivity of pca for traffic anomaly detection," in *Proc. ACM SIGMETRICS*, SIGMETRICS '07, (New York, NY, USA), pp. 109–120, ACM, 2007.

[12] A. Abdelkefi, Y. Jiang, B. Helvik, G. Biczok, and C. A., "Assessing the service quality of an internet path through end-to-end measurement," *Computer Networks*, vol. 70, pp. 30–44, 2014.

[13] W. Fan, H. Wang, P. S. Yu, and S. Ma, "Is random model better? on its accuracy and efficiency," in *Third IEEE International Conference on Data Mining*, pp. 51–58, Nov 2003.

[14] W. Fan, "On the optimality of probability estimation by random decision trees," in *AAAI*, pp. 336–341, AAAI Press, 2004.

[15] A. Lakhina, M. Crovella, and C. Diot, "Characterization of network-wide anomalies in traffic flows," in *Proc. ACM IMC*, 2004.

[16] D. Brauckhoff and et. al., "Anomaly extraction in backbone networks using association rules," in *ACM IMC*, 2009.

[17] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *VLDB*, 1994.

[18] R. Fontugne, Y. Himura, and K. Fukuda, "Evaluation of anomaly detection method based on pattern recognition," *IEICE transactions on communications*, vol. 93, pp. 328–335, 2010.

[19] K. Xu, Z. L. Zhang, and S. Bhattacharyya, "Internet traffic behavior profiling for network security monitoring," *IEEE Transactions on Networking*, vol. 16, no. 6, pp. 1241–1252, 2008.

[20] C. Gates, J. Mcnutt, J. B. Kadane, and M. Kellner, "Detecting scans at the isp level," tech. rep., 2006.

[21] H. Liu, Y. Sun, V. Valgenti, and M. Sik Kim, "Trustguard: A flow-level reputation-based ddos defense system," in *IEEE CCNC*, 2011.

[22] A. Soule, K. Salamatian, and N. Taft, "Combining filtering and statistical methods for anomaly detection," in *Proc. ACM IMC*, 2005.

[23] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in *Workshop on Internet measurment*, 2002.

[24] K. Sequeira and M. Zaki, "Admit: anomaly-based data mining for intrusions," in *Proc. ACM SIGKDD*, 2002.

[25] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "An empirical evaluation of information metrics for low-rate and high-rate ddos attack detection," *Pattern Recognition Letters*, vol. 51, pp. 1–7, 2015.

[26] M. Mardani, G. Mateos, and G. B. Giannakis, "Dynamic anomalography: Tracking network anomalies via sparsity and low rank," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, pp. 50–66, 2013.

[27] G. Fernandes and et. al., "Automated classification of network traffic anomalies," *Security and Privacy in Communication Networks*, 2009.

[28] K. Wang and et. al., "A methodology for root-cause analysis in component based systems," in *IWQoS*, pp. 243–248, 2015.