Detecting VoIP Calls Hidden in Web Traffic

Emanuel P. Freire, Artur Ziviani, *Member, IEEE*, and Ronaldo M. Salles, *Member, IEEE*

***Abstract*—Peer-to-peer (P2P) voice over IP (VoIP) applications (e.g. Skype or Google Talk) commonly use Web TCP ports (80 or 443) as a fallback mechanism to delude restrictive firewalls. This strategy renders this kind of traffic quite difficult to be detected by network managers. To deal with this issue, we propose and evaluate a method to detect VoIP calls hidden in Web traffic. We validate our proposal considering both Skype and Google Talk generated traffic by using real-world experimental data gathered at a commercial Internet Service Provider (ISP) and an academic institution. Our experimental results demonstrate that our proposed method achieves a performance of around 90% detection rate of VoIP calls hidden in Web traffic with a false positive rate of only 2%, whereas a 100% detection rate is achieved with a false positive rate limited to only 5%. We also evaluate the feasibility of applying our proposal in real-time detection scenarios.**

***Index Terms*—Network anomaly detection, Skype, P2P VoIP systems, HTTP traffic.**

I. Introduction

In the last few years, voice over IP (VoIP) applications have faced a huge increase in popularity, in particular those based on the peer-to-peer (P2P) communication paradigm for scalability purposes. In comparison with services from the traditional Public Switched Telephone Networks (PSTNs), which normally use a per-minute charge for long-distance calls, VoIP calls rely on services from the TCP/IP protocol stack provided by the Internet infrastructure, which in turn are usually charged with a fixed flat monthly fee disregarding the transmitted traffic volume. As a consequence, VoIP calls are usually much cheaper than traditional long distance telephone calls to PSTN users, or even free if a call is placed directly from a VoIP end user to another one. Indeed, the combination of VoIP and P2P disruptive technologies has lead to a new disruptive technology [1]. The increasingly popular P2P VoIP applications face a great success based on low to free costs per call allied with the easy of use offered by a dynamic self-adaptive approach to restrictive network environments.

From the network management point of view, an efficient classification of the application protocol responsible for a given traffic is a fundamental issue—and identifying ongoing calls from P2P VoIP applications is no exception. Typically, network operators rely on TCP/UDP port numbers to allow or deny access into their domains, following a list with registered and well known TCP/UDP port numbers provided by the Internet Assigned Numbers Authority (IANA) [2]. Recently, however, the efficient classification of the application protocol responsible for a given traffic has became a more challenging problem, mainly because TCP/UDP port numbers are no longer a reliable information source to identify the application responsible for a given network traffic [3]. There is usually no control to ensure that an application only uses its reserved ports to send or receive data, and with the increasing use of network elements such as firewalls, NAT boxes, and proxies, network applications evolved to operate in different environments with minimal user configuration, e.g. by dynamically choosing a TCP/UDP port number.

In many organizations connected to the Internet, very restrictive firewalls are commonly adopted by network managers in an effort to give a better security to the internal network and optimize the use of network resources. Although being usually very restrictive, such firewalls are unlikely to block Web traffic because it is usually perceived as a fundamental service considered essential for Internet access and the daily use of most modern organizations. As a consequence, HTTP is considered the most popular application protocol in the Internet [4], and thereby it has become usual to find new applications—such as recent P2P file sharing systems [5] as well as media streaming [6] or VoIP systems—using TCP ports 80 (HTTP) or 443 (HTTPS) for delivering non-HTTP traffic, thus fooling restrictive firewalls to gain network access.

Among the P2P VoIP applications that adopt the strategy of disguising their flows as Web traffic to delude firewalls and other network elements, Skype [7] is of particular importance due to its huge popularity [8]. Skype is a very popular VoIP application with a proprietary closed-source protocol and encrypted end-to-end traffic. It is known that Skype can easily work in many different network environments without further user configuration, as it can automatically detect network characteristics and use the best option available to send its traffic. It is also known that Skype can delude a network firewall by using Web ports to establish communication with other Skype peers, as a last resort fallback mechanism in very restrictive environments. Such a strategy renders Skype traffic disguised as Web traffic quite difficult to be detected by network operators. Another VoIP application that is gaining popularity and can automatically use Web ports to send or receive traffic as a fallback mechanism for NAT traversal is Google Talk [9] (see Section III-C for further discussion on Skype and Google Talk characteristics).

In this paper, we tackle the problem of detecting VoIP calls hidden in Web traffic. Traffic from P2P-based VoIP systems is composed of signalling flows and the media flows. The former refers to the traffic to establish and maintain the overlay P2P

Manuscript received February 1, 2008; revised September 24, 2008; approved November 18, 2008. The associate editor coordinating the review of this paper and approving it for publication was N. Anerousis.

E. P. Freire and R. M. Salles are with the Military Institute of Engineering (IME), Praça General Tibúrcio, 80 – 22290-270 – Rio de Janeiro, RJ, Brazil (e-mail: salles@ieee.org).

A. Ziviani is with the National Laboratory for Scientific Computing (LNCC), Av. Getúlio Vargas, 333 – 25651-075 – Petrópolis, RJ, Brazil (e-mail: ziviani@lncc.br).

Digital Object Identifier 10.1109/TNSM.2009.041102

network of the VoIP system as well as to the traffic to signal the call establishment and release. The latter, media flows, refer to the exchange of packets containing voice data of an ongoing VoIP call. In this paper, we focus on detecting the latter kind of traffic, i.e. actual ongoing VoIP calls. Based on the detection of VoIP calls hidden in Web traffic, network administrators may better know the application breakdown as well as evaluate the real demand for such kind of application within the system they manage. In [10], we have investigated a method to automatically detect Skype calls hidden in Web traffic using metrics taken from two Goodness-of-Fit tests, the Kolmogorov-Smirnov distance and the chi-square χ^2 value. This paper extends the results obtained in [10], including the evaluation of the detection method applied to both Skype and Google Talk VoIP calls hidden in Web traffic as well as a complete study on the feasibility of applying our proposed method in real-time detection scenarios, a key feature to many network operators. To the best of our knowledge, ours is the first initiative to address the particular problem of detecting VoIP calls hidden in Web traffic.

Our detection approach consists in building a model of some relevant HTTP parameters and comparing unknown flows with our derived model. In other words, we detect VoIP calls hidden in Web traffic by detecting the presence of flows that differ in a certain number of key characteristics from the expected ("normal") behavior of Web traffic (i.e. by detecting network anomalies in the HTTP flows [11], [12]). One may also try detecting VoIP calls searching for specific program signatures or some known communication pattern, but such an approach is likely to be more dependent of a given program and its version (related work is analyzed in Section II). In contrast, our detection method is fully based on general underlying characteristics of ongoing VoIP calls, such as the regular flow of small packets, that allow distinguishing them from legitimate Web browsing traffic, for instance. Therefore, we consider our detection approach more robust than signature-based ones because it can detect traffic from VoIP calls disregarding payload information and without knowing any specific traffic details.

We evaluate our detection methodology through experiments using real-world data gathered at a commercial Internet Service Provider (ISP) and an academic institution. Our findings show that the $\chi 2$ metric is more likely to achieve better results than the Kolmogorov-Smirnov metric for detecting VoIP calls hidden in Web traffic. Our experimental results also show that the proposed methodology performs well in detecting VoIP calls hidden in Web traffic. Such a good performance may be illustrated in our experimental results of a 90% detection rate of disguised VoIP calls with a false positive rate of only 2%, whereas a 100% detection rate of VoIP calls in Web traffic with a false positive rate limited to only 5%.

We also evaluate the feasibility of applying our proposal in a real-time detection scenario. In this case, the capture time was limited to short time periods (10, 30, and 60 seconds) to verify if our detection results maintain a good performance based on a limited observation time. In the case of a real-time detection of VoIP calls, network managers could choose to perform an immediate action after detection, for example, blocking all traffic identified as VoIP from his network, giving such a traffic a differentiated treatment, or any policy-based management measure a network operator prefers.

The remainder of this paper is organized as follows. Section II briefly discusses related work in the classification of VoIP applications. In Section III, we introduces our proposed methodology to distinguish VoIP calls from Web traffic, based on which we are able to detect VoIP calls hidden in Web traffic. Section IV explains the gathering of experimental real-world data to form our training and evaluation datasets. Section V presents our experimental results evaluating the proposed detection system. Finally, we conclude and discuss future work in Section VI.

II. Related Work

Application identification and classification have been the focus of recent related work [13]–[16]. Nevertheless, to the best of our knowledge, so far no method was specifically designed to deal with the detection of network protocol anomalies—such as the those caused by VoIP calls—in Web traffic.

The presence of VoIP calls hidden in Web traffic may be thought of as a network traffic anomaly. In other words, considering Web browsing traffic as the expected ("normal") kind of flow to be found using HTTP or HTTPS ports, VoIP calls found mixed in the Web traffic aggregate may be seen as an anomalous traffic. In this paper, we also adopt the approach of building a model of the normal behavior for the Web traffic and compare it with the observed behavior of the Web traffic aggregate, thus detecting deviations (i.e. anomalies) caused by the presence of VoIP traffic hidden within it. In order to build the model for Web traffic, we take benefit of a Web workload model developed in a previous work of ours [17], which was based on the particular subject of HTTP workload characterization [18], [19].

Due to the huge popularity achieved by Skype in the last few years, analysis of the Skype protocol and characteristics have been the focus of recent related work. Baset and Schulzrinne [20] present an analysis of the Skype behavior during login, call establishment, firewall/NAT traversal, and other operations. Guha et al. [21] performed a set of experiments to analyze Skype traffic characteristics and better understand its operation. The exchange of Skype signaling information has been recently studied in [22]. Suh et al. [23] monitored the use of relay nodes by Skype traffic in its overlay network using heuristics and statistical analysis to detect Skype relayed traffic. Yu et al. [24] investigate some structure features of the Skype P2P overlay network. Ehlert et al. [25] studied Skype network traffic searching patterns and traffic signatures that can allow Skype to be detected. A comparison between the quality of voice calls provided by Skype and Google Talk is carried out in [26]. Kushman et al. [27] analyze the usability of VoIP applications, such as Skype and Google Talk, for long distance calls and the impact BGP updates exert on the quality of the VoIP calls. In [28], authors present a Skype traffic characterization. Skype reactions to changes in network conditions, Skype signaling traffic, and some aspects of Skype flows derived from real Skype calls were also analyzed.

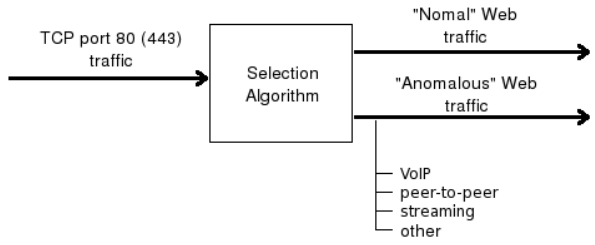

Fig. 1. Distinguishing between normal and anomalous Web traffic.

The recent work by Bonfiglio et al. [29] is the closest to ours and presents approaches to identify Skype traffic in TCP or UDP flows. Their method uses some Skype-specific information for UDP detection and it uses the Pearson's chi-square (χ^2) test to verify if the payload data appears to be random, using this feature as a trigger for UDP or TCP Skype detection. In contrast, we use the χ^2 value to compare parameters of flows carrying VoIP calls with empirical data derived from real Web flows. Our methodology was specifically designed to deal with the HTTP protocol to distinguish ongoing VoIP calls hidden in Web traffic. We believe this methodology can be eventually extended to detect other applications that may be using HTTP ports as well (see Section VI for a discussion on future work).

In short, contrasting with related work, our proposed detection system is able to distinguish ongoing VoIP calls hidden in Web traffic without searching for application signatures in TCP/IP packets and disregarding payload information (experimental results are detailed in Section V). We believe this is a significant contribution to network monitoring as our relatively simple methodology presents promising results in detecting VoIP calls (as those generated by Skype or GoogleTalk) hidden in Web traffic, a key challenge that has not been addressed in the literature so far.

III. Proposed Methodology

The detection process can be subdivided in two steps. First, we define a HTTP workload model and capture real Web data to build empirical distributions of some relevant parameters. We then capture Web traffic with VoIP calls hidden in it, calculate the same relevant parameters for each flow and use metrics taken from two Goodness-of-fit tests to decide whether the computed parameters are compatible (or not) with the empirical distributions derived in the previous step, classifying each flow as legitimate Web traffic or not. Figure 1 illustrates this methodology in the general case of distinguishing "normal" Web traffic from "anomalous" traffic hidden in the Web traffic aggregate.

In this paper, we focus on the case of detecting anomalous traffic within the Web traffic aggregate caused by VoIP call from a P2P VoIP application such as Skype or Google Talk. Along this section, we briefly review the considered HTTP workload model for Web traffic and the adopted statistical tests. We also present an overview of the Skype program and its main operation characteristics that play a key role in allowing its detection within the Web traffic aggregate.

A. HTTP Workload Model

A first step towards HTTP traffic characterization is the selection of some HTTP relevant parameters. We are interested in finding VoIP calls hidden among Web flows. Since we avoid relying on program signatures or patterns that can be easily changed, we must define a model to evaluate Web "normal" behavior. In this paper, we build upon the model defined in a previous work of ours [17], which was based on the particular subject of HTTP workload characterization [18], [19]. Contrasting with related work in the characterization of a HTTP workload model, our previously proposed workload model was not designed to be used in simulations or traffic generators, but to distinguish anomalous flows within Web traffic, in particular those generated by ongoing VoIP calls to distinguish VoIP calls from legitimate Web traffic. This model has the following parameters:

- Web request size;
- Web response size;
- Interarrival time between requests;
- Number of requests per page;
- Page retrieval time.

The Web request size is the size in bytes of the HTTP request message. The Web response size is the size in bytes of the HTTP response, sent by some Web server. The time interval between two consecutive requests of the same client for the same Web server is the interarrival time between requests, if these requests are close enough to be considered parts of the same Web page. A Web page may have one or many requests and it is important to identify page boundaries. Based on this information, we can compute parameters such as the number of requests per page, page retrieval times, and request interarrival times. The number of requests per page is the number of HTTP request messages in the same Web page and the page retrieval time is the time elapsed from the first request to the last response. These parameters have been chosen because they are the most discriminant ones [17] to distinguish VoIP calls from legitimate Web traffic. The workload characterization of our experimental datasets based on these parameters is presented in Section IV-A and also serves as an illustration of the applicability of this methodology.

B. Goodness-of-fit tests

In the case where we do not know the underlying distribution of some population, we can use a goodness-of-fit measure to test if a particular distribution can be satisfactory as a population model. The chi-square test had already been used for anomaly-based intrusion detection [30], or to verify the presence of random payloads in Skype detection [29]. The Kolmogorov-Smirnov (K-S) test was one of the tests used for anomaly detection in [31]. Nevertheless, in our work, we do not use these tests as goodness-of-fit tests to accept or reject an initial hypothesis with a given significance level based on some known distribution. We use the chi-square χ^2 and the Kolmogorov-Smirnov D values to detect VoIP calls by directly comparing the calculated χ^2 and D values with given thresholds to decide whether some flow present in the Web traffic is likely to be a legitimate Web flow or not. This solution provides more simplicity and flexibility to our

proposal because we only need to change the threshold values to get a more loose classification or a more conservative one. In other words, we are able to provide a simple tuning knob for network operators to configure the level of sensitivity in the detection of the VoIP calls at the expense of taking some risks of also enhancing the false positive rate, as it will be further analyzed in Section V. In the following, we briefly review the adopted metrics.

1) Chi-square test: The chi-square (χ^2) goodness-of-fit test, was first investigated by Karl Pearson in 1900 [32]. Basically, it tests a null hypothesis that the observed frequencies of some independent events follow a specified distribution. Suppose we have n observations from a population classified into k mutually exclusive classes and there is some theory or hypothesis which says that an observation falls into class i with probability p_i $(i = 1, \ldots, k)$, so, the number of events expected in class i is $E_i = np_i$. If O_i is the number of events observed in class i, the chi-square statistic χ^2 is the sum over all bins as given by

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}. \quad (1)$$

A large value of the sum indicates that is rather unlikely that the O_i values are drawn from the population represented by the E_i.

2) Kolmogorov-Smirnov test: The Kolmogorov-Smirnov test [33] also evaluates if a sample comes from a population with a hypothesized distribution. It is based on the maximum difference between two cumulative distributions, $F_0(x)$ and $S_N(x)$. $F_0(x)$ is some specific cumulative frequency distribution function—in our case, the empirical distribution function derived from the training Web dataset IV-A. $S_N(x)$ is the cumulative step function of a sample of N observations or, in other words, $S_N(x) = c/N$ where c is the number of observations with a value less than x. The Kolmogorov-Smirnov distance (D) value is given by

$$D = \max\big(|S_N(x) - F_0(x)|\big). \quad (2)$$

This value is limited between 0 and 1. A value near zero indicates that $F_0(x)$ is very similar to $S_N(x)$.

C. P2P VoIP characteristics

We are interested in detecting actual VoIP calls hidden in Web traffic originated by P2P VoIP applications, such as Skype or Google Talk. In this subsection, we review the basic characteristics of these P2P VoIP systems.

1) Skype background: Skype [7] adopts a proprietary protocol to perform peer-to-peer communication among its users. It does not use SIP (Session Initiation Protocol) [34], or other known signaling protocol for VoIP calls, and all its VoIP traffic is end-to-end encrypted. Skype has the ability to automatically detect network characteristics and choose the best option available to communicate with other Skype peers. As shown in papers characterizing Skype operations [20], [25], [35], Skype only uses Web ports as a last resort fallback mechanism, typically when UDP is not available—a rather usual scenario when operating behind restrictive firewalls. In adopting this strategy, Skype can successfully work behind many restrictive firewalls or proxies without further user configuration, leading to an easy of use that is crucial for being massively adopted by the general public.

Skype is also known to regularly generate network traffic when the program is running but not being used, as it needs to keep periodically in contact with the its P2P overlay network and it may relay traffic from other Skype hosts [22], [23]. Any Skype host with sufficient resources might automatically start relaying traffic from other Skype users, but this apparently does not happen if operating behind restrictive firewalls. Skype also generates signaling traffic to verify if its peers are still active and for other operations. As indicated in [22], [28], the vast majority of Skype signaling flows are single packet probes and there exists some persistent signaling activity between Skype nodes. When Skype uses TCP, it also keeps the TCP connection open for some time after the end of call. This signalling traffic does not represent ongoing VoIP calls and ideally should not be identified as Skype in our tests as our goal is to detect actual VoIP calls.

2) Google Talk background: Google Talk [9] refers to the client provided by Google to access its instant messaging and VoIP P2P-based service. Contrary to Skype that relies on proprietary protocols to perform communication among its peers, Google Talk is largely based on open standards for communication. Indeed, Google Talk is based on the open-standard Extensible Messaging and Presence Protocol (XMPP) [36], an open, XML-based protocol originally aimed at extensible instant messaging (IM) and presence information (a.k.a. buddy lists). For P2P signalling to establish multimedia interactions such as voice or video, Google Talk uses Jingle [37], an extension to the XMPP protocol for that purpose. One known feature of Google Talk is its ability to perform NAT traversal and make VoIP calls behind restrictive firewalls using Web ports, just like Skype.

D. Detecting VoIP calls in Web traffic

Our proposed detection process is based on the characterization of the five parameters defined in Section III-A for each flow in the Web traffic aggregate. The number of requests per page and the page retrieval time have a single value in each Web page and are somewhat correlated, so we used the number of requests per page as a filter to remove smaller flows. In fact, a flow corresponding to a VoIP call typically persists for at least some seconds, so it is quite likely to present many "requests" in a single flow.

The distributions of the other three parameters—namely the Web request sizes, the Web response sizes, and the interarrival times of Web requests—are used in Equations (1) and (2) to generate a χ^2 or a Kolmogorov-Smirnov D score. Such a score represents how much the observed distribution of each parameter in a given flow (present in the Web traffic aggregate) deviates from the expected distributions for normal Web browsing traffic. Each parameter generates a score, so in order to make a classification, we have three values that can be compared with thresholds to define if this set of related request-response messages is likely to be originated by a legitimate Web browsing experience or by the traffic of a

TABLE I
WEB TRAINING DATASETS.

Trace	Period	Average daily volume
ISP-T1	18-20 Jun 2007	45 GB
ACD-T1	14-17 Aug 2007	35 GB
ISP-T2	14-16 Aug 2007	50 GB
ISP-T3	18-20 Oct 2007	60 GB

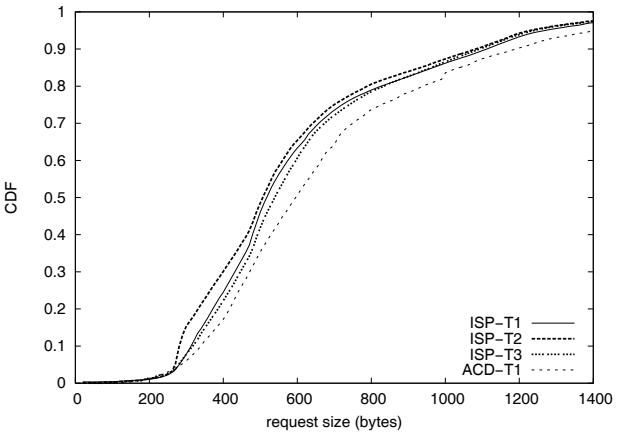

Fig. 2. Sizes of the Web requests.

disguised VoIP call. The characterization of the normal Web traffic using these parameters is presented in Section IV-A when discussing the gathering of experimental data to form our Web training dataset. For classifying the entire flow, the results obtained for each flow component are combined. If the majority of flow components are identified as a VoIP call, the flow itself is classified as traffic from a VoIP call. The evaluation of the best combination of these parameters is performed in Section V-A.

In short, our detection methodology takes benefit of some of the underlying characteristics of P2P VoIP operation during an actual VoIP call discussed in Section III-C to distinguish it from normal Web browsing traffic. As we show in our experimental results discussed in Section V, the same proposed detection methodology presents similar performance results for Skype and Google Talk traffic as well. This suggests that both P2P VoIP applications may share common underlying characteristics, at least similar enough between them to be clearly differentiated from the normal Web browsing traffic and trigger our detection system of VoIP calls.

IV. EXPERIMENTAL DATASETS

The first step of the detection process is to adopt a representative training dataset to characterize a "normal" Web traffic behavior. In Section IV-A, we discuss the gathering of experimental data to form our Web training dataset and we characterize it using parameters from the HTTP workload model presented in Section III-A. The second step is the detection itself. We discuss in Section IV-B the gathering of the adopted experimental datasets to evaluate the detection efficiency of our proposed methodology.

A. Training Datasets

In order to form the Web training dataset, we capture HTTP full packet traces using `tcpdump` [38]. We have developed a software based on `tcpflow` [39] to read these `tcpdump` trace files and compute the parameters of the our considered Web workload model as defined in Section III-A. `tcpflow` is a GPL software that can read `tcpdump` captured data and separate each flow present in it. Our software works only with Web traces; it can separate each flow present in a trace file, define boundaries of Web pages for each flow, and calculate the parameters of our HTTP workload model for each Web page. In this context, a Web page is considered as the complete set of one or more objects in a Web document, normally a HTML file and some figures, for instance. In this step, while searching for boundaries of Web pages, we inspect HTTP headers to clearly identify a Web request or a Web response as well as to compute the inactivity time between Web messages. We must also assure that our training dataset is composed only of Web browsing traffic (i.e. HTTP-related messages), so we made a full packet capture to filter all non-HTTP data. After the calculations for all Web flows present in the trace file, the results are combined and we build empirical distributions to be used in our statistical tests.

We used two types of real-world packet traces, one gathered at a commercial Internet service provider (ISP) and the other originated from an academic institution (ACD). Information about our training traces is shown in Table I. Traces ISP-T1, ISP-T2, and ISP-T3 were captured at the same ISP, located in Niterói, Brazil, with a 2-months interval between each capture. At the time of our captures, the ISP had approximately 3 thousand active clients. Trace ACD-T1 was captured from the main Internet link of an academic institution located in Petrópolis, Brazil, with approximately 3 hundred users.

The average daily volume in Table I represents the amount of traffic on TCP port 80 captured divided by the number of days of each data gathering experiment. The volume of traffic in each monitored link varies depending on the hour of the observation. In order to minimize the contribution of these traffic fluctuations, all ISP traces had exactly 2 days (48 hours) of capture. With the same objective, the ACD trace had 72 hours of capture.

Figure 2 presents the cumulative distribution function (CDF) of the sizes of the Web request messages for all traces. We observe that all ISP traces have a very close result, whereas the ACD-T1 trace presents slightly larger Web requests. For each trace, the volume of Web requests larger than 1,500 bytes was less than 0.8% of all observed requests. In Figure 3, we show the CDF of the observed sizes of the Web response messages for all traces. Again, all ISP traces are very similar and the ACD-T1 trace presents slightly larger Web responses before the 40,000 bytes mark. In all traces, however, the number of Web response transfers larger than 100,000 bytes was rather insignificant (less than 1%), but they represent around 30 to 40% of the total traffic volume for Web responses.

Figure 4 shows the CDF for the interarrival times between Web requests. This represents the time interval between two consecutive requests for the same Web page, therefore this is a parameter for Web pages with two or more requests. As the number of requests in a page increases, more values are generated for these three parameters and there will be more terms in test Equations (1) and (2). We assume that a

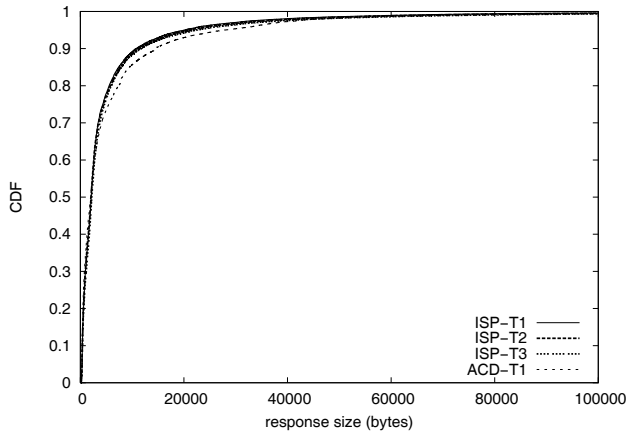

Fig. 3. Sizes of the Web responses.

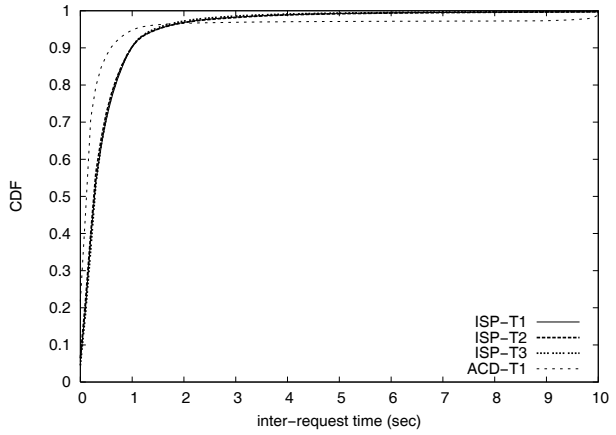

Fig. 4. Interarrival times between requests of the same Web page.

larger number of terms in the test equations produces more accurate results, so this is another reason to filter flows with few requests per page.

From the similar results observed for traces ISP-T1, ISP-T2, and ISP-T3, captured with a two months interval between each of them, we may assume that the training dataset generated remains valid for all this time period. There are also some differences between HTTP versions 1.0 and 1.1. The number of HTTP/1.0 messages was less than 5% of the total messages in all traces. Therefore, all analysis were based only on HTTP/1.1 data and the HTTP/1.0 messages were discarded.

B. Evaluation Datasets

As in the gathering of the training dataset, we captured Web packet traces using `tcpdump` to build our evaluation datasets, but for the latter we only capture the TCP/IP headers. We developed another software to read the `tcpdump` trace files and compute each model parameter. This software is different from the program used for the training dataset in Section IV-A because here the calculations and the classification of flows into Web pages are done without examining TCP payload (i.e. the HTTP headers). Working with such a constraint is important as our proposed detection methodology should not rely on payload information, thus not depending on particular application-level signatures. The procedure for defining the sizes of the Web messages is to consider every MTU-sized packet as a part of the same Web message, if there is not too much inactive time between them. The procedure for defining the boundaries of the Web pages is also based on the inactivity time with a fixed threshold.

TABLE II
EVALUATION DATASETS.

Trace	Date	Duration	Number of VoIP flows
ISP-D1	23 Jul 2007	8h	80
ISP-D2	22/23 Aug 2007	16h	85
ISP-D3	24/25 Oct 2007	14h	115

In order to evaluate our detection methodology, we captured the evaluation datasets shown in Table II. For traces ISP-D1 and ISP-D2, VoIP calls of different durations were produced in a controlled way by a small network of computers behind port-restrictive firewalls running the Skype program. In these experiments, we used Skype versions 1.3 and 1.4 for Linux and version 3.5 for Windows. For trace ISP-D3, both Skype and Google Talk were used to generate VoIP traffic. Skype and Google Talk work in a similar way; they only use Web TCP ports when UDP is not available. The Google Talk version able to make VoIP calls is only available for Windows platforms. Our collaborative ISP provides valid dynamic IP addresses for its clients, and there are no closed ports or firewall restrictions in their way to the Internet. As a consequence, we may suppose there is no VoIP traffic in these Web traces other than our controlled VoIP calls, since the considered P2P VoIP applications—Skype and Google Talk—only use Web ports as a fallback mechanism.

V. EXPERIMENTAL RESULTS

In order to present our results, we use ROC curves [40]. ROC is the acronym for receiver operating characteristic, a graphical representation of the efficiency measured for a given binary classifier (i.e. detection system). The ROC curve of a detection system shows its performance as a trade-off between the selectivity and the sensitivity of the analyzed system. In other words, given a detection system, a ROC indicates how many false positives one must tolerate to be guaranteed a certain percentage of true positives. The ROC curve is thus obtained plotting the false positive rate versus true positive rate while a sensitivity or threshold parameter is varied. The true positive rate is estimated as the number of positive events correctly classified over the number of total positive events and the false positive rate is estimated as the number of negatives incorrectly classified over the number of total negatives. The ideal performance of a detection system is achieved when all true positives are found (true positive rate is 1) with no false positives (false positive rate is 0). The performance curve of such an ideal detection system would be represented by a single point in the top-left corner (true positive rate equals 1 while the false positive rate equals 0) of the ROC plot. Therefore, the closer the performance of a detection system is to the top-left corner of a ROC curve, the better it is.

The empirical distributions shown in Section IV-A are now compared with data generated from each individual flow. In our evaluation datasets (ISP-D1, ISP-D2 and ISP-D3), we

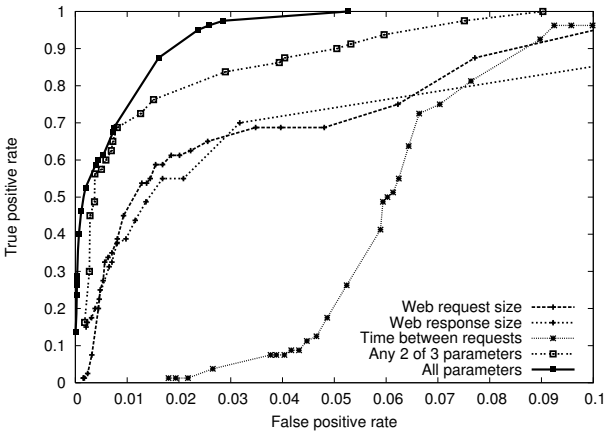

Fig. 5. Evaluating the selection of model parameters using χ^2.

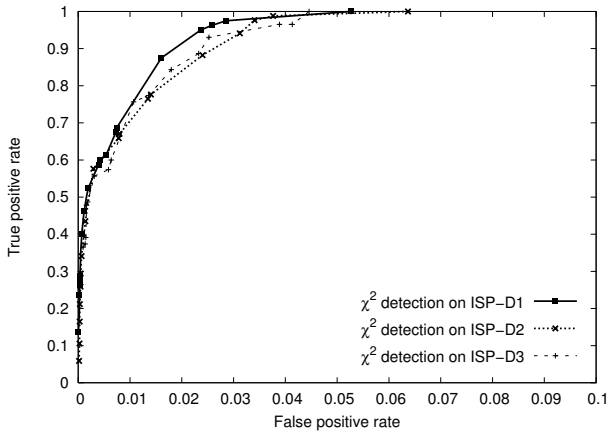

Fig. 6. ROC Curves for χ^2 detection.

TABLE III
TRUE POSITIVE RATE (TPR) AND FALSE POSITIVE RATE (FPR) OBTAINED WITH THRESHOLDS 1, 2, 3 FOR TRACE ISP-3 AND χ^2 DETECTION.

Point	Thr. 1	Thr. 2	Thr. 3	TPR	FPR
1	3,000,000	100,000	5,000	0.1375	0.0001
2	3,000,000	50,000	5,000	0.2375	0.0002
3	2,000,000	50,000	5,000	0.2625	0.0003
4	2,000,000	50,000	2,000	0.2750	0.0003
5	1,500,000	50,000	2,000	0.2875	0.0004
6	1,500,000	25,000	1,000	0.4000	0.0007
7	1,500,000	10,000	1,000	0.4625	0.0011
8	1,000,000	10,000	1,000	0.5250	0.0019
9	500,000	8,000	800	0.5875	0.0039
10	500,000	5,000	750	0.6000	0.0043
11	250,000	5,000	750	0.6125	0.0054
12	100,000	5,000	750	0.6750	0.0073
13	100,000	4,000	500	0.6875	0.0075
14	20,000	1,000	500	0.8750	0.0161
15	10,000	1,000	500	0.9500	0.0236
16	10,000	1,000	250	0.9625	0.0258
17	10,000	500	200	0.9750	0.0285
18	5,000	200	100	1.0000	0.0527

captured TCP/IP headers from ports 80 and 443 and manually identified all flows generated by our VoIP calls to serve as a reference dataset. The comparison between the reference dataset and the output of our method produces a point in our ROC curve, thereby defining a true positive rate and a false positive rate associated with a given threshold value that represents how sensitive our detection system is for VoIP calls hidden in Web traffic.

A. Selecting the best model parameter combination

At first, we used trace ISP-D1 to find out how the three main discriminant parameters of our HTTP model (size of Web requests, size of Web responses, and interarrival times between Web requests) can be used to produce the best results in our detection method. As shown in Figure 5, we compare the detection results when each parameter is considered individually and when they are jointly considered. In this latter case, we analyzed a detection criterion based only on two parameters, a detection based on any two of the three parameters, a detection based on metric χ^2 and K-S as well as a detection decision that requires a simultaneous positive classification from all the three parameters (all parameters plot). For the sake of clarity, some evaluated combinations do not appear in Figure 5, as they were all less accurate in comparison with the χ^2 detection using all parameters in a joint way.

Before the calculation of the set of points that will represent a ROC curve, we must define a set of threshold values to be used. When only one parameter is considered, the set of threshold values is a simple ordered list of positive values for χ^2 detection and an ordered list of values between 0 and 1 for K-S detection. In the case when two or more parameters are present, we have multiple lists to be combined. The final set of threshold values is obtained after a process with many iterations, by keeping the values that produce the best results and generating new ones for the combined list. In the final sequence, the set of threshold values is represented by two or more ordered lists. In Table III we have the optimal values obtained for trace ISP-D1 and χ^2 detection when the simultaneous triggering of all thresholds are required for a positive classification. Thresholds 1, 2 and 3 are χ^2 comparative values for request size, response size, and inter-request time, respectively. Each line represents a point in the curve shown in Figure 5 obtained with all parameters. As the threshold values decrease, the true positive rate grows, but the false positive rate also increases. Based on this preliminary analysis, the combination of all three parameters is used in χ^2 and K-S detections as it yields the most efficient detection result when considering the trade-off between true positive rate and false positive rate (i.e. it provides the closest result in the ROC curve to the ideal result of true positive rate 1 and false positive rate 0).

B. Performance of the VoIP call detection method

The ROC curves for χ^2 detection on the traces presented in Table II are shown in Figure 6. The parameters were calculated only for flows with more than 20 requests, performing a total of 17,374 flows in trace ISP-D1 and 24,662 flows in trace ISP-D2. We observe that results for all datasets are quite similar, although results for trace ISP-D1 were slightly better. This suggests that the performance of the proposed method is consistent over different datasets with experimental data collected with a 2-months interval between each of them. Note that the proposed methodology achieves a performance of around 90% of the VoIP calls correctly identified (i.e. true

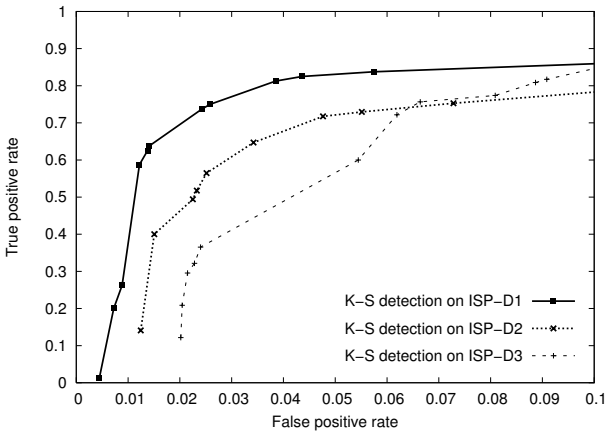

Fig. 7. ROC Curves for Kolmogorov-Smirnov D detection.

positive rate) with less than 2% of the total number of flows incorrectly identified as a VoIP call (i.e. false positive rate). Likewise, our experimental results also show a 100% detection rate with around 5% of false positives.

The results for the K-S detection are presented in Figure 7. For the ISP-D1 trace, we can achieve a true positive rate of 70% with a false positive rate around 2% or a 80% detection with 5% of false positives. Comparing Figure 6 and Figure 7, we observe that the K-S results are not so good as the χ^2 detection because the points over the χ^2 ROC curve are always closer to the top-left corner in comparison with the K-S curve.

Our experimental results thus suggest that using our methodology the χ^2 detection is a better choice than the K-S detection to efficiently detect VoIP flows hidden in Web traffic.

Considering the experimental results for the χ^2 detection shown in Figure 6, our methodology provides enough flexibility for a network manager to adopt different approaches regarding the possible detection of VoIP calls hidden in Web traffic. On the one hand, if one wants a VoIP detection system with few false positive errors (i.e. a conservative approach), one may choose the thresholds used to generate a point closer to the Y-axis at the expense of a reduced detection rate. On the other hand, if one wants to capture almost all true positives (i.e. a loose classification), one may choose the thresholds used in a point near the top axis at the expense of tolerating a higher false positive rate.

C. On the feasibility of real-time detection

The results obtained in Section V-B were all based on an offline analysis of the captured data. However, a network administrator may want to identify the VoIP calls that are currently using the network, not the calls made some minutes or hours ago. In this section, we evaluate the performance of our detection methodology taken into account time constraints on the traffic observation by the proposed system. A simple evaluation of the feasibility of real-time detection may be performed by limiting the capture time. The methodology proposed for this evaluation relies on the same training data (see Section IV-A), but essentially differs in the detection part: here the data is captured and analyzed using limited short time intervals.

TABLE IV
GENERATED DATASETS FOR REAL-TIME DETECTION EVALUATION.

Dataset name	Source	Content
RT1-10	ISP-D1 and ISP-D2	125 files of 10s
RT1-30	ISP-D1 and ISP-D2	125 files of 30s
RT1-60	ISP-D1 and ISP-D2	125 files of 60s
RT2-10	ISP-D3	114 files of 10s
RT2-30	ISP-D3	114 files of 30s
RT2-60	ISP-D3	80 files of 60s

The main goal here is thus to analyze the outcome of the detection methodology when flows may be observed only for a limited time period, thus evaluating the feasibility of real-time detection. In this case, if a reasonable performance is achieved, a classification can be done while the observed flow is still active. As we focus on detecting ongoing VoIP calls for classification purposes, we chose 10 seconds as a reasonable time interval between updates for our detection tool. Practical VoIP calls are usually much longer than that [41], [42], then network management systems (or network operators) can wait that long for receiving updated information and still have time to perform some action with the detected ongoing VoIP calls, if needed. We also analyze the results obtained with time intervals of 30 and 60 seconds for comparison purposes.

We generate two new test datasets (RT1 and RT2) to evaluate a real-time detection with time periods of 10, 30, and 60 seconds allowed for traffic observation. We extract from the evaluation traces ISP-D1 and ISP-D2 a total of 125 capture files with a chosen duration. These 125 capture files were not contiguous, but rather separated by several time intervals in order to get different VoIP calls in each capture file, forming the RT1-10 dataset. After the generation of these files, we manually identified all VoIP calls present in them to serve as a reference for our detection methodology, counting 115 VoIP calls. We launched the χ^2 detection tool used in the previous section to generate a new ROC curve. The K-S detection was not used because the results obtained in the previous section were not so good as the χ^2 results. A small modification was made in our detection program to use a lower limit for the number of requests present in a flow. Given the small size of capture files, we calculated parameters for every flow with more than 10 requests. The χ^2 ISP-D1 detection curve was recalculated after this modification for comparison with RT1-10. Applying similar procedures, we derived the datasets with limited time periods of observation presented in Table IV.

We observe in Figure 8 that the χ^2 detection using the generated trace RT1-10 (with 10s capture files) has a true positive rate up to 85% with a smaller number of false positives compared with the χ^2 detection using the ISP-D1 trace. Beyond this point, however, the number of false positives grows significantly, and the χ^2 ISP-D1 combination presents more accurate results than the χ^2 detection on RT1-10. The time needed to analyze each captured file was insignificant compared to the 10s interval using a standard desktop computer for this job. This result suggests that this approach with a 10 seconds time bin is sufficient for detecting most VoIP calls, but we should expect some VoIP calls to be not distinguishable from Web flows under this relatively tight time constraint. With a larger time bin, however, the curve is

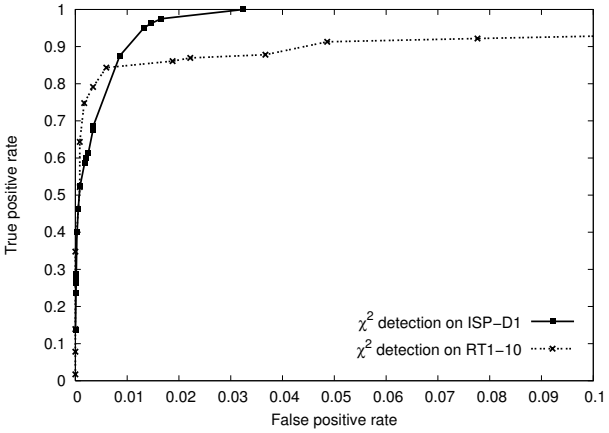

Fig. 8. Comparison between the χ^2 detection on RT1-10 and ISP-D1.

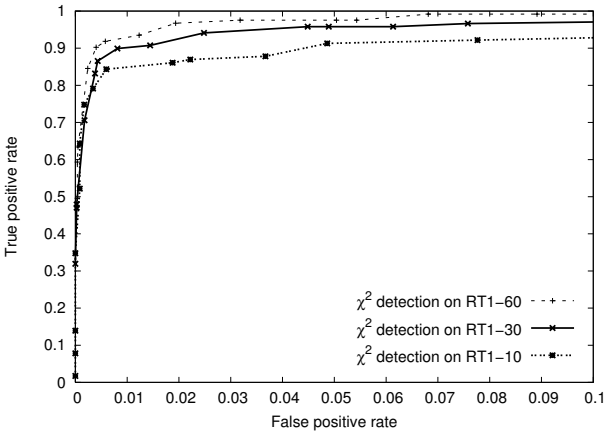

Fig. 9. Results for χ^2 detection on RT1-10, RT1-30, and RT1-60.

expected to approach the χ^2 results for ISP-D1 or ISP-D2 with no time constraints, given that the RT1-10 trace is derived from both. The results for χ^2 detection on RT1-30 and RT1-60 in Figure 9 confirm this expectation—with a larger capture time for traffic observation, our proposed method achieves a better result, i.e., a higher true positive rate for the same false positive rate. Results in Figure 9 also show that using a 60 seconds capture time (RT1-60 dataset), our proposed methodology is able to achieve true positive rate of more than 95% with less than 5% of false positive rate. A capture time larger than 60 seconds was not evaluated because it was not considered feasible for detecting a real-time event.

Figure 10 shows the results obtained with datasets RT2-10, RT2-30, and RT2-60. We recall these derive from the ISP-D3 dataset that includes both Skype and Google Talk VoIP calls. There is no clear distinction between the RT2-30 and RT2-60 curves, and the detection on RT2-10 was slightly inferior. Nevertheless, the key outcome is that, with a 10 seconds observation time, we can achieve a true positive rate of 90% with a false positive rate around only 2.5%. The results observed in Figure 9 and Figure 10 strongly suggest that using our proposed methodology a real-time detection of VoIP calls hidden in Web traffic is quite feasible with results reasonably close to those found in the offline analysis presented in Section V-B.

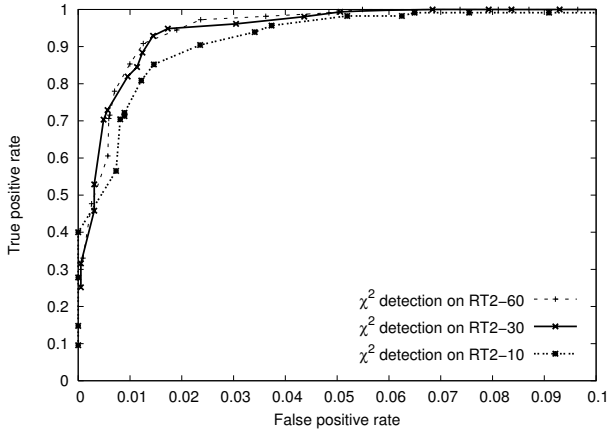

Fig. 10. Results for χ^2 detection on RT2-10, RT2-30, and RT2-60.

VI. Conclusions

VoIP applications based on P2P networks have became very popular in recent years. The main causes of their success are the lower costs of calls (as compared to PSTNs) and their high capacity for adaptive operation in restrictive network environments in a transparent way for the end user. Skype and Google Talk are typical examples of these P2P VoIP applications capable of operating behind restrictive firewalls, proxies, or other network elements. At the same time, Web browsing is a "must-have" service for many institutions and enterprises connected to the Internet. Therefore, it is rather common to find non-HTTP traffic using Web ports to fool firewalls and other network elements.

In this paper, we proposed and evaluated a detection system based on a HTTP workload model to efficiently detect VoIP calls hidden in Web traffic. Our experimental results are derived using real-world data gathered at a commercial Internet Service Provider (ISP) and an Academic Institution (ACD) at different points in time with months of interval to assure a diverse collection of datasets for training and evaluation. Important features of the proposed detection system include its capacity of detecting VoIP calls hidden in Web traffic without searching for particular application signatures and disregarding payload information (so not raising privacy issues).

Based on a training experimental dataset, we characterized real Web flows to build empirical distributions to represent the "normal" behavior of Web traffic. We produced VoIP calls in a controlled way using both Skype and Google Talk to build evaluation datasets to verify that the proposed methodology is able to efficiently pinpoint VoIP calls hidden in Web traffic from two different popular P2P VoIP applications. Using metrics taken from two Goodness-of-Fit tests, the χ^2 value and the Kolmogorov-Smirnov distance, we show that P2P VoIP calls can be clearly detected, but our results suggests that the χ^2 detection is a much better choice.

Our experimental results demonstrate that the proposed method achieves a performance of around 90% detection rate of VoIP calls with a false positive rate of only 2%, whereas a 100% detection rate of VoIP calls hidden in Web traffic is achieved with a false positive rate limited to only 5%. The results are similar for both Skype and Google Talk

hidden flows. Our results may be dependent on the adopted training dataset. The presented workload characterization for the training dataset, however, indicates that the same set of empirical distributions remains similar for several weeks, thus suggesting it is quite representative. We also evaluate the feasibility of applying our proposal in real-time detection scenarios.

As future work, we intend to build and evaluate an optimized version of our tool to perform real-time monitoring in network links. The proposed HTTP workload model can also be seen as a building block to the development of an automatic detection system of other kinds of non-HTTP flows hidden in Web traffic, such as P2P file sharing and media streaming applications. Clearly, to achieve this, further investigations are needed to identify the proper parameters in the HTTP workload model to detect each target disguised application. The key point to notice is that the proposed framework is general; it is not restricted to Web (non-Web) traffic detection. Instead, VoIP traffic could be used as a reference in the framework and so the scheme would be able to detect non-VoIP traffic as well. In this sense, a third type of traffic using Web ports (disguised P2P file sharing, gaming data, an so on) could be identified as non-Web and non-VoIP. Therefore, in future work, we plan to investigate this possible generalization of our current detection method.

ACKNOWLEDGMENT

The authors would like to thank the collaboration of Antônio Tadeu Gomes, Marcos Gomes Pinto Ferreira, and Marcos Vinícius do Couto that made it feasible to capture the Web traffic used in our experimental datasets. This work was partially supported by CNPq and FAPERJ.

REFERENCES

[1] B. Rao, B. Angelov, and O. Nov, "Fusion of disruptive technologies: lessons from the Skype case," *European Management J.*, vol. 24, no. 2-3, pp. 174–188, 2006.

[2] IANA, "Port numbers," http://www.iana.org/assignments/port-numbers.

[3] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "Blinc: multilevel traffic classification in the dark," in *SIGCOMM'05: Proc. 2005 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, 2005, pp. 229–240.

[4] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-level traffic measurements from the Sprint IP backbone," *IEEE Network*, vol. 17, no. 6, pp. 6–16, 2003.

[5] T. Karagiannis, A. Broido, M. Faloutsos, and K. claffy, "Transport layer identification of P2P traffic," in *IMC '04: Proc. 4th ACM SIGCOMM Conference on Internet Measurement*, 2004, pp. 121–134.

[6] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An analysis of live streaming workloads on the Internet," in *IMC '04: Proc. 4th ACM SIGCOMM Conference on Internet Measurement*, 2004, pp. 41–54.

[7] Skype, http://www.skype.com/.

[8] K. T. Chen, C. Y. Huang, P. Huang, and C. L. Lei, "Quantifying Skype user satisfaction," *ACM SIGCOMM Computer Commun. Rev.*, vol. 36, no. 4, pp. 399–410, 2006.

[9] Google Talk, http://www.google.com/talk.

[10] E. P. Freire, A. Ziviani, and R. M. Salles, "Detecting Skype flows in Web traffic," in *NOMS 2008: Proceedings of the 2008 IEEE/IFIP Network Operations and Management Symposium*, 2008.

[11] P. Barford and D. Plonka, "Characteristics of network traffic flow anomalies," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, Nov. 2001.

[12] A. Ziviani, M. L. Monsores, P. S. S. Rodrigues, and A. T. A. Gomes, "Network anomaly detection using nonextensive entropy," *IEEE Commun. Lett.*, vol. 11, no. 12, pp. 1034–1036, Dec. 2007.

[13] A. Moore and K. Papagiannaki, "Toward the accurate identification of network applications," in *Proc. Passive and Active Measurement Workshop (PAM2005)*, Mar./Apr. 2005.

[14] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian, "Traffic classification on the fly," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 2, pp. 23–26, 2006.

[15] J. Ma, K. Levchenko, C. Kreibich, S. Savage, and G. M. Voelker, "Unexpected means of protocol inference," in *IMC '06: Proc. 6th ACM SIGCOMM Internet Measurement Conference*, 2006, pp. 313–326.

[16] Y. J. Won, B.-C. Park, H.-T. Ju, M.-S. Kim, and J. W. Hong, "A hybrid approach for accurate application traffic identification," in *Proc. 4th IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services*, Apr. 2006, pp. 1–8.

[17] E. P. Freire, A. Ziviani, and R. M. Salles, "On metrics to distinguish Skype flows from HTTP traffic," in *LANOMS 2007: Proc. 5th Latin American Network Operations and Management Symposium*, Sept. 2007.

[18] B. A. Mah, "An empirical model of HTTP network traffic," in *INFOCOM '97: Proc. 16th Joint Conference of the IEEE Computer and Communications Societies*, 1997.

[19] H.-K. Choi and J. O. Limb, "A behavioral model of Web traffic," in *ICNP '99: Proc. 7th International Conference on Network Protocols*. IEEE Computer Society, 1999, pp. 327–334.

[20] S. Baset and H. Schulzrinne, "An analysis of the Skype peer-to-peer Internet telephony protocol," in *INFOCOM'06: Proc. 25th IEEE International Conference on Computer Communications*, Apr. 2006.

[21] S. Guha, N. Daswani, and R. Jain, "An experimental study of the Skype peer-to-peer VoIP system," in *IPTPS'06: Proc. 5th International Workshop on Peer-to-Peer Systems*, Feb. 2006, pp. 1–6.

[22] D. Rossi, M. Mellia, and M. Meo, "Following Skype signaling footsteps," in *IT-NEWS'08: Proc. 4th International Telecommunication Networking WorkShop on QoS in Multiservice IP Networks*, Feb. 2008.

[23] K. Suh, D. R. Figueiredo, J. Kurose, and D. Towsley, "Characterizing and detecting relayed traffic: a case study using skype," in *INFOCOM'06: Proc. 25th IEEE International Conference on Computer Communications*, Apr. 2006.

[24] Y. Yu, D. Liu, J. Li, and C. Shen, "Traffic identification and overlay measurement of Skype," in *IEEE Int. Conf. on Computational Intelligence and Security*, 2006.

[25] S. Ehlert, S. Petgang, T. Magedanz, and D. Sisalem, "Analysis and signature of Skype VoIP session traffic," in *Proc. CIIT 2006: 4th IASTED International Conference on Communications, Internet, and Information Technology*, Nov./Dec. 2006, pp. 83–89.

[26] B. Sat and B. Wah, "Analysis and evaluation of the Skype and Google Talk VoIP systems," in *IEEE Int. Conf. on Multimedia and Expo*, July 2006.

[27] N. Kushman, S. Kandula, and D. Katabi, "Can you hear me now?!: it must be BGP," *ACM SIGCOMM Comp. Commun. Rev.*, vol. 37, no. 2, pp. 75–84, 2007.

[28] D. Bonfiglio, M. Mellia, M. Meo, N. Ritacca, and D. Rossi, "Tracking down Skype traffic," in *INFOCOM'08: Proc. 2008 IEEE INFOCOM*, Apr. 2008.

[29] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, and P. Tofanelli, "Revealing Skype traffic: when randomness plays with you," in *SIGCOMM'07: Proc. 2007 ACM SIGCOMM*, Aug. 2007.

[30] N. Ye and Q. Chen, "An anomaly detection technique based on a chi-square statistic for detecting intrusions into information systems," *Quality and Reliability Engineering International*, vol. 17, no. 2, pp. 105–112, 2001.

[31] J. M. Estévez-Tapiador, P. García-Teodoro, and J. E. Díaz-Verdejo, "Measuring normality in HTTP traffic for anomaly-based intrusion detection," *Computer Networks*, vol. 45, no. 2, pp. 175–193, June 2004.

[32] K. Pearson, "On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling," *Philos. Mag. Series 5*, vol. 50, pp. 157–172, 1900.

[33] F. J. Massey, Jr., "The Kolmogorov-Smirnov test of goodness of fit," *J. Amer. Statist. Assoc.*, vol. 46, pp. 68–78, 1951.

[34] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP: Session Initiation Protocol," Internet RFC 2543, 1999.

[35] X. Wang, S. Chen, and S. Jajodia, "Tracking anonymous peer-to-peer VoIP calls on the Internet," in *CCS'05: Proc. 12th ACM conference on Computer and communications security*, 2005, pp. 81–91.

[36] Extensible Messaging and Presence Protocol (XMPP), http://www.xmpp.org.

[37] XEP-0166: Jingle, http://www.xmpp.org/extensions/xep-0166.html.

[38] V. Jacobson, C. Leres, and S. McCanne, "tcpdump," http://www.tcpdump.org/.

[39] J. Elson, "tcpflow," http://www.circlemud.org/~jelson/software/tcpflow/.
[40] N. A. Macmillan and C. D. Creelman, *Detection Theory: A User's Guide*. Cambridge University Press, 1991.
[41] M. Perényi, A. Gefferth, T. D. Dang, and S. Molnár, "Skype traffic identification," in *Proc. IEEE Globecom 2007*, Washington, DC, USA, Nov. 2007.
[42] M. Perényi and S. Molnár, "Enhanced Skype traffic identification," in *Proc. 2nd International Conference on Performance Evaluation Methodologies and Tools - ValueTools'07*, Nantes, France, 2007, pp. 1–9.

Emanuel Pacheco Freire received the B.Sc. degree in Computer Engineering from the Military Institute of Engineering (IME), Rio de Janeiro - Brazil in 2002. He worked as a network administrator until 2005 and he received the MSc. degree in Computer Science from the Military Institute of Engineering (IME) in 2008.

Artur Ziviani received a B.Sc. in Electronics Engineering in 1998 and a M.Sc. in Electrical Engineering (emphasis in Computer Networking) in 1999, both from the Federal University of Rio de Janeiro (UFRJ), Brazil. In 2003, he received a Ph.D. in Computer Science from the University of Paris 6, France, where he has also been a lecturer during 2003-2004. Since 2004, he is with the National Laboratory for Scientific Computing (LNCC), Brazil. He is a member of the Editorial Board of IEEE COMMUNICATIONS SURVEYS & TUTORIALS. His research interests include mobile computing, Internet measurements, and the application of networking technologies in computer-aided medicine.

Ronaldo Moreira Salles received the B.Sc. degree in Electronic Engineering from the Military Institute of Engineering (IME), Rio de Janeiro - Brazil in 1992. He then worked in the Laboratory of Instrumentation and Control at CTex (Brazilian Army) until 1996. He received the MSc. degree in Computer Science also from the Military Institute of Engineering (IME) in 1998 and the Ph.D. degree in 2004 at the Imperial College London in the Comm. & Sig. Proc. Group. Currently, he holds a position of Postgraduate Course Coordinator of the Department of Computer Engineering at IME.