Revealing Skype Traffic: When Randomness Plays with You

Dario Bonfiglio
Politecnico di Torino
Dipartimento di Elettronica
dario.bonfiglio@polito.it

Marco Mellia
Politecnico di Torino
Dipartimento di Elettronica
marco.mellia@polito.it

Michela Meo
Politecnico di Torino
Dipartimento di Elettronica
michela.meo@polito.it

Dario Rossi
ENST Télécom Paris
Informatique et Réseaux
dario.rossi@enst.fr

Paolo Tofanelli
Motorola Inc.
Torino - Italy
fnkb76@motorola.com

ABSTRACT

Skype is a very popular VoIP software which has recently attracted the attention of the research community and network operators. Following a closed source and proprietary design, Skype protocols and algorithms are unknown. Moreover, strong encryption mechanisms are adopted by Skype, making it very difficult to even glimpse its presence from a traffic aggregate. In this paper, we propose a framework based on two complementary techniques to reveal Skype traffic in real time. The first approach, based on Pearson's Chi-Square test and agnostic to VoIP-related traffic characteristics, is used to detect Skype's fingerprint from the packet framing structure, exploiting the randomness introduced at the bit level by the encryption process. Conversely, the second approach is based on a stochastic characterization of Skype traffic in terms of packet arrival rate and packet length, which are used as features of a decision process based on Naive Bayesian Classifiers.

In order to assess the effectiveness of the above techniques, we develop an off-line cross-checking heuristic based on deep-packet inspection and flow correlation, which is interesting per se. This heuristic allows us to quantify the amount of false negatives and false positives gathered by means of the two proposed approaches: results obtained from measurements in different networks show that the technique is very effective in identifying Skype traffic.

While both Bayesian classifier and packet inspection techniques are commonly used, the idea of leveraging on randomness to reveal traffic is novel. We adopt this to identify Skype traffic, but the same methodology can be applied to other classification problems as well.

Categories and Subject Descriptors

C.4 [**Computer Communication**]: Measurement Techniques; C.2.5 [**Computer Communication Network**]: Internet

General Terms

Experimentation, Measurement

Keywords

Traffic Identification, Passive Measurement, Naïve Bayesian Classification, Pearson Chi-Square Test, Deep Packet Inspection

1. INTRODUCTION

The last few years witnessed VoIP telephony gaining a tremendous popularity, as also testified by the increasing number of operators that are offering VoIP-based phone services to residential users. Skype [1] is beyond doubt the most amazing example of this new phenomenon: developed in 2003 by the creators of KaZaa, it recently reached over 100 millions of users, becoming so popular that people indicate Skype IDs in their visiting cards.

A number of reasons for such a success can be acknowledged. First, today Internet (in terms of capacity, responsiveness, robustness) makes it possible to provide new and demanding services, including real-time interactive applications such as telephony. Second, the users attitude toward technology has deeply changed in the last few years: users are willing to accept a good service for free, even though service continuity and quality is not guaranteed; they (we?) like to have access from the same terminal and even the same application environment to a number of different communication facilities; new ways and tools to be connected to each other are easily accepted and experienced by people. Last but not least, Skype is an extremely good piece of software, carefully engineered, user friendly and efficient at the same time.

The importance of Skype traffic identification –besides being instrumental to traffic analysis and characterization for network design and provisioning– is clear when considering the interest of network operators, ranging from traffic and performance monitoring, to the design of tariff policies and traffic differentiation strategies. To date however, despite the interest recently exhibited by the research community, reliable identification of Skype traffic remains a challenging task, given that the software is proprietary and the traffic is obfuscated. The objective of this paper is to define a framework, based on two different and complementary techniques, for revealing and classifying Skype traffic from a traffic aggregate, irrespectively of the transport layer protocol that is being used (i.e., TCP or UDP). Both techniques are scalable, can be performed online, and are applicable to a more general extent than the context of Skype traffic identification. The first approach, based on Pearson's Chi Square test, is used to detect Skype's fingerprint from the packet framing structure but is agnostic to VoIP-related traffic characteristics. To the best of our knowledge, this work is the first to introduce this methodology for the purpose of traffic identification: in the following, we refer to this novel identification approach as *Chi-Square Classifier* (CSC). Conversely, the second approach is based on a stochastic characterization of Skype traffic in terms of packet arrival rate and packet length, which are employed as features of a decision process based on *Naive Bayesian Classifiers* (NBC): however, while the above features successfully allow to identify VoIP traffic, they are not representative of the application that generated it.

To proof-check the correctness of the statistical techniques, we develop a *Payload Based Classifier* (PBC), that relies on traditional technique of deep-packet inspection, combined with a per-host analysis that allows us to identify Skype clients and their generated traffic. We use the PBC to cross-check the results obtained from the statistical approaches. In particular, from the controlled testbed experiments and from real traffic traces as well, the PBC is used to create a benchmark dataset in which we classify Skype flows with a very high confidence level. Moreover, the benchmark dataset is used to tune parameters of the above classifiers, as well as to quantify the number of NBC/CSC false-positives and false-negatives. Indeed, by running the NBC and CSC onto the benchmark dataset, we can assess the effectiveness of the two classifiers, when they are either separately or jointly used. We anticipate that the combination of NBC and CSC yields to astonishingly good results. The joint NBC+CSC classification method effectively limits the number of false positives, yielding to conservative results, in the sense that the number of non-Skype flows erroneously classified as such is negligible.

A mythological analogy fits to the above framework if we personify the problem of Skype traffic identification with *Khaos*, which typically refers to unpredictability and in Greek mythology [2] was referred to as the primeval state of existence, from which the *protogenoi* (i.e., the first gods) appeared. The payload based heuristic allows us to bring light to the problem solution, and it can thus be represented by *Hemera*, the female goddess of eagles and fatheads and the personification of day. Opposite to the deep-packet inspection, the statistical techniques lye. We can identify with *Nyx*, the primordial goddess of the night, and *Erebos*, the personification of darkness and shadow, respectively the Chi-Square test and Bayesian classification. A tight relationship exists among the techniques, exactly as it exists among the protogenoi: *Hemera* is the daughter of *Nyx* and *Erebos*, which are one other's sister and brother, and are sons of *Khaos* in their turn. *Hemera*, *Nyx* and *Erebos* are used to reveal and understand *Khaos*, i.e., Skype traffic.

2. RELATED WORK

As the use of Skype spreads and increases in importance, there is the need for tools and techniques to analyze Skype traffic, which recently drawn much interest of the research community [3, 4, 5, 6, 7], as briefly overviewed in what follows.

In [3], authors provide an overview of Skype design and functions, exploring many Skype aspects under different network setups: both Skype users with machines with public IP addresses, or one of them behind port-restricted NAT, or, finally, both users behind a port-restricted NAT and UDP-restricted firewall. Authors of [4], of which unfortunately only the slides from an oral presentation are available, provide a very deep understanding of Skype internals, including many details gathered from a reverse engineering of Skype protocol and application, with a special focus on security issues. The format of low level datagram is inspected: it emerges that almost everything is cyphered, that data can be fragmented, and that an extensive use of data compression (based on arithmetic compression) is made as well. The work in [5] presents an experimental study of Skype, where results are collected by means of measurements over a five months period. Authors analyze user behavior only for *relayed* [1], rather than direct, sessions. Results pertain the population of on-line clients and their usage pattern, the number of super-nodes and bandwidth usage: thus, the identification problem is no longer related to Skype traffic but, rather, to Skype *users*.

The works closest to ours are [6, 7]. [6] deals with the evaluation of the QoS level provided by Skype calls. However, the paper focus is more on the QoS evaluation rather than on the identification of the flows. Specifically, authors consider a valid VoIP session whenever i) flow duration is longer than a threshold (namely, 10 seconds), ii) average packet rate is within a reasonable range (between 10 and 100 pkt/sec), iii) average packet size is small (between 30 and 300 bytes), and iv) Exponentially Weighted Moving-Average of the packet size falls in a given range (between 35 and 500 bytes) for the whole flow duration. However, these characteristics are typical of all VoIP traffic, and not only of Skype traffic. Therefore, authors propose a complex algorithm to identify the UDP port used by Skype. All traffic originated from, sinked by that (IP address - UDP port) will be labeled as Skype traffic. Besides being very complex, this approach applies only when Skype uses UDP at the transport layer and it needs the Skype login phase to be monitored: it is, thus, likely to fail on backbone links. Moreover, any modification during the login phase in future Skype releases will make the algorithm useless. We, on the contrary, would like to pin out all VoIP traffic generated by Skype only, possibly with simple algorithms and in any scenario. Finally, in [7] authors focus on the identification of *relayed* traffic, and present an application to Skype. The adopted approach is to correlate, at the relay node, the incoming and outgoing packet time series and bandwidth usage. More on details, authors focus on what they call "burst" of traffic. Then, for every pair of packet bursts, an analysis of the correlation of the packet arrival series within the burst is performed. While this technique has been proved to be reliable in identifying relayed Skype sessions, we point out that it is not suitable to our purpose, i.e., identifying all Skype VoIP traffic.

To summarize, the above works focus on either the analysis of Skype internals (gained by both black box approaches and reverse engineering), or the characterization of different aspects of the Skype users and traffic (in which case only fairly simple identification heuristics are used), or on methodologies to identify Skype *relayed* traffic. Therefore, we feel that the problem of identification of Skype *direct* sessions, which is the aim of this paper, has not been accorded the attention that it deserves. We reach our goals using algorithms that work in real-time, are very reliable, and can detect Skype voice flows that last few seconds only.

3. THE SKYPE SOURCE MODEL - KHAOS MODEL

In this section, we combine the knowledge gained by the work overviewed in Sec. 2 with an original study of the traffic generated by a Skype client: the aim is to both derive a model of Skype traffic sources, and gain insights that allow us to build our effective identification method. In order to derive a traffic source model, we performed several experiments in a controlled environment: our

[1]A session is *relayed* if packets from a source to a destination are routed through an intermediate node which acts as an application layer relay.

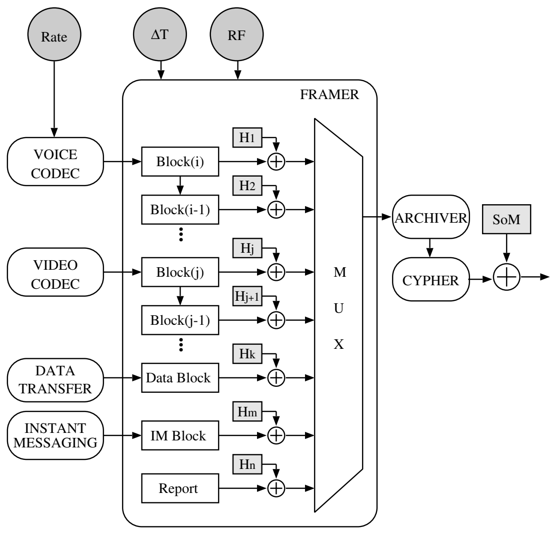

Figure 1: Schematic diagram representing the Skype message building process.

testbed involved several PCs connected by a Linux NAT / Firewall / Router / Traffic-Analyzer box. Different versions of Skype were installed, running under different operating systems such as Windows, Linux and Pocket-PC. Different network scenarios were tested, either emulated by using NIST Net [8] to enforce several combinations of delay and packet losses, or using different access networks (e.g., Ethernet, xDSL, UMTS access data-link).

Skype offers VoIP capabilities in two different "modes". In the first one, traffic is generated between two end-hosts, each of which is running a Skype client: we call *End-to-End* (E2E) the generated traffic. The second communication mode happens between an end-host and a traditional PSTN phone, involved through the Skype-out/Skypein services: in this case, we call *End-to-Out* (E2O) the generated traffic. Moreover, we anticipate that we seek to derive a quite general and tunable model that, instead of being precisely but rigidly fitted to our testbed measurements, can adapt to future software releases.

3.1 The Skype Source Model

A schematic diagram of the Skype source model is reported in Fig. 1. Since Skype supports voice, video, chat and data transfer, several information sources are highlighted. Each source has different characteristics, but all generate information *blocks* that are then multiplexed in a *frame*.

Considering the voice source, the voice encoder used during a call, i.e., the *Voice Codec*, outputs blocks of encoded voice. Since our focus is on voice calls, in the following we denote the voice encoder simply as the *Codec*. The *Framer* is then responsible for creating *Skype frames*, by multiplexing into a single frame one or more blocks (e.g., to cope with the potential loss of the immediately preceding frames, or to modify the message generation rate). Possible video/data/chat/report blocks can be multiplexed as well in a single frame and some additional headers, named $H_1, H_2, \ldots$ in the figure, can be added. Once a frame has been created, it is then arithmetically compressed by the *Archiver* and encrypted by the *Cypher*. Finally, an additional non-cyphered header may be present, denoted by *Start of Message* (SoM) in the figure. The SoM will play a crucial role in the PBC, and we discuss it deeply in Sec. 3.2.1. The output of the process depicted in Fig. 1 is a *Skype message*, that will be then encapsulated in either a UDP or TCP segment.

Codec	Frame Size [ms]	Bitrate [Kbps]
ISAC	30,60	$10 \div 32$
ILBC	20,30	13.3, 15.2
G.729	10	8
iPCM-wb	10,20,30,40	80 (mean)
EG.711A/U	10,20,30,40	48,56,64
PCM A/U	10,20,30,40	64

Table 1: Nominal Characteristics of Skype Codecs.

At the input side, three parameters have a crucial role in determining the characteristics of the generated traffic: i) *Rate* is the bitrate used by the source, i.e., the Codec rate. ii) ΔT, that represents the Skype message framing time, is the time elapsed between two subsequent Skype messages belonging to the same flow. iii) *RF* is the Redundancy Factor, i.e., the number of past blocks that Skype retransmits, independently from the adopted Codec, along with the current block.

All parameters may change during the connection lifetime, depending on the network working conditions. As an example, Fig. 2 reports a message trace observed during a voice call between two clients in which we artificially enforced the available bandwidth. Top plot reports the average bitrate evaluated every second and the imposed bandwidth limit; middle plot reports ΔT, and bottom plot reports the packet size versus time. Looking at the message size, the effects of both variable-bitrate Codec and the Framer are clearly visible. Indeed, at the connection beginning (time [0:30]s), messages are approximately double the size of the messages in the second part of the call (time [40:150]s). In the last portion of the connection, a change in the message size pattern is probably due to a change of the Codec bitrate. By looking at ΔT, it is possible to observe that the average message framing takes values of 20, 30 or 60 ms.

In our experiments, we observed many different operating points, where an operating point is defined as $(Rate, \Delta T, RF)$ tuple. In particular, we observed RF between 1 and 4, and ΔT varying between 0.5 and 6 times the actual Codec framing time. Despite we did not observe all the possible combinations of the three parameters, we cannot exclude that other network conditions could lead to Skype selecting other operating points: therefore, we adopt a conservative approach and prefer a slightly more general model that takes into account such possibilities as well.

For completeness, Tab. 1 reports the details about the different Codecs supported by Skype[2]: the Codec name, nominal frame size and bitrate are reported. All Codecs are standard except the ISAC one, which is a proprietary solution of GlobalIPSound [9]. ISAC is the preferred Codec for E2E (End-to-end) calls, while the G.729 Codec is preferred for E2O (Skypeout) calls.

3.2 More on Skype Messages

3.2.1 The Start of Message (SoM)

According to [10], Skype uses state-of-the-art AES and RSA algorithms to cypher messages. However, the choice of the transport layer protocol has an important implication. TCP implements

[2]Codec are automatically selected. However, it is possible to force Codec selection enclosing the list of disabled Codecs between the tag `<DisableCodecs> <\DisableCodecs>` in the `config.xml` file.

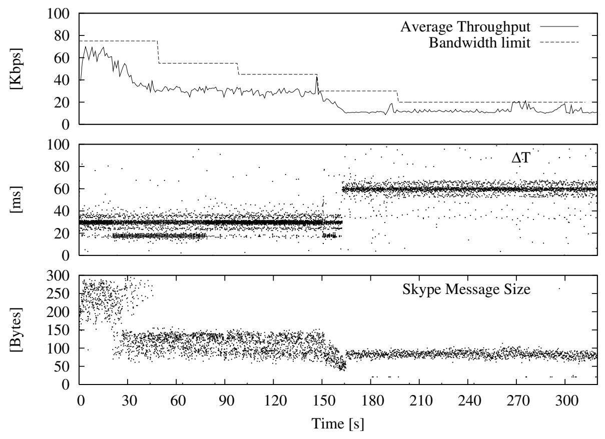

Figure 2: Sample trace: Skype message size, framing and average bitrate during a voice call.

a connection-oriented reliable transport and the application is guaranteed to receive all data segments in the correct sequence. When using TCP, Skype therefore cyphers the whole content of all messages. Conversely, the connectionless unreliable service offered by UDP no longer guarantees in-sequence and all data delivery. Therefore, when using UDP, Skype receiver must extract from the application layer header some additional information to detect and deal with possible incorrect stream lining. Such information cannot be protected by means of a stream cypher, but can only be *obfuscated* by means of some function based on single packet payload [4]. Thus, with UDP, Skype cannot encrypt the *whole* message. Moreover, Skype sends and receives UDP segments using a fixed port to avoid additional signaling needed to negotiate a dynamic port. Therefore some application layer headers must be added to multiplex different messages into the same transport flow (e.g., voice, instant messaging, file transfer and signaling messages between the same two end-points). Based on the two previous deductions, we conclude that, when Skype messages are encapsulated in UDP segments, a portion of the Skype messages can be identified by inspecting the UDP payload: this is what we call the Start of Message (SoM).

3.2.2 End-to-end Messages

By looking at the UDP encapsulated messages generated between two Skype clients, i.e., E2E messages, the following fields are identified:

- `ID`: a 16-bit long identifier (byte 1 and byte 2) used to uniquely identify the message; it is randomly selected by the sender query, and copied in the receiver reply.
- `Fun`: a 5-bit long field obfuscated into a byte (byte 3) stating the payload type. Three random bits can be removed by considering a `0x8f` bitmask.
- Frame: containing a possibly multiplexed sequence of cyphered information and voice blocks.

The `ID` and `Fun` fields are part of the SoM header. By inspecting the values assumed by the `Fun` field, we have that `0x02, 0x03, 0x07` and `0x0f` are used to indicate signaling messages generated during Skype login phase, or connection management. `0x0d` indicates a DATA message, that can contain: i) encoded voice blocks, ii) video encoded blocks, iii) chat messages or even iv) chunks of files transferred by users during a Skype session. We never observed other values being assumed by the `Fun` field.

3.2.3 Skypeout Messages

Skypeout calls have an initial signaling phase between the client and some super-nodes. Then, a voice encoded stream of messages is activated among the caller and a gateway node, which converts the call to the PSTN. The UDP port value of the PSTN gateway is set to 12340, while its IP address may change according to the selected gateway. Looking at the payload of the UDP encapsulated voice messages, we notice that after a variable number of initial messages, the first four bytes take always the same value, therefore hinting to a different SoM format than E2E messages: we assume that these four bytes are used by the PSTN gateway as a unique Connection IDentifier (`CID`). Note that it is possible for the `CID` to change during the connection lifetime, and our testbed showed that it is very likely for such changes to happen during connection start. Bytes from 5 to the end of the message show no deterministic meaning.

3.3 Khaos: How to proceed with Skype traffic identification

In the following, we exploit the above information to classify Skype traffic. In particular, we propose three classifiers that exploit different aspects and characteristics of Skype traffic.

Nyx, the first classifier referred to as *Chi-Square Classifier* (CSC) in the following, focuses on the cyphering mechanisms: the CSC decision is based on the analysis of the message content randomness introduced by the cypher. When TCP is employed at the transport layer, the whole content of a Skype message is encrypted, so that message bytes appear to be randomly distributed. Conversely, in the UDP case, only a certain portion of the message carries randomly distributed values, whereas other portions of the messages (e.g., the `Fun` and `ID` fields for E2E and the `CID` for E2O calls) exhibit statistical properties typical of deterministic data. The CSC allows us to distinguish the traffic generated by *Skype clients* from the one of other VoIP sources since they use different header format, e.g., RTP. In the case of UDP, it provides some additional information on the Skype mode (E2E or E2O).

Erebos, the second classifier, referred to as *Naive Bayes Classifier* (NBC) in the following, is based instead on the peculiar properties of the traffic generated by the voice Codec and the framer. Specifically, we characterize the stochastic properties of the traffic generated by the source in terms of packet length and inter-packet gap. Then, we define a classifier whose decision process is based on the quantitative evaluation, by means of a Bayesian technique, of the resemblance of potential Skype flows to the expected stochastic characteristics.

Hemera, the third classifier referred to as *Payload-Based Classifier* (PBC) in the following, is based on a more traditional approach of deep packet inspection, and, in particular, it leverages on the SoM header content. This classifier applies only to flows transported by UDP and couples the per-flow inspection of packet content with more sophisticated per-host state information (such as the port used for the data exchange, or traces of previous Skype activities by the same host), so as to reduce the number of misidentification.

4. THE CLASSIFIERS

4.1 Nyx: Chi-Square Classifier

The first classifier we present uses the Pearson's Chi-Square statistical test to assess whether and which message portions are encrypted. The aim of this test is to check if the message under analysis complies with one of the Skype message formats early described, and can thus be considered an analogous of forensic methods to reveal fingerprints. More on details, depending on the kind of flow, we can expect different characteristics of the message content after cyphering. In particular, we distinguish the following cases:

- *E2E Skype flow transported by UDP.*
  The FUN bits are deterministic, while all the other bits in the message are cyphered. By assuming that the cypher is optimally working, we can expect cyphered bits to appear completely *random*, i.e., uniformly distributed.

- *E2O Skype flow transported by UDP.*
  The first four bytes of the message are *deterministic*, since they represent the CID. Besides these four bytes, the remaining part of the message is cyphered and the corresponding bits appear to be random.

- *Skype flow transported by TCP.*
  The whole message is cyphered, irrespectively of the E2E/E2O Skype mode: therefore, the whole message appears to be random.

In order to classify if a flow is generated by Skype, we use the above observation to test whether the message content is compliant with our expectation by means of Pearson's Chi-Square test. The test is designed to verify whether the behavior of an object, observed for a finite number of times, follows an expected behavior: this is done by computing the deviation of the observed output values with respect to the expected distribution of the outputs. Assume that we build an experiment over the object by observing its output for n_{TOT} times (with n_{TOT} large) and assume that there are n possible outputs for each observation. If the expected distribution of the output is such that output value i, with $i = 0, \dots, n-1$, occurs with probability p_i, the expected number of occurrences of i is $E_i = n_{TOT} p_i$. Now, let O_i be the number of occurrences of i actually observed during the whole experiment. The value

$$\chi^2 = \sum_{i=0}^{n-1} \frac{(O_i - E_i)^2}{E_i} \tag{1}$$

is a measurement of the deviation of the observed behavior with respect to the expected behavior. If the observed object really behaves as expected, then the χ^2 value computed in (1) is distributed according to a Chi-Square distribution with $n-1$ degrees of freedom, the deviation being simply due to the finite nature of the experiment.

The test is often employed for a single experiment in the following way. Make the *hypothesis* that the output of the object is distributed according to probabilities p_i's and perform the experiment as described above. The hypothesis is rejected with *significance level* α if the value χ^2 computed as in (1) is larger than the $(1-\alpha)-$th quantile of the Chi-Square distribution with $n-1$ degrees of freedom. We build our CSC on this idea, by checking whether the content of the messages belonging to a given flow complies with one of the different random/deterministic Skype traffic behavior described above.

Our experiment works as follows. For each message belonging to a flow, we consider the first G groups of b bits (i.e., we consider the first Gb bits of the message) and we compute, for each block $g = 1, \dots G$, the variables O_i^g which count the number of times that the g-th block assumed value i, with $i = 0, \dots, 2^b - 1$. At the

Skype Mode	SoM			Payload
Byte pos.	1–2	3	4	5-...
E2E over UDP	Rnd	Mixed	Rnd	Rnd
E2O over UDP	Det	Det	Det	Rnd
E2E-E2O over TCP	Rnd	Rnd	Rnd	Rnd

Table 2: Message content characteristics.

flow end, we evaluate the Chi-Square test for each group of bits,

$$\chi_g^2 = \sum_{i=0}^{2^b-1} \frac{(O_i^g - E_i)^2}{E_i} \quad \text{with} \quad g = 1, \dots G. \tag{2}$$

We have then to test whether the values of χ_g^2 are such that our hypothesis is verified. For the hypothesis, we use the message content characteristics early discussed and summarized in Table 2: we expect, depending on the kind of flow and the transport layer protocol, parts of the Skype messages to be random and parts to be deterministic (or almost deterministic). For example, E2E flows over UDP have the first, second and fourth bytes cyphered, or random ('Rnd' in the table), while byte 3 contains a few random bits and some constant bits (this case is labeled 'Mixed' in the table). The whole SoM of the E2O flows over UDP are deterministic ('Det' in the table), while messages are completely random for flows transported by TCP.

In order to check if group g is random, deterministic or mixed, we consider, as expected behavior, the Chi-Square distribution obtained for uniformly distributed bits. In this case, $E_i = n_{TOT}/2^b$ for all i, where n_{TOT} is the number of messages belonging to the flow. We then compare the measured values of χ_g^2 with some thresholds derived from the distribution of the Chi-Square with $n - 1 = 2^b - 1$ degrees of freedom; we denote the thresholds by $\chi^2(\text{Rnd})$, $\chi^2(\text{Mixed})$, and $\chi^2(\text{Det})$.

Different criteria can be proposed based on the value of G and b, and the thresholds. Our choice is $b = 4$ bits, and $G = 16$, meaning that the first 8 bytes (half of which account for the SoM header and half for the payload) are considered. The reference Chi-Square distribution in this case has $2^b - 1 = 15$ degrees of freedom and $E_i = n_{TOT}/16$ for all $i = 0, \dots, 15$. We then classify the flow according to the following criteria.

- *E2E over UDP.*

  $$\max_{g \in \mathcal{G}'}(\chi_g^2) < \chi^2(\text{Rnd}) \quad \wedge \quad \min_{g \in \{5,6\}}(\chi_g^2) > \chi^2(\text{Mixed})$$

  where $\mathcal{G}' = \{g \mid 1 \le g \le G, g \ne 5, 6\}$ is the set of groups corresponding to the random part of the E2E message. The rationale of this criterion is that random blocks should be "similar" to the uniform distribution leading to relatively small values of χ_g^2. At the same time, mixed blocks that contain a few deterministic bits, tend to have larger values of χ_g^2, because they significantly deviate from the typical random behavior.

- *E2O over UDP.*

  $$\min_{g=1,\dots,8}(\chi_g^2) > \chi^2(\text{Det}) \quad \wedge \quad \max_{g=9,\dots,16}(\chi_g^2) < \chi^2(\text{Rnd})$$

  In this case, the SoM (i.e., the first four bytes, or eight groups of $b = 4$ bits) should be deterministic, whereas the remaining portion random.

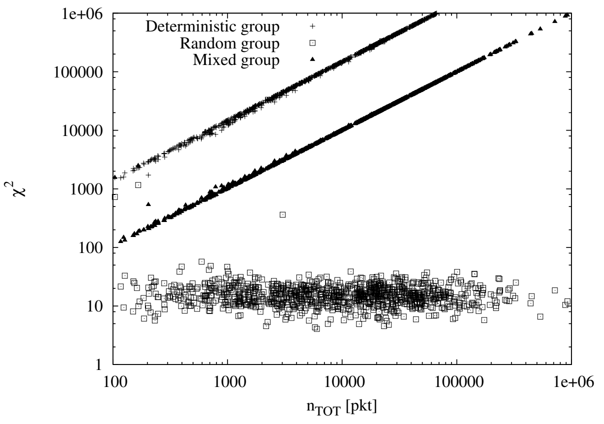

Figure 3: χ_g^2 versus the number of observations for deterministic, random and mixed (1 deterministic and 3 random bits) groups.

- *E2E or E2O over TCP.*

$$\max_{g=1,\ldots,16}(\chi_g^2) < \chi^2(\text{Rnd})$$

All considered groups should be random.

- *Not Skype.*

Otherwise.

Notice that the Chi-Square test works if the number of observations n_{TOT} is large, say that n_{TOT} is such that $E_i \geq 5$ for all possible i. In our case, this means that we can apply the test only to flows whose number of messages is,

$$\frac{n_{TOT}}{2^b} \geq 5 \tag{3}$$

that is $n_{TOT} \geq 80$ with our choice for b. In the numerical results Section, we choose $n_{TOT} \geq 100$, which roughly corresponds to 3 s of a voice flow. The difference in the observed values of χ_g^2 between a deterministic or a random block significantly increases with the flow length. Indeed, for a deterministic block, since only one value for the block is possible, the value of χ_g^2 is,

$$\chi_g^2 = \sum_{i=0}^{2^b-1} \frac{(O_i^g - E_k)^2}{E_i} \tag{4}$$

$$= \frac{(n_{TOT} - E)^2 + (2^b - 1)\,E^2}{E} = n_{TOT}\left(2^b - 1\right). \tag{5}$$

χ_g^2 linearly increases with n_{TOT}, meaning that the longer the flow is, the more reliable the identification of a deterministic block is. Reliability of the test increases also with b; however, due to the implication expressed by (3), the value of b should be kept small.

Consider now the case of a mixed block. If one bit is fixed and the others are random (as is the case of the fifth group of an E2E flow transported by UDP), $O_i = 0$ for half the possible values of i, and larger than 0 for the remaining i. Since the possible values of i are 2^b in total, the χ_g^2 of a mixed block is,

$$\begin{aligned}\chi_g^2 &= \sum_{i=1}^{2^{b-1}} \frac{(O_i^g - E)^2}{E} + 2^{b-1}\frac{E^2}{E} \\ &= 2\chi^2_{2^{b-1}-1} + n_{TOT} \end{aligned}\tag{6}$$

where $\chi^2_{2^{b-1}-1}$ is the Chi-Square with $2^{b-1} - 1$ degrees of freedom; in other terms, it is a value that can be achieved from an experiment with random bits over 2^{b-1} possible values. This result suggests that in the case of a group with 1 deterministic bit χ_g^2 is again linearly increasing with n_{TOT}. Similarly, it is possible to estimate what has to be expected from the case of more than one deterministic bit, as is the case of group 6 in a E2E flow over UDP.

Figure 3 reports the value of χ_g^2 for the considered kinds of groups, computed over the flows that were identified as Skype flows in the experiments described in Sec. 5. In particular, we selected block 1 and 5 from E2E flows, and block 1 from E2O flows, which are examples of random, mixed and deterministic blocks. At flow end, the values of the corresponding χ_g^2 is then plotted with x-axis value equal to the flow length. Notice the growth of χ_g^2 with n_{TOT} for both deterministic and mixed groups, with completely deterministic groups having a much larger deviation from random behavior. The figure suggests that, as the behavior of the three groups are clearly distinguishable from the value of the χ_g^2, the setting of the thresholds is not critical. Thus, in order to reduce the number of parameters, we simply set $\chi^2(\text{Det}) = \chi^2(\text{Rnd}) = \chi^2(\text{Mixed}) = 150$.

4.2 Erebos: Naive Bayes Classifier

As mentioned before, while the CSC is based on the cypher, the NBC proposed in this section makes use of the peculiar characteristics of real-time voice traffic. The algorithm is based on a traffic characterization that, combining the voice Codec with the framer, provides a stochastic description of Skype traffic: resemblance of the measured traffic to the Skype source is then evaluated by a coordinated set of *Naive Bayes Classifiers* [11]. Naive Bayes Classifiers are successfully adopted in the data-mining field, and, despite their simplistic assumptions, are known to provide excellent results [11]. Recently they have also been adopted to classify Internet traffic, showing very good performance, see [12] and references therein.

4.2.1 The Classifier

The Naive Bayes Classifier (NBC) technique derives from the Bayes theorem and is based on the following idea. Suppose that we observe an object that can be described by a number of observable quantities called *features*. Many different features can be gathered. Let vector $\mathbf{x} = [x_i]$ represent the different samples, where x_i is the $i-$th observed value of feature X. Then, we are interested in quantifying the probability $P\{C|\mathbf{x}\}$ that the object belongs to class C, given that we observed the sequence $\mathbf{x}$ for feature X. The NBC technique relies on the knowledge of the *a-priori* probability $P\{\mathbf{x}|C\}$ to infer the *a-posteriori* probability $P\{C|\mathbf{x}\}$,

$$P\{C|\mathbf{x}\} = \frac{P\{C,\mathbf{x}\}}{P\{\mathbf{x}\}} = \frac{P\{C,\mathbf{x}\}}{P\{\mathbf{x}\}}\frac{P\{C\}}{P\{C\}} = \frac{P\{\mathbf{x}|C\}}{P\{\mathbf{x}\}}P\{C\}. \tag{7}$$

NBC is based on the assumption that the x_i's are independent (the attribute "naïve" refers to this assumption), so that the following holds:

$$P\{\mathbf{x}|C\} = \prod_i P\{x_i|C\}. \tag{8}$$

Often, instead of evaluating $P\{C|\mathbf{x}\}$, one is interested in identifying the likelihood of being of class C rather than other classes, so that a maximum-likelihood criteria can be used. $P\{\mathbf{x}|C\}$ is often referred to as *belief* in the literature: the larger the belief of a given class is, the larger the probability of belonging to that class is.

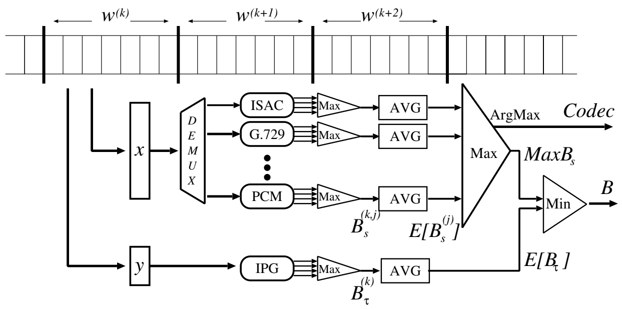

Figure 4: Schematic diagram of the NBC.

4.2.2 *Feature Selection*

In order to properly select a small number of features that can help in classifying Skype flows, we consider the main aspects in which Skype traffic differentiates from typical Internet traffic. Given the nature of real-time voice communication, a Skype client generates a low bit-rate flow which lasts for several tens of seconds and is composed of several small messages, whose size depends on the Codec rate and framing time ΔT. Data traffic, on the contrary, tends to be much burstier, with large messages and possibly high bit rates. Thus, we select as features:

- The *message size*, that is the length of the message encapsulated into the transport layer protocol segment. In particular, we consider a window of w messages and for each of them we check the message size. Thus, we have:

  $$\mathbf{x} = [s_1, s_2, \ldots, s_w]$$

  where s_i is the message size of the i-th packet of a window of w consecutive packets. Being the message size strongly dependent on the specific Codec, we develop a NBC for each of the Codecs supported by Skype, see Tab. 1.

- The *average-Inter Packet Gap* (average-IPG), evaluated as $1/w$ times the time elapsed between the reception of the first and the w-th packet in a window. In this case, we have a single classifier that does not depend on the employed Codec:

  $$\mathbf{y} = [\tau] = [(t_w - t_1)/w]$$

We define the belief of the message size and average-IPG features, respectively denote by B_s and B_τ, in terms of sums of logarithms instead of products as in (8), so as to limit numerical cancellation, i.e., to avoid that observations with very low probability make the belief go to 0.

$$B_s(C) = \frac{1}{w}\sum_{i=1}^{w} \log P\{s_i|C\} \qquad B_\tau(C) = \log P\{\tau|C\}. \quad (9)$$

It is worth noticing that we chose the features as some aspects of traffic characteristic over a short window of samples, so that the effect of the high variability of traffic is smoothed. In particular, we decided to consider the average-IPG value over a window of packets rather than the instantaneous IPG values, in order to get rid of the effects of access and queuing delays faced by packets. Moreover, average-IPG, in the way we define it, is rather insensitive to the specific point of the network in which the monitoring tool is settled. For the choice of w we verified that, the impact of its value is negligible as far as it is larger than 10. We therefore set w to 30, that roughly corresponds to a time window of 1 s.

4.2.3 *Feature Characterization*

Let us first start by considering the message size feature. Based on the Skype source model described in Sec. 3, we characterize the message size distribution for each possible Codec using both nominal parameters (e.g., Codec rates) and observations obtained from test-bed experiments (e.g., to derive RF). In particular, a Codec can be in different working points and we represent the message size for a given working point by means of a Gaussian distribution properly fitted to the measurements obtained in our test-bed. A working point corresponds to different settings of the following parameters: i) $Rate$, ii) header length H, iii) redundancy factor RF, and iv) message framing time ΔT. For each combination of the values $\{Rate, H, RF, \Delta T\}$, the message size distribution is represented by a Gaussian distribution, $\mathcal{N}(\mu, \sigma)$, with mean value set to:

$$\mu = (Rate\Delta T + len(H))RF + len(SoM). \qquad (10)$$

The following values have been considered: $Rate$ as in Tab. 1, $RF \in \{1, 2, 3, 4\}$, $len(SoM) = 4$ Bytes, $len(H) = 8$ Bytes, $\Delta T \in \{10, 20, 30, 40, 50, 60\}$ ms. For the setting of the standard deviation σ our fitting suggests to use $\sigma = 1$ for constant bitrate Codecs, e.g., G729, and $\sigma = 7.5$ for the variable bitrate Codecs, e.g., ISAC. A better fitting procedure could be used, but since Skype internal details are unknown, it would be difficult to actually include all possible cases.

For what concerns the average-IPG, it is modeled by a Gaussian distribution with $\mu = \Delta T$ and $\sigma = 1$, with no Codec specific distinction.

4.2.4 *Derivation of the Beliefs*

Fig. 4 sketches the proposed classifier design, that combines two sets of NBCs: some NBCs are jointly employed for the message size, and a single NBC is used for the average-IPG.

Consider the message size. In the k-th measurement window, a new value of the belief is evaluated for any Codec j (i.e., any message size NBC), by choosing the maximum $B_s^{(k,j)}(C)$ among all the NBC beliefs (i.e., different working states of the j-th Codec): let this maximum be denoted by $B_s^{(k,j)}$. Then, a sequence of beliefs over time k is generated for each Codec: the *flow* belief for Codec j is obtained by taking the time average of the maximum beliefs of j,

$$E[B_s^{(j)}] = E_k[B_s^{(k,j)}]. \qquad (11)$$

Now, in order to set a belief for the message size feature, the maximum between the beliefs is taken,

$$MaxB_s = \max_j(E[B_s^{(j)}]) \qquad (12)$$

so that the most likely Codec is

$$Codec = argmax(\max_j(E[B_s^{(j)}])). \qquad (13)$$

The rational behind this procedure is that, since Codecs may change their working state, we first derive a time evolution of the belief for each Codec; then, by taking the time average, we gather the flow belief for each Codec and we can choose which is the most likely employed Codec.

The NBC relative to the average-IPG operates in a similar way, but does not need to take into account for the different Codecs. For each window k, the belief $B_\tau^{(k)}$ is derived. The *flow* belief is then obtained by the time average,

$$E[B_\tau] = E_k[B_\tau^{(k)}]. \qquad (14)$$

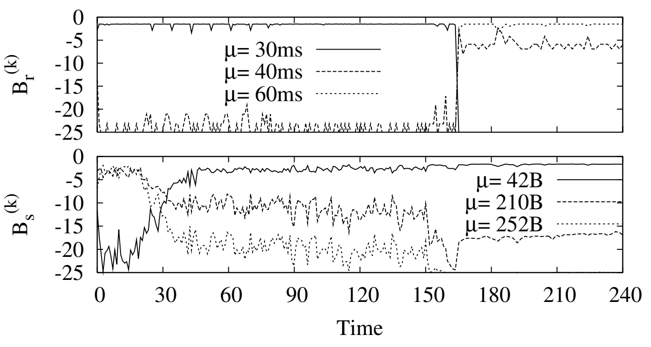

Figure 5: Example of the belief evolution during the flow lifetime for the ISAC message size NBC on the bottom plot and average-IPG on the top plot. ISAC Codec is considered.

Finally, in order to decide whether the flow was generated by Skype or not, the message size and average-IPG classifiers are combined by evaluating the minimum of the previously derived beliefs,

$$B = min\,(MaxB_s, E[B_\tau]) \tag{15}$$

and by comparing it to a threshold B_{min}. If

$$B > B_{min} \tag{16}$$

the flow is classified as being a Skype voice flow. Note that this is equivalent to require that the flow is a valid Skype flow according to both the message-size and average-IPG classifiers, therefore defining a conservative approach.

As an example, Fig. 5 shows the beliefs obtained running the NBC on the same sample-flow whose evolution is reported in Fig. 2, in which the ISAC Codec was used. Figure reports the outputs of the IPG and ISAC NBCs corresponding to the different working states of the Codec. Each working point is identified by the corresponding mean (μ) value of the Gaussian distribution. Considering the average-IPG feature (top plot), it can be observed that the $\mu = 30$ ms class exhibits the highest belief until about 170s. Then, the NBC correctly detects the change on ΔT to the $\mu = 60$ ms class. The output of the message-size based NBC is more fuzzy, since the message-size is more variable. In particular, it is possible to observe the transient phase at about 25 s and 170 s, corresponding to a change in the RF and $Rate$ of the Skype source. In both cases, the maximum windowed belief is most of the time larger than -5, and the flow belief is also very large, hinting to a valid Skype flow.

4.3 Hemera: Payload Based Classifier

Payload-based classification (PBC) is a traditional technique used to detect different traffic classes from a traffic aggregate. By exploiting the knowledge of protocol header format, payload-based classification consists in inspecting the packet payload at different layers and matching the headers to those of known applications.

Following this approach and exploiting the observations presented in previous sections –in particular, exploiting the presence of the SoM in Skype messages transported by UDP– we devise a PBC algorithm to reveal Skype traffic. As previously stated, this task is made difficult by both obfuscation and cryptographic techniques [4, 10]: therefore, additional per-host information is required to assist the per-flow identification process.

While the coupled per-host per-flow approach makes PBC identification very robust and reliable, it must not be neglected that PBC has several important drawbacks. First, since it requires packet payload inspection and per-host state, it is expensive. Second, and even more important, PBC is unfeasible when TCP is selected as transport layer protocol, or when tunneling techniques, e.g., VPNs, are used. Third, PBC possibly needs to be constantly updated according to changes in the Skype SoM format. Finally, the PBC requires to observe all flows generated by a given source host. It is therefore impractical in backbone links, due to asymmetric routing and to different paths followed by packets directed to different destinations. For these reasons, we stress that our aim is to use PBC only for cross-checking purposes.

4.3.1 Per-flow Identification

A UDP flow is identified by using the traditional tuple (source IP, destination IP, source port, destination port, protocol type). A flow starts when a packet is first observed, while an inactivity timeout (conservatively set to 20 s) is adopted to define the flow end.

For each UDP flow, upon arrival of a new packet, the packet is classified according to the following criteria. A Skype *signaling* message is detected if the `Fun` field takes a value in {`0x07`, `0x02`, `0x03`, `0x0f`}, while a `Fun` field equal to {`0x0d`} is used to identify Skype E2E messages. E2O voice messages are instead detected when the four-bytes `CID` Skypeout signature is matched: i.e., a valid E2O message is therefore required to have the same `CID` as the previous one, and to differ in byte 5. UDP port 12340 is also required to be used as either source or destination port.

Following this packet classification, four counters are updated for each flow: i) the total number of packets in the flow, n_{TOT}, ii) the number of Skype signaling messages, n_{SIG}, iii) the number of Skype E2E data/video/chat/voice messages, n_{E2E}, and iv) the number of E2O voice messages, n_{E2O}. We notice that $n_{SIG} + n_{E2E} + n_{E2O} \leq n_{TOT}$, as some packets may be classified as non-Skype.

When the inactivity timer expires, the flow is considered to be complete and a threshold based algorithm is adopted to classify the flow. Let T_{voice}, P_{voice}, T_{sig} and P_{sig} represent the minimum number, minimum fraction of voice messages and signaling packets, respectively. The algorithm identifies a flow as *potential* E2O, E2E, signaling or non-Skype flow according to the following strategy:

- *E2O* if $(n_{E2O} > T_{voice}) \wedge (n_{E2O}/n_{TOT} > P_{voice})$ $\wedge(\mathit{src\_port} = 12340 \vee \mathit{dst\_port} = 12340)$
- *E2E* if $(n_{E2E} > T_{voice}) \wedge (n_{E2E}/n_{TOT} > P_{voice})$
- *SIGNALING* if $(n_{SIG} > T_{sig}) \wedge (n_{SIG}/n_{TOT} > P_{sig})$
- *NOT_SKYPE* otherwise.

The thresholds are set to: T_{voice} =100 packets, T_{sig} =10 packets, and $P_{voice} = P_{sig}$ =0.9.

4.3.2 Per-host Identification

Since the per-flow classification algorithm may lead to an excessive number of misclassified flows, we also develop a companion algorithm based on per-host identification: only potential Skype flows identified by the per-flow classifier that also pass the per-host identification are considered as generated by a Skype source.

The per-host algorithm is based on the fact that a Skype client always uses the same UDP port to send/receive traffic. During its period of activity, a Skype client contacts several other nodes, so that each client generates many flows to different clients. Moreover, to setup a voice connection, signaling messages are exchanged between the endpoints, leading to several signaling E2E flows. By exploiting this behavior, it is possible to identify a Skype client running in a given host by considering the (source IP, source UDP port) couple. Let N_{SIG}, N_{E2E} and N_{E2O} denote the number of signaling, E2E, E2O *flows* generated by the same *source*, i.e., the same (source IP address and UDP port) couple, according to the per-flow identification criteria described in the previous section, and let N_{TOT} be the total number of flows generated by the same source. A flow is identified as being generated by Skype if the following conditions hold:

- At least 80% of the flows generated by the source are Skype signalling flows, $\frac{N_{SIG}}{N_{TOT}} > 0.8$
- At least 95% of the flows generated are Skype flows, signalling or voice, $\frac{N_{SIG}+N_{E2E}+N_{E2O}}{N_{TOT}} > 0.95$
- The source generated at least 5 flows, $N_{TOT} > 5$

In order to double-check the classification, all outputs of the PBC have been manually inspected, so that possible false negatives/positives are added/removed. We checked if the host (address,port) was a Skype port by performing a fake HTTP "GET" request and looking for the Skype reply "HTTP/1.0 501 Not Implemented". We repeated this check for 1 week. In case no Skype error message was received or reply was observed, we discarded the flow to be conservative. Therefore, we consider the PBC output to be error free. Note that the PBC is very effective in identifying Skype E2E and E2O flows. However, it is impossible to distinguish voice/video/data/chat E2E flows, since the SoM header is identical in all cases.

An important remark has to be made concerning the complexity of the per-host algorithm – which is much larger than the simple per-flow classification algorithm, as it requires to monitor and correlate all flows generated by every client. While this may be feasible in a stub network in which all packets flow through the same node independently of their destination, it is clearly unfeasible in backbone links. Moreover, the per-host algorithm requires to track the number of flows generated per source (IP address, UDP port) identifier, which can grow very large even in a small campus network.

Notice also that, given a host IP address, more than one UDP port can be identified as Skype port, e.g., in case NAT is used, or in case addresses are leased using DHCP mechanisms.

5. EXPERIMENTS

In this section we present experimental results obtained running the classification algorithms on real data traffic – while we defer the analysis of the sensitivity of classification on the thresholds of the decision process to Sec. 5.2. In particular, we select two datasets:

- CAMPUS: refers to a 95 hours long trace collected at our campus access link starting on Monday the 29th of May 2006.
- ISP: refers to a one day long trace, collected from the POP of FastWeb [13], a major Italian ISP, on Monday the 15th of May 2006.

The CAMPUS dataset is representative of a typical data connection to the Internet [14], in which most of the traffic is due to TCP data flows carrying web, email, bulk file transfer services. Users can be administrative, faculty members or students. No P2P traffic is allowed by means of strict policies enforced by firewalls. The ISP dataset is, on the contrary, very peculiar, since it refers to a very innovative ISP which is providing end users (residential, SOHO or large companies) with data, voice and video over IP by means of either an ADSL or a FTTH link (no PSTN link is offered). Traffic is therefore composed of data transfers over TCP, VoIP and VideoIP traffic over RTP/UDP. Moreover, users make extensive use of P2P applications, VPN services, etc. ISP dataset is therefore more heterogeneous than the CAMPUS one, and we expect it to represent a stiffer scenario. At the same time, in the ISP dataset we also expect little Skype traffic, as the ISP offers flat-rate tariff to phone calls.

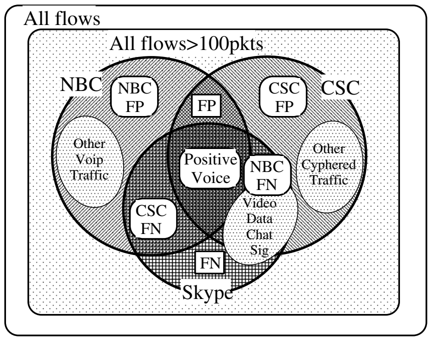

Figure 6: Representation of False Positives, False Negatives with the separate or joint NBC and CSC usage.

5.1 Measurement Results

Assuming that the PBC classification is reliable for UDP traffic, Fig. 6 sketches all the possible combinations of flow classification:

- *All flows* included in the dataset.
- *All eligible flows* that have $N_{TOT} > 100$.
- *Positive Voice* flows pass both the CSC and NBC test, and are verified by the PBC to be Skype flows.
- *False Negatives (FN)* are classified as Skype by the PBC, but are discarded by both the NBC and the CSC.
- *CSC False Negatives (CSC-FN)* are discarded by the CSC only.
- *NBC False Negatives (NBC-FN)* are discarded by the NBC only.
- *CSC False Positives (CSC-FP)* pass the CSC test only.
- *NBC False Positives (NBC-FP)* pass the NBC test only.
- *False Positives (FP)* pass both the NBC and CSC tests but fail the PBC test.

In the following, the percentage of FN is computed with respect to the number of flows identified by the PBC as Skype E2E/E2O; the percentage of FP is evaluated with respect to the number of flows that are eligible but not Skype (i.e., flows with at least 100 packets that are discarded by the PBC). To give the reader the intuition about the relative percentage of FP and FN evaluation, let us make an analogy. Suppose you want to assess the quality of a new pregnancy test. You select a set of 1000 women. You know that 100 of them are pregnant, according to an oracle. The new test returns N=120 positive results. Of those, 80 are actually referring to pregnant women. Therefore FP=120-80=40 are false positive tests and FN=100-80=20 are false negative tests: computing the percentage of FP and FN, you have $FP\% = 100 \times 40/(1000 - 100) = 4.44\%$ and $FN\% = 100 \times 20/100 = 20\%$. In our scenario, we use the

		N	OK	FP	*FP%*	FN	*FN%*
PBC	E2E	1014	—	—	—	—	—
	E2O	163					
NBC	E2E	1236	726	510	*0.68*	288	*28.40*
	E2O	441	153	288	*0.38*	10	*6.13*
CSC	E2E	2781	984	1797	*2.40*	30	*2.96*
	E2O	161	157	4	*0.01*	6	*3.68*
NBC ∧ CSC	E2E	716	710	6	*0.01*	304	*29.98*
	E2O	147	147	0	*0.00*	16	*9.82*
TOT	≥ 100	76025	—	—	—	—	—
		487729					

Table 3: Results for UDP flows, CAMPUS dataset.

		N	OK	FP	*FP%*	FN	*FN%*
PBC	E2E	65	—	—	—	—	—
	E2O	125					
NBC	E2E	27437	50	27387	*73.73*	15	*23.08*
	E2O	295	124	171	*0.46*	1	*0.80*
CSC	E2E	191	57	134	*0.36*	8	*12.31*
	E2O	190	123	67	*0.18*	2	*1.6*
NBC ∧ CSC	E2E	51	49	2	*0.01*	16	*24.62*
	E2O	163	122	41	*0.11*	3	*2.40*
TOT	≥ 100	37212	—	—	—	—	—
		258634					

Table 4: Results for UDP flows, ISP dataset.

		CAMPUS	ISP
NBC	E2E	20910	60
	E2O	2034	646
CSC	E2E E2O	403996	46876
NBC ∧ CSC	E2E	621	12
	E2O	313	0
TOT	≥ 100	1646424	108831
		23856424	1614553

Table 5: Results for TCP flows, both datasets.

PBC as oracle, so that flows that pass the PBC classification form a reliable dataset. We refer to this set as the *benchmark dataset*. In particular, this dataset is built by Skype voice flows considering the E2O case. In the E2E case, voice, video, data and chat flows are present, since it is impossible to distinguish among them from packet inspection. Our tests are the NBC, the CSC and the joint NBC-CSC classifiers. Notice that the NBC test is expected to fail when a video/data/chat benchmark E2E flow is tested.

From a preliminary set of experiments on the testbed traces, containing more that 50 Skype voice calls, we tuned the PBC and CSC classifier thresholds to $B_{min} = -5$ and $\chi^2(\text{Thr}) = 150$, respectively. Using such choices, further discussed in Sec. 5.2, all flows were correctly identified as E2E or E2O, and neither FP nor FN were identified. Using the same threshold setting, we then apply the classification to real traffic traces: the results are summarized in Tab. 3, 4 and 5. For each dataset, the number N of flows identified by the different classifiers is reported, splitting E2E and E2O cases. Considering UDP case, Tables report also the correcly identified flow number (OK), FN and FP absolute numbers and relative percentages. Finally, last row reports the total number of flows, highlighting the amount of eligible flows.

Let us focus on the CAMPUS dataset reported in Tab. 3, in which the PBC identifies 1014 E2E flows and 163 E2O flows: this is the benchmark dataset. When the NBC classification is used in isolation, while the classification is very conservative (% NBC-FP= 0.68), there is an important number of discarded flows (% NBC-FN = 28.40): by manual inspection, we observed that most of the latter flows exhibit a bitrate much larger than the typical voice bitrate, hinting to video/data transfers being classified as E2E by the PBC. Indeed, as neither video nor data transfers are allowed toward the PSTN gateway, the percentage of NBC-FN is significantly smaller in the E2O case. Considering now the effectiveness of the CSC alone, we observe that in the E2E case the percentage of CSC-FP is quite large (2.40%). This phenomenon is due to Skype signaling flows that (correctly) passes the CSC test. In the E2O case, no signaling flow is present, yielding thus a negligible CSC-FP. Finally, looking at the percentage of CSC-FN, less than 3% of flows that are valid Skype flows are discarded, which is possibly due to a conservative $\chi^2(\text{Thr})$ setting. When the CSC and NBC are combined, the percentage of FP drops to almost zero (indeed, no FP are identified at all in the E2O case, and only 6 flows are FP in the E2E case) yielding thus to a conservative classification engine. The percentage of FN is 29.98% (9.82%) considering E2E (E2O) flows: based on previous remarks, the CSC/NBC combination allows to discard video and data transfers, and correctly identify only those *Skype* flows that actually carry *voice* traffic.

A similar reasoning applies to the ISP UDP dataset reported in Tab 4. Few Skype calls are found by the PBC classifier, possibly due to the flat rate offered to phone calls by the ISP. It is worth noticing that the NBC (correctly) identifies 27437 *voice* flows, most of which correspond to actual ISP's VoIP flows carried over RTP. Only combining the CSC allows to detect the true Skype voice flows. These results confirm that the NBC-FP may be due to non-Skype VoIP applications that generate packets with the same packet size and inter-packet gap characteristics. The significant percentage of E2E NBC-FN is due to the fact that the NBC is tuned to only identify voice flows. It discards E2E flows that carry video, data and chat services. As previously stated, this is due to the fact that the NBC cannot isolate video/chat/data flows from E2E flows.

Finally, we consider flows carried by TCP reported in Tab. 5. No PBC cross-checking is possible in this case. Considering the CAMPUS dataset, the NBC identifies almost 21000 (1.27%) flows as possible voice flows. The CSC instead identifies more than 400.000 (24.54%) flows as "random-payload" flows: this is not surprising, since applications relying on TCP usually transport raw/compressed data with no deterministic header at the segment start. The joint NBC/CSC usage reduces Skype identifications to only 621 E2E and 313 E2O flows. By manually inspecting those flows, we notice that 258 TCP flows are using server port equal to 443, which is suspicious as the port it associated to the HTTP over TLS service [16]. However, we verified that those flows were true Skype voice flows since, i) Skype uses port 443 to pass firewalls, and ii) TLS has a 2 Bytes long deterministic header that would make the CSC test fail. Similarly, 52 TCP flows are associated to the port 22 (i.e., the SSH protocol [17]): we suspect these flows to be False Positives, as the CSC cannot be trusted in this case and Skype is not known to use port 22. No other suspicious port number can be identified: moreover, most of TCP flows identified as Skype voice flows are terminated at the Campus WiFi NAT-box IP address, hinting with a very high probability to Skype being used under restrictive network policies. We can conclude therefore that in the TCP case, the percentage of FPs is negligible, since the 52 potential FPs have to be weighted against more than 1.64 million of connections.

From the above analysis, we can conclude that

- the NBC is very effective in identifying *all voice traffic* over IP, independently from the application;

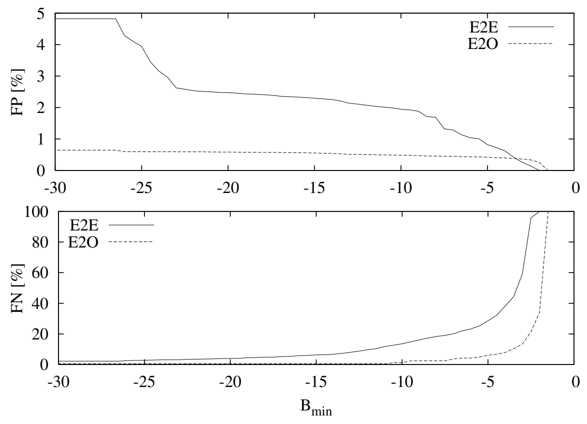

Figure 7: Percentage of FPs (top plot) and FNs (bottom plot) for E2E and E2O flows versus B_{min}.

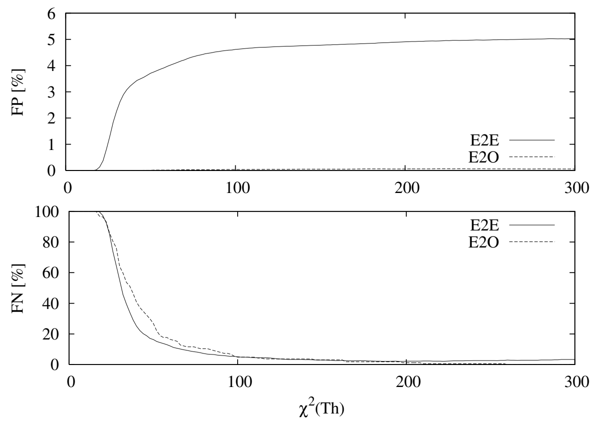

Figure 8: Percentage of FPs (top plot) and FNs (bottom plot) for E2E and E2O flows versus $\chi^2(\mathrm{Thr})$.

- the CSC is very effective in identifying *all Skype traffic* over UDP; when dealing with TCP flows, it identifies all encrypted or compressed traffic, which however corresponds to a quite large fraction of the traffic;
- the joint usage of NBC and CSC is very effective in detecting *Skype voice traffic* over UDP or TCP: almost zero False Positives are identified, yielding to a conservative identification, and few percentage of False Negatives are left out.

5.2 Parameter Tuning

In this section we investigate the sensitivity of the NBC and CSC classifications to the specific setting of the thresholds used in the decision processes. Sensitivity analysis is performed on the reliable benchmark dataset, gathered applying the PBC classification to the CAMPUS UDP traffic.

We start by considering the key of the NBC decision process, i.e., the minimum threshold B_{min} that both inter-packet gap and message size beliefs must pass for the flow to be classified as voice. When B_{min} is small (i.e., large negative value), the NBC is not strict in requiring that the observed characteristics are compliant with the Skype voice source model (yielding to NBC-FPs). Conversely, when B_{min} is large (i.e., tends to zero), the NBC requirements are tight, and some true Skype flows may be discarded (NBC-FNs). The impact of the threshold B_{min} on the number of False Positives and False Negatives is shown in top and bottom plots of Figs. 7 respectively. Percentages are shown by separately considering E2E and E2O flows classes. Observe that the NBC is more robust and less sensitive to B_{min} when it is applied to E2O traffic. Indeed, the presence of other services makes the benchmark dataset includes E2E flows that are not voice flows (the dataset includes video calls, chat messages, file transfers, etc.): such flows are possibly correctly discarded by the NBC but accounted for as False Negatives in the graph. so that even for very small values of B_{min}, the percentage of False Positives is quite small, confirming that our stochastic characterization is accurate. According to these graphs, B_{min} could be reasonably set between -8 and -3 (our choice is $B_{min} = -5$).

Considering now the CSC, we assess the impact of the threshold $\chi^2(\mathrm{Thr})$ on the classification results. From top plot of Fig. 8 we gather that, by increasing $\chi^2(\mathrm{Thr})$, the classifier requirements become less tight, and the number of E2E CSC-FPs increases; however, the increase is very steep and for $\chi^2(\mathrm{Thr}) > 100$ the number of False Positives does not significantly increase any more. Notice that almost no False Positive occurs for E2O traffic, since, as was already noticed in Fig. 3, the value of χ^2 for deterministic groups (as those in the header of E2O flows are) is very large and easily distinguishable by the CSC from non-deterministic groups of bits. Bottom plot of Fig. 8 shows that also the CSC-FN curve is steep: a very small value of the threshold induces the CSC to discard most of the flows. Flows start to be properly classified when the threshold increases beyond 50. The slight CSC-FN increase for larger values of $\chi^2(\mathrm{Thr})$ that can be observed in the E2E curve is due to the misclassification of mixed groups in the E2E message header: if the threshold is large enough, these groups are classified as random and not mixed groups, and the flow is discarded. From the joint analysis of these two graphs, we can conclude that correct setting of $\chi^2(\mathrm{Thr})$ is between 100 and 200 (our choice is $\chi^2(\mathrm{Thr}) = 150$).

So far, we have investigated the robustness of the classifiers in *isolation*: we now intend to analyze the effect of their joint usage, i.e., when a flow is classified as generated by Skype only upon agreement of both CSC and PBC. Focusing again on the benchmark dataset, we explore a number of combinations of B_{min} and $\chi^2(\mathrm{Thr})$ in terms of the obtained number of False Positives and Negatives for E2E and E2O cases. Left plot of Fig. 9 reports the FN percentage for E2E flows, where the thick solid line is obtained for $\chi^2(\mathrm{Thr}) = 150$. It is worth recalling that the quite large number of FNs derives from the fact that data/video/chat flows are considered as E2E flows in the PBC benchmark dataset: as these flows differ significantly from voice traffic, they are correctly discarded by the NBC. We notice that small values of $\chi^2(\mathrm{Thr})$ induce large numbers of False Negatives, because true Skype flows are erroneously discarded by the CSC classifier (and furthermore these CSC-FNs add to the NBC-FNs). Similarly, for the E2O case depicted in right plot of Fig. 9, although with a smaller magnitude in reason of the absence of data/video/chat heterogeneity.

Finally, we plot the False Positives detection triggered by the join usage of NBC+CSC in Fig. 10, which testifies the conservativeness of the joint classifier. Indeed, the FP number is extremely low, about one over ten thousand flows: furthermore, with a proper threshold choice, practically no flow is erroneously classified as being Skype. E2O case is not reported as only either 1 or 0 FP occur whatever setting. These results, beside highlighting the effectiveness of the joint use of NBC and CSC classifiers, also confirm the

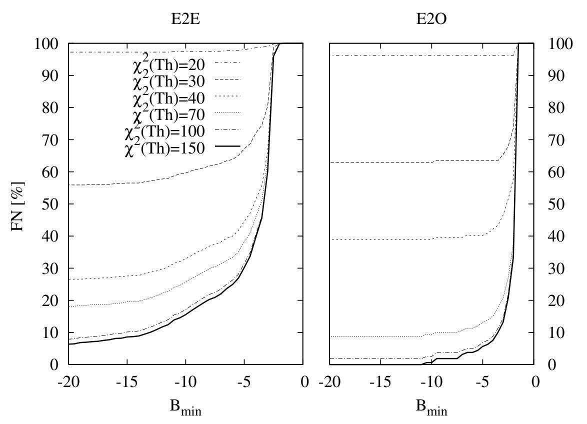

Figure 9: FN percentage for E2E (left plot) and E2O (right plot) cases versus B_{min}, and different $\chi^2(\mathrm{Thr})$.

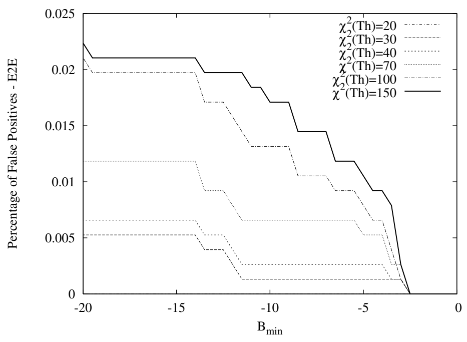

Figure 10: Percentage of FPs for E2E flows versus B_{min}, for different values of $\chi^2(\mathrm{Thr})$.

validity of our initial $B_{min} = -5$ and $\chi^2(\mathrm{Thr}) = 150$ thresholds choice.

6. CONCLUSIONS

This paper tackled the problem of the identification of Skype voice calls. We presented two classifiers to reveal Skype traffic from aggregate streams of packets. The first approach exploits the statistical properties of message content to let patterns and structures naturally emerge. The second approach instead makes use of Naive Bayesian techniques to match the stochastic characteristics of voice traffic generated by Skype sources. Our experimental results show that the combination of the above techniques is effective in both discriminating the voice streams from the traffic aggregate, and in further identifying the application, i.e., Skype.

The performance of such classifiers has been compared and cross-checked with those achieved through a traditional deterministic approach, based on deep-packet inspection. The design of the payload-based classifier stems from a partial, and incomplete, reverse engineering of the Skype messages. Results show that the joint use of the statistical classifiers outperforms the payload-based technique, as the kind of information that can be exploited yields to a more robust classification. Negligible False Positives are detected, and very few False Negatives are left out by the proposed approach.

While the Bayesian and the payload based classifiers follow traditional design, the Chi-Square test leverages on randomness introduced by the obfuscation techniques used by Skype. We believe that this last approach can be successfully extended to the more general traffic classification problem.

7. ACKNOWLEDGMENTS

This work was supported by the Italian Ministry of University, Education and Research (MIUR) under the PRIN MIMOSA. We would like to thank FastWeb for allowing us to monitor their backbone links, and the Sigcomm shepherd for her valuable suggestion.

8. REFERENCES

[1] Skype web site, `http://www.skype.com`

[2] Hesiod, "Theogony," ca 700 BC

[3] S. A., Baset, H. Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol." *IEEE Infocom'06*, Barcelona, Spain, Apr. 2006.

[4] P. Biondi, F. Desclaux, "Silver Needle in the Skype." *Black Hat Europe'06*, Amsterdam, the Netherlands, Mar. 2006.

[5] S. Guha, N. Daswani and R. Jain, "An Experimental Study of the Skype Peer-to-Peer VoIP System", *5th International Workshop on Peer-to-Peer Systems*, Santa Barbara, CA, Feb. 2006.

[6] K. Ta Chen, C. Y. Huang, P. Huang, C. L. Lei "Quantifying Skype User Satisfaction", *ACM Sigcomm'06*, Pisa, Italy, Sep. 2006.

[7] K. Suh, D. R. Figuieredo, J. Kurose, D. Towsley, "Characterizing and detecting relayed traffic: A case study using Skype.", *IEEE Infocom'06*, Barcelona, Spain, Apr. 2006.

[8] M. Carson, D. Santay, "NIST Net: a Linux-based network emulation tool." *ACM SIGCOMM Computer Communication Review*, V.33, N.3, July 2003, pp:111-126.

[9] GlobalIPSound web site, `http://www.globalipsound.com/`

[10] T.Berson, "Skype Security Evaluation." Online Technical Report, `http://www.skype.com/security/files/2005-031securityevaluation.pdf`, Oct. 2005.

[11] D.S.Sivia,"Data Analysis: A Bayesian Tutorial." *Oxford University Press*, Sep. 1996.

[12] A.Moore, D. Zuev, "Internet Traffic Classification Using Bayesian Analysis Techniques." *ACM SIGMETRICS'05*, Banff, CA, Jun. 2005.

[13] FastWeb web site, `http://company.fastweb.it/`

[14] M.Mellia, R.Lo Cigno, F.Neri, "Measuring IP and TCP behavior on edge nodes with Tstat", *Computer Networks*, Vol. 47, No. 1, pp. 1–21, Jan 2005

[15] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", *RFC3550*, Jul. 2003.

[16] E. Rescorla, "HTTP Over TLS", *RFC 2818*, May 2000.

[17] S. Lehtinen, C. Lonvick, "The Secure Shell (SSH) Protocol Assigned Numbers", *RFC 4250*, Jan. 2006.