Skype Traffic Identification

Marcell Perényi, András Gefferth, Trang Dinh Dang and Sándor Molnár
Budapest University of Technology and Economics,
Department of Telecommunications and Media Informatics, Budapest, Hungary
Email: {perenyim, gefferth, trang, molnar}@tmit.bme.hu

***Abstract* – Skype uses strong encryption to secure communication inside the whole Skype network. Clients choose communication ports randomly. Therefore, traditional port based or payload based identification of Skype traffic is not feasible. In this paper we introduce a novel flow dynamics based identification method to discover Skype host and voice calls as well. Our method only uses packet headers and the extracted flow level information. The whole identification process is scripted in Transact-SQL, thus it can be executed automatically. We also present the validation of the algorithm together with some analyses of the identification results.**

***Keywords* – Skype, traffic identification, analysis**

I. Introduction

Skype is a P2P VoIP network. It allows users to initiate and receive voice calls to/from other Skype (or even PSTN) users. In addition, instant messaging (chat) and file transfer is also possible within the Skype infrastructure.

Our goal is to analyze Skype traffic from a network operator point of view. Network operators might be interested in the nature of the traffic carried by their networks in order to optimize network performance and forecast future needs. They also want to identify and study popular applications and services for marketing purposes.

Skype traffic needs to be identified first. This is not trivial, since there is no unique standard port for Skype traffic, the protocol is not public, the data is encrypted and different software versions behave differently. Furthermore, the Skype binary uses a variety of techniques to prevent reverse engineering [8].

Several attempts for Skype identification are made. Ehlert et al. [1] describe a method for identification using traffic patterns and payload information from Skype login phase. For efficient blocking Skype activity has to be identified. Methods for blocking are published (see e.g. [2]), but these are tailored to a given firewall type or security setting and assume that Skype communication starts after enabling the blocking mechanism. Suh et al. [7] present a method for the detection of relayed traffic by comparing input and output traffic patterns.

Guha et al. [3] present some results about Skype usage patterns. Their results are based on active measurements and provide global information about the number of clients, the number of super nodes (see next Section) and general traffic patterns of a single Skype session.

Kuan-Ta Chen et al. [4] describe an identification method for relayed Skype flows. Some of the characteristic flow properties they examine to select Skype voice sessions are similar to that of ours. However, they aim to detect relayed flows only, and use the collected data for investigating correlation between call duration and voice quality.

In our approach we aimed at detecting Skype traffic even if the Skype client was started before the traffic measurement, in which case we cannot rely on some typical login traffic patterns or payload information. We also wanted to detect all Skype traffic regardless of software version and to avoid the use of payload information which is often not available.

II. Skype Overview

Here we give a brief overview of Skype focusing on those aspects that will be needed for our detection algorithm. A detailed description can be found, for instance, in [5] and [9].

A. Skype Components

The Skype P2P network consists of the following elements: ordinary nodes (clients), super nodes (SNs), login servers, update servers and buddy-list servers.

An ordinary node is a leaf-node of the Skype overlay network; it is the equipment of the user that is used for the communication. SNs are the switching elements in the overlay network responsible for maintaining a Global Index distributed directory which allows users to find each other. Each SN keeps track of a small number of ordinary nodes. SNs can also function as ordinary nodes, and in fact every ordinary node with public IP address and sufficient capabilities (free CPU, memory, bandwidth capacity) is a candidate to become a SN.

The login server stores the account information of users. It is responsible for user authentication at the beginning of the session, i.e. when the client is started. According to our observation there are several login servers storing information in a distributed way.

The update server (212.72.49.131:80) is also contacted by the client after it starts to check whether a newer version of the software is available.

The so-called buddy-list server [5] is responsible for storing the contact list of the users. Although this list is stored locally on the host computer, the role of this entity is to make sure that the contact list is also available if the user logs on from another host. Therefore, this server is contacted when changes in the contact list occur, or the user is logged in from a

different host than the previous time. Known buddy-list servers include 212.72.49.142 and 195.215.8.142 [1].

B. Skype operation

When a Skype client is launched it tries to establish a connection with a SN. For the duration of the session this SN will be responsible for the client. The client first contacts several (about 20) SNs to investigate whether they are alive and ready to accept the client. After some message exchange the client selects one of the SNs to establish a connection.

An initial set of SNs (*bootstrap* SNs) is hard-coded in the executable. These are operated by Skype. Upon the first run of the client these SNs are contacted, but a list of other SNs is retrieved and stored by the client. When the client is started for the next and all subsequent times the stored SNs are used.

At startup the client also contacts one of the login servers for authentication. Although there are several login servers, we have found that Skype clients in our measurements always connected to either of the following two servers: 212.72.49.141 or 195.215.8.141. The connection to the login server might be relayed by a SN, for example when direct connection is blocked, but the connection might be relayed even if the direct communication would be possible.

When the user is on-line, there is a periodic message exchange between the host of the user and the selected SN.

III. Skype identification

Although the application-layer protocol of Skype is concealed, we can still monitor the network and transport layer protocols and analyze the used IP addresses and ports. The statistical characteristics of the Skype data flows and packets can be studied as well, including flow bandwidths, packet sizes and several other properties. Our proposed identification method is based on these observable open parts of the Skype communication.

Unfortunately, the regular check for software updates does not guarantee that every client runs the latest version of Skype, therefore we need to deal with the behavior of different versions of clients. Although we did not have a chance to analyze each client, we based our identification algorithm on those properties which seem to be invariant amongst different software versions. In this section we first present a method to detect Skype activity even if no calls are made, then we present our method for the detection of Skype voice calls.

C. Filtering out known applications

The first step of our algorithm is to identify traffic of known, non-Skype applications. A database of popular applications is used, which contains default TCP and UDP communication ports for known applications. TCP port 80 is considered as an exception, since besides HTTP it is also used by Skype.

D. Skype specific connections

In the next step we look for Skype-specific connections, such as the connection to a login server, buddy-list server or bootstrap SN. The occurrence of any of these infers the presence of Skype, since these connections are very unlikely to be initiated by non-Skype applications. On the other hand these connections are not necessarily present (or visible) during a Skype communication:

The **connection to the login server** is in some cases relayed through a SN and therefore invisible.

The **connection to one (or more) of the bootstrap SNs** is necessarily attempted at the first execution of the application, but during subsequent executions the host may choose to contact other SNs.

The **buddy-list** server is contacted only in the cases mentioned in Section II.

There are also some kinds of connections which are not unique, but characteristic to Skype. We have found two of these:

The connection to the **update server** is initiated at the beginning of the session. However, the same IP and port is used, in some cases, to reach the www.skype.com web server.

A TCP connection to **port 33033** is likely to be initiated by a Skype client, since this is the default port for SN connections.

It is very unlikely that both of these connections are present if the host does not run a Skype client; therefore, we decided to conclude on Skype presence if both of these are found.

E. A method for Skype signal flow identification

It is possible that the user logged on to Skype before the traffic measurement began. In such a case we cannot detect login, buddy-list update or software update attempt. Furthermore, even if these connections can be observed they give no indication for the end time of the session. Therefore we investigated Skype network activity to find characteristic traffic patterns that last for the entire duration of the session.

A Skype client maintains one permanent connection to a SN while the user is logged on to the Skype network. At startup the client tries to establish several outbound connections to find an appropriate SN. After a few seconds these transient connections are terminated, and only one or two permanent TCP connections remain. One of these connections has a traffic pattern which can relatively easily be identified. Data packets are limited in size and both inbound and outbound flows (belonging to the specific TCP connection) have restricted bandwidth and packet intensity. In addition, the timing of outgoing packets follows a well-defined pattern. The connection persists as long as the user is logged on to Skype. Due to these properties we selected this flow to be scanned in order to identify Skype clients, and constructed a method to identify such flows.

In some cases we observed that the original signaling connection was replaced by a new one with exactly the same properties. It was possibly caused by the original SN going offline due to a failure or other problem. Nevertheless, the signaling flows are generally long enough to be detected. Long duration of the signaling flow is a key issue, since the proposed algorithm utilizes statistical properties among others.

Based on widespread analysis of Skype signaling connections we regard a TCP connection as a Skype signaling connection if it obeys all of the following rules:

1. Outbound and inbound data rate is not larger than 40 byte/sec,
2. Number of packets per second (in outbound and inbound direction separately) is not larger than 0.4 packets/sec,
3. Every packet in outbound direction is smaller than 1000 bytes (including IP and TCP headers),
4. Periodicity of 1 minute is observable for outbound packets of size between 70 byte and 250 byte. E.g. a certain percentage of packets (between 70 B and 250 B in size) arrive in a specific, periodic time slot.

The unique time behavior of signaling flows is caught by the 4th rule. We realized that most of the outgoing data packets are rather small except for some special packets. The exceptional packets have relatively much larger packet size (between 70 B and 250 B including IP and TCP headers), and an inter-arrival time of one minute. We assume that these packets are some periodic keep-alive messages of the client to the SN.

Unfortunately, the picture is not that clear: some factors make the identification more difficult. Occasionally, out-of-period packets appear in the outbound flows, a few of these packets fall into the interval of 70 and 250 bytes. Sometimes keep-alive messages happen to drop out, though periodicity is still preserved (no shifting). In some cases (perhaps when the user has few contacts on his/her buddy-list) the size of the keep-alive messages is not significantly larger than that of other packets in the outbound flow, which results in unclear separation of keep-alive packets. However, when the user initiates a voice call (becomes active), the size of keep-alive messages increases and falls into the specified interval in all investigated cases. After finishing the voice call the size of keep-alive packets usually returns to its original value.

We chose a simple method for detecting periodicity in Rule 4. The modulo 60 remainder is calculated for the arrival time (measured in seconds) of every packet (with packet size between 70 and 250 B), which is then rounded to the closest integer. This yields a time slot of 1 second. Then we calculate the distribution (histogram) of modulo 60 remainders, and mark every remainder where the frequency of the remainder in the histogram is over a certain threshold (0.08). The connections considered as Skype signaling are the ones for which only one remainder is over the threshold, i.e. only one remainder is significant. We chose this technique to detect periodicity, for it is adequate, fast, simple and easy to implement in SQL. However, other techniques could also be applied, e.g. DFT or wavelet transform.

Minor shifting (few ten milliseconds) was observed in the arrival times of the periodic keep-alive packets, which was most likely due to the inaccuracy of the internal clock of the host computer. This can cause two significant modulo 60 remainders (next to each other) to appear in the histogram. This problem can be avoided by shifting the arrival process by 0.5 second (adding 0.5 to each arrival time), and calculating the above mentioned histogram for this new process. If the periodicity is present, then at least either of the original and the shifted process will reveal it. We could only find few new signaling flows with this technique.

Although in theory other applications might generate similar traffic patterns, in our experience we could not find any. Therefore, if such a pattern is found we attribute it to Skype.

F. Method for Skype voice-traffic identification

Our final objective is to identify Skype voice calls. At the time of installation Skype chooses a random port as the default port for both TCP and UDP communication, thus port based voice traffic identification is not possible.

Skype prefers UDP as the primary transport protocol, and switches to TCP whenever UDP communication is restricted. It adapts quickly to changing network conditions by switching voice codec and transport protocol even in the middle of a call.

ISAC and iLBC codecs are used in both TCP and UDP cases. All codecs adapt their transfer rate and packet size to the available link capacity; consequently we can only set up a lower and an upper threshold as preliminary filter conditions for voice flows. According to our experiences the average voice packet size varies from 40 B to as high as 320 B, while a speech flow in one direction has a bandwidth of 20 Kbit/sec to 80 Kbit/sec. Therefore, we defined a loose upper bound of 400 B for packet size and 128 kilobit/sec for flow bandwidth. Flows failing to match any of these criteria are discarded.

In order to discover real Skype flows we had to find some more characteristic properties. Skype codecs have basically constant bit rate, even if the parameters of the codec, like packet size, bit rate, inter-arrival time, might be dynamically modified as a reaction to high delay, jitter or packet loss. The inter-arrival time of voice packets was either 30 ms or 60 ms in all measurements, which results in a packet rate of 33 or 16 packets per second respectively. In case of a TCP connection and obsolete Skype clients we also detected an inter-arrival time of 20 ms (50 packets per second). This property was confirmed by our measurements and by several other studies as well. In addition, the inter-arrival times of voice packets (together with other technical information including used codec) are visible in a popup during voice calls, when the appropriate setting is switched on in the client software.

Fortunately, the packet rate can be calculated and checked at flow level, knowing the arrival time, end time and the number of packets in the flow; hence flows not corresponding to this condition can be discarded. However, we cannot expect that packet rate will be exactly 33 or 16 for all Skype flows, which makes identification of speech flows more problematic. The main reason is that a voice call begins and ends not exactly at the time when the corresponding UDP (or TCP) connection starts and finishes. In addition, there is some transient behavior at the beginning of the session, when the bandwidth, packet rate (and packet-size) differ significantly from the properties in steady state. Furthermore, the end of

UDP "flows" is not well-defined, but only indicated by a timeout. After the call has finished there are still "stray" UDP packets transferred, which makes it difficult to accurately determine the end of the flow. To make things worse, the used codec and the occupied bandwidth might change during a call when necessitated by the changes in network conditions. Apart from this, packet rate is still a suitable property to decrease the number of candidate speech flows. A rate of 13 packets/sec is chosen as a lower bound and 53 packets/sec as an upper bound. Flows not corresponding to the packet rate condition are discarded. For all these reasons flow-level properties are not enough for the recognition of Skype speech flows, and the identification method should include some packet-level characteristics as well. We found inter-arrival time as the most characteristic property. We calculate the distribution (histogram) of inter arrival times for each remaining flow and mark the highest value (the main mode) of the histogram.

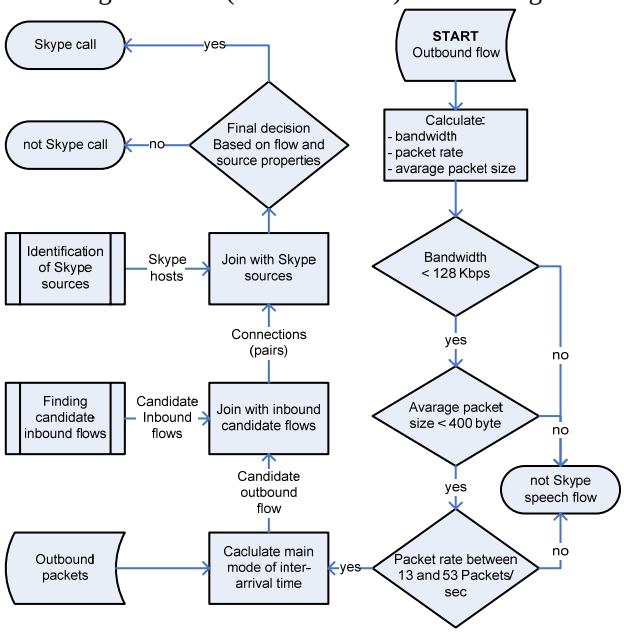

Figure 1. Flowchart of Skype speech flow identification method

Afterwards, inbound and outbound flows are paired to one another to create voice connections (sessions). The terms of pairing are the following: arrival time and end time of inbound and outbound flows are required to be close to one another, and also source address, source port, destination address and destination port should correspond to each other.

In Skype it is possible that the inbound and outbound directions of a voice session are served by different TCP connections. In this case the similarity of source and destination ports is not required.

In the last step those connections are selected for which the main mode of both inbound and outbound flows has a value of 20, 30 or 60 (ms), and source IP is among the previously identified Skype hosts. The whole identification process is shown in Fig. 1.

It is possible that some non-Skype flows meet some of the conditions. However, it is unlikely that flows other than Skype (even flows generated by other VoIP applications) meet all the conditions. In addition, the list of Skype sources, which was identified in a previous step, is also used to avoid misdetection.

IV. Traffic Measurement

Two traffic measurements were conducted; the summary of the data sets is presented in Table I.

The first measurement (called *Callrecords 2*) was carried out at one of the largest Internet providers in Hungary in April 2006. In the chosen network segment the traffic of about 1000 ADSL subscribers is multiplexed before entering the ATM access network. The logging was performed in one of the routers at the border of the access and the core networks. Further details of the measurement configuration are presented in [10].

In the second measurement (*Verification*) the traffic of our university department was logged, carrying the traffic of about one hundred users. We performed this experimental traffic logging to validate our Skype identification method.

In both measurements only IP and TCP/UDP headers were logged. Flow level information was extracted from the traces including source addresses, ports, packet number, transmitted bytes, start time and end time of the flow. Packet level information (packet size, packet arrival-time) was also preserved and used for the identification.

TABLE I.
DATA SETS USED FOR SKYPE TRAFFIC IDENTIFICATION

Data set	Time of measurement From - To	Number of flows	Total traffic (GB)
Verification	07. 11. 2006 10h 08. 11. 2006 16h	1 663 752	61.42
Callrecords 2	25. 04. 2006 11h 26. 04. 2006 11h	36 896 516	766.02

Both inbound and outbound traffic were logged, since data from both directions is necessary for accurate identification. However, our method can also be applied if only one direction is available, but the reliability decreases, since inbound and outbound speech flows cannot be paired to each other. Therefore, we recommend using our method in edge routers, where inbound and outbound traffic flows are carried through the same router. This is not necessarily true in backbone routers due to asymmetric routing.

V. Validation

The validation of the identification algorithm raises a couple of questions. For an exhaustive validation of our algorithm we need a large number of verified Skype signaling and voice flows from several clients. It is not easy to build such a managed environment.

The purpose of this validation is to verify the parameters of the identification method. These parameters were determined based on several local measurements on single computers in

different types of network environments (e.g. LAN access, ADSL, dial-in access, etc).

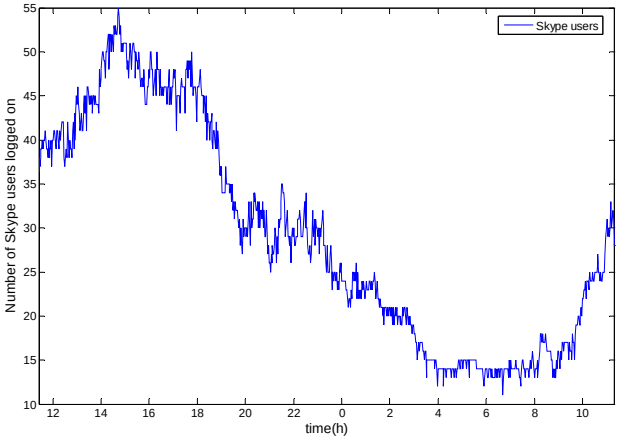

Figure 2. Daily fluctuation of the number of Skype users based on signal flow activity

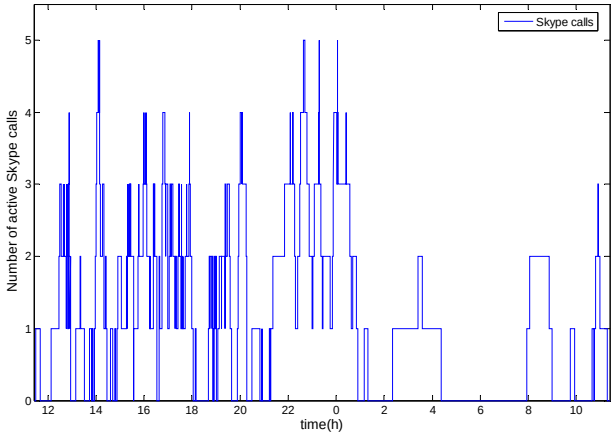

Figure 3. Daily fluctuation of the number of active calls in the network

We carried out an experimental traffic measurement in our university department. After the logging has finished, we interviewed all the colleagues whether a Skype client was running on their computer and whether they made any calls during the logging period. In addition, we also collected all the history logs of the clients, which contain exact information of the calls, e.g. date, time and call duration.

Then we applied our identification method to the experimental data set to detect Skype hosts together with Skype calls. Based on the comparison of detected Skype hosts and known Skype hosts from the user feedback we state that both host and voice call identification methods work well. Especially, the signal flow identification method got good marks: we could not observe any mistakes. Update connections were also detected in most of the cases. Login-, Buddy-list- and SN connection were rarely identified.

All Skype calls extracted from history logs were detected as well. We did not experience any false positive or false negative mistakes.

The validation study, however, cannot be considered as an exhaustive verification of the identification methods, since all Skype voice calls were made in an ideal network environment (100 Mbit Ethernet). Thus always the best-quality Skype codec was used by the clients.

VI. Traffic Analysis

In this section we present the results of our analysis of the *Callrecords 2* dataset.

In Fig. 2 the daily fluctuation of Skype users is presented based on recognizable signaling connections. Users not sustaining visible signaling connection cannot be taken into account, since only a single event (login, update, etc.) can be detected and we do not know when the user leaves the Skype network. Therefore the real number of logged-on Skype users can be somewhat higher.

We can realize that the number of Skype users logged on to the network follows the general daily tendency of the total number of users, which suggests that a certain ratio of users use a Skype client at home. Some users seem to keep their computer switched on during the night period.

The total number of active calls (Fig. 3) also follows similar daily fluctuation. Calls are coming more frequently in the daytime, though we can also recognize some surprising activity in the 1.00-6.00 AM interval, which suggests some “night birds” among the users or overseas calls.

The calls seem to be shorter in the daytime and definitely longer in the 9 PM – 1 AM period, which could be reasonable, because the users have more free time for chatting at night. However, we could detect only about 130 calls during the 24 hour period. For this reason, we do not want to draw far-reaching consequences.

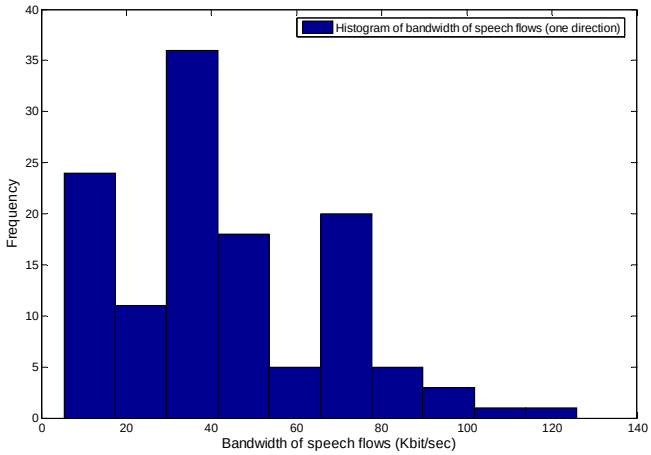

Figure 4. Histogram of the bandwidth of Skype speech flows in *Callrecords 2* dataset

There is only a small ratio of active Skype users who initiate calls indeed. Most of the users seem to prefer chat service or just to stay connected and reachable if needed.

The next two figures (Fig. 4, Fig. 5) show the bandwidth and the packet rate of the detected Skype calls. Fig. 4 shows

that the bandwidth of Skype calls is usually between 18 and 70 Kbps, typically around 40 Kbps. Fig. 5 shows three peaks in the histogram of the packet rate of Skype speech flows, which correspond to the typical three inter-arrival times (20, 30 and 60 ms). It can be seen that packet rates smaller than the typical ones (16, 33 and 50 packets/sec) also occur. The reason for this is that the termination of a flow cannot be determined accurately in some cases.

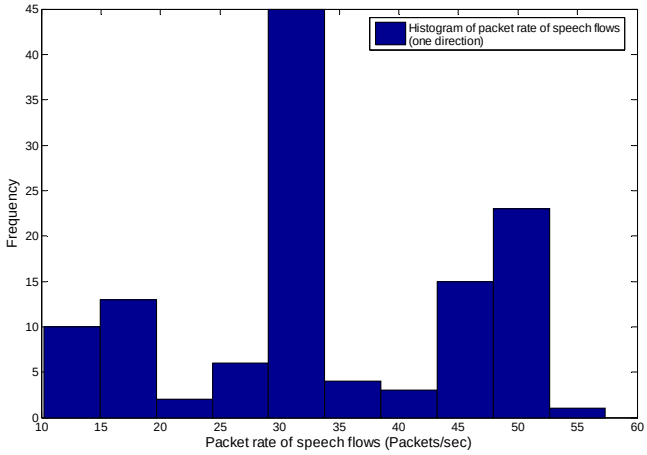

Figure 5. Histogram of the packet rate of Skype speech flows in *Callrecords 2* dataset

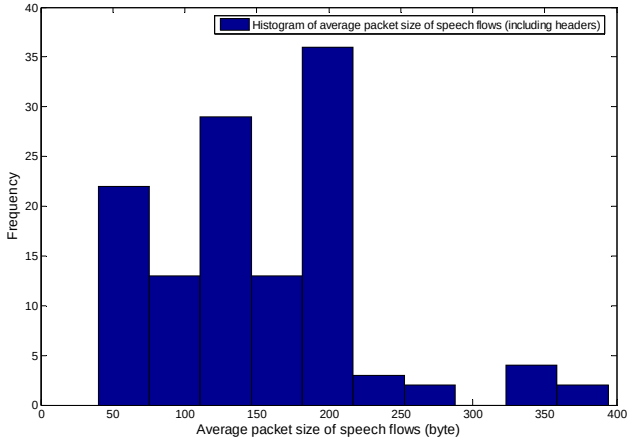

Figure 6. Histogram of the average packet size of Skype speech flows in *Callrecords 2* dataset

The average packet size of Skype speech flows is plotted in Fig. 6. The figure shows that the typical packet size (including IP and TCP/UDP headers) is somewhere between 100 B and 200 B, which is also confirmed by our test measurements. Smaller packet size – and bandwidth – occur in one direction when separate inbound and outbound TCP flows belong to the call.

Fig. 7 shows the histogram of the duration of Skype calls. Due to the few sample (few calls) it is hard to determine the exact distribution, but it seems to be an exponential-like distribution.

VII. Conclusion

We proposed a novel Skype identification algorithm based on observable parts of Skype protocol. First, candidate Skype hosts are detected using traditional IP and port-based identification together with a special signaling flow identification method. Then Skype calls are discovered exploiting the properties of speech flows, timing of voice packet and candidate hosts found in the first step. The algorithm uses only packet headers and the extracted flow-level information, but no packet payload is necessary. It expects logged (offline) data as input.

We also presented the validation of the identification of the algorithm based on a test measurement in our department. In addition, we showed some analyses results of a 24h real data set from an ADSL domain in Hungary including properties of calls and daily fluctuation of Skype users and calls.

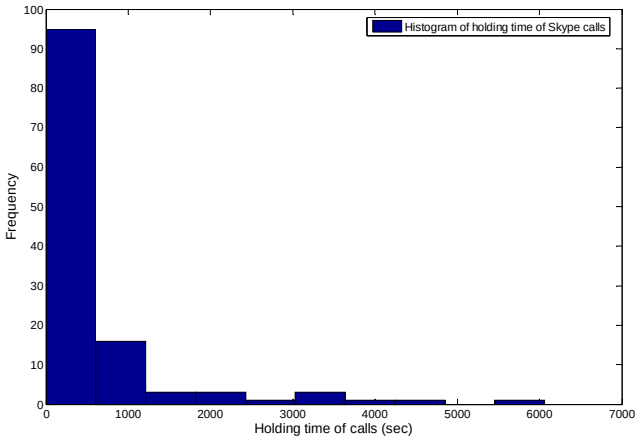

Figure 7. Histogram of the duration of Skype calls in *Callrecords 2* dataset

Acknowledgement

The authors are grateful to Ericsson Hungary Ltd for the financial support and to P. Varga and L. Kovács for their help in the traffic measurement.

References

[1] S. Ehlert, S. Petgang, "Analysis and Signature of Skype VoIP Session Traffic", *Technical Report NGNI-SKYPE-06b*, Fraunhofer FOKUS, Berlin, Germany

[2] W. Ghandour, "Blocking Skype Using Squid and OpenBSD", *Help Net Security (www.net-security.org)*, 2005

[3] S. Guha et al., "An Experimental Study of the Skype Peer-to-Peer VoIP System", *in Proc. of IPTPS'06*, Santa Barbara, USA, 2006

[4] Kuan-Ta Chen et al., "Quantifying Skype User Satisfaction", *in Proc. of SIGCOMM*, Pisa, Italy, 2006

[5] Skype Technologies S.A., "Skype - Guide for Network Administrators", 2005

[6] S. A. Baset, H. Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol", *in Proc. of INFOCOMM'06*, Barcelona, Spain, 2006

[7] K. Suh, et al., "Characterizing and Detecting Skype-Relayed Traffic", *in Proc. of INFOCOMM'06*, Barcelona, Spain, 2006

[8] Fabrice Desclaux, "Skype uncovered", *EADS*, 2005

[9] P. Biondi, F. Desclaux, "Silver needle in the Skype", *EADS*, 2006

[10] T. Dinh Dang et al., "On the Identification and Analysis of P2P Traffic aggregation", *in Proc. of Networking 2006*, Coimbra, Portugal, 2006