On the Stability of Skype Super Nodes

Anat Bremler-Barr[1] and Ran Goldschmidt[2]

[1] Interdisciplinary Center, Herzliya, Israel
bremler@idc.ac.il
[2] University of Haifa, Israel
ran.goldschmidt@gmail.com

Abstract. The heart of skype services, one of the most ubiquitous P2P networks, is based on a set of super nodes. Choosing stable SNs is an important task, since it improves the whole performance and quality of the P2P network [1, 2]. In this paper we shed light on the life cycle of SNs using extensive data sets on Skype Super nodes, which were gathered over a period of 3 months. We then suggest how to choose a more stable SNs set.
The dynamic of nodes is inherent to the use of a computer, which is unplugged for some time or is mobile. Hence it is natural to predict that a Super Node would have multiple sessions correlated with the time the computer is up. Surprisingly, we show that 40% of the Super Nodes have only one session, with median residual life time of 1.75 days. These nodes also have a significantly shorter lifespan than Super Nodes that have multiple sessions, which have median residual life time of 67.5 days. We propose and give evidence that nodes with one session are nodes with dynamic IP addresses, and hence they have ended their life cycle due to a change of IP address. We show that the nodes with multiple sessions are mostly nodes with static IPs, and that choosing super nodes with static IPs would increase the availability and stability of the P2P network significantly.

1 Introduction

In P2P network that uses Super Nodes (SNs), choosing stable SNs is an important task, since the super nodes serve as the control tier for all the P2P nodes. The dynamic nature of P2P networks, where there are consistent changes in the set of nodes that participate in the p2p service, poses a huge challenge: designing a reliable service, while the core of the service is based on dynamic set of super nodes. By choosing stable Super Nodes, the whole performance and quality of the P2P network can be improved [1, 2].

Our paper sheds new light on the dynamic nature of SNs. We have collected an extensive data set of SNs from the ubiquitous P2P network, Skype. In a short period of time of 15 minutes we have collected 10,000 active SNs and followed their life cycles over a a long period of 3 months. We show that the SNs enjoy longevity: median residual life of 22.7 days and with median session length of 3 days, where a session of a node is a consecutive time that a node is up. Surprisingly, we show that high percentage (40%) of the SNs has only one session. Inspired by this fact, we classify the SNs into two groups according to the number of sessions they have during their life time: Single Session in Life Time (SSLT) SNs group and Multiple Sessions in Life Time (MSLT) SNs group. We measure different parameters (life time, session length and availability) on the two groups, and found out that the nodes in the MSLT are more stable: with median residual life length of 67.5 days, while the SNs in the SSLT have median residual life of 1.75 days. Moreover we show that if we choose nodes only from the group of MSLT the P2P system would be more stable and have a lower churn (less by 19%) and higher accessability (higher by factor 2.1). We then show that SSLT is primarily composed from dynamic IP addresses. A dynamic IP address, changes its IP address from time to time (usually after it disconnects from its ISP). From the P2P point of view, an SN that changes its IP no longer exists in the P2P network and it is considered as a new SN node. The MSLT is primarily composed from SNs with static IP addresses, which are usually servers or part of academic networks and due to their primarily job need to be up most of the time and hence they are more stable.

While the dynamic nature of P2P network was extensively researched in many papers [2–8] our paper reveals new findings due to several reasons: the research is on super nodes and not on the regular peer nodes; we check the stability of the SNs over long time period; we measure Skype and not file transfer applications. Those differences may explain the main reason why the role of dynamic IPs was left in the shade until now. Most of the previous papers concentrate on file transfer p2p applications such as Gnutella, BitTorrent and Kad where the dynamic nature of nodes is more likely due to user activities such as shutting down the application after completing the task of downloading the file. And indeed, measurements on these networks observe sessions with length of a couple of hours [8]. However, in Skype usually the application is on all the time the computer is on (see Section 3), hence the dynamic of nodes is due to a network event, such as disconnecting the computer from the network. And indeed we measure median session duration of a couple of days. Moreover we measure subsets of the skype clients, the Super Nodes, which have relatively long lifes and the effect of the dynamic IP address which occurs in long time scale is shown. Note that those nodes were nominated by Skype to be Super Nodes. Even though the code of the Skype application is confidential and unknown, it is reasonable to assume that Skype tries to nominate peers that are more stables to be super nodes. The only previous paper that concentrated on Skype SN measurements by Guha *et al.*[9] did not concentrate on the stability of SNs, moreover its data was not suitable for measuring and understanding stability. That paper [9] checked only Super-Nodes that were alive after a period of 3 months from the time they were collected. Note that using our measurement we discover that only 20-30% of the SNs are still alive after 3 months.

One important conclusion from our work is a simple guideline of how to choose Super-Nodes. The previous technique[7], focuses on algorithms that take as input the history of the Super-Nodes. We show, that a key impact of stability of the super-Nodes is its IP address type. Super-nodes from static IP addresses have greater chance of remaining alive for a long period of time in the network, and hence should be preferred. While classifying the type of address (static or dynamic) using the IP address alone is a hard task, this is an easy task to the P2P client application, which can observe all the outgoing traffic and can detect the constant change in IP addresses and thus can conclude the type of the IP address.

2 The Model

In this section we model our P2P network, a partially centralized architecture[10] (i.e., P2P with SNs). The model suits the Skype P2P system model, but also suits other P2P networks with super nodes such as Kazaa[11], joost[12], iMesh[13], Morpheus[14]. The model is also applicable to fully distributed P2P networks, where we can consider each client as an SN.

The participating nodes in our P2P system are divided into two categories: Super-Nodes (SNs) and ordinary clients. The Super-Nodes create the control level, and basically are regular clients with good network connections that the P2P network decided to nominate to be SNs. Clients request control services[3] only through the super nodes. Each client maintains a list that contains a subset of the SNs, and when it wishes to connect the P2P network it picks one of the SNs in its list. Roughly speaking if none of the SNs in the list are active then the client cannot connect to the service. This is not completely accurate, since there are usually also bootstrapping servers that are located in the premises of the company that handles the P2P service. However, the P2P cannot rely on these servers due to the fact that these known servers are vulnerable to filtering attempts (the incentive of filtering P2P traffic is discussed in Section 5). Moreover the scalability of the P2P network is based on the fact that most of the clients are connect to the SNs and not to the main servers. A Super-Node is defined according to its IP and Port. At any given time an SN is in one of the following states: up (available) or down (fail). Nodes fail and recover according to some unknown process. A *session* of an SN is the continuous period of time the node is up.

We assume that the client's SN list is continuously updated with the "up" SNs as long as the client is connected to the P2P network. In the next section we show that our experiments with Skype clients reveal that this is the case with Skype.

[3] Such as connections to the P2P network, querying the P2P network on the IP of the callee and so on. After finding the IP address of the callee the caller communicates directly with the callee and initiates the call. If the client is unable to communicate directly with another client, then the SNs can also relay all communication, thus effectively bypassing firewalls.

3 Experiment Methodology

In spite of its massive popularity, little is known about Skype's inner-workings. Skype is a closed-source application and consequently, Skype Ltd. does not disclose its protocols and architecture. Extensive studies [10, 15–19, 9, 20–24] were conducted to disclose the Skype architecture, protocols and inner-workings by using reverse engineering and measurement techniques. The precise algorithm of how Skype chooses which clients to nominate to SN and which SNs would serve a specific client is unknown.

In this paper we use the fact that in Skype versions 2-2.5 each Skype client holds a list of up to 200 SNs and their connection ports in a specific XML file (*%appdata%\Skype\shared.xml*). This SN list is not encrypted in those versions, while in later versions Skype encrypts the SN list.

Our first goal is to measure the changes in the SN list of a regular client. In Figure 1 we measure the changes in the SN list over time. We took a snapshot of the list at time t_0 and then at each time unit t_i we calculate the percentage of SNs that did not change and appeared also in the original SN list at time t_0. We repeated the test on 100 Skype clients and for duration of 2000 minutes. Figure 1 (see line labeled "SN List Regular Updates") shows that the SN list is updated constantly and that after 2000 minutes, which is a little less than a day and half, only 64% of the original SNs still appear in the SN list. On average Skype client received an update every 20-30 minutes.

Motivated by the fact that the SN list is constantly changing, our next step is to measure the dynamics of SNs, specifically the on and off time of the SNs. We first collected a set of SNs from multiple clients at the same time. In this process, named by us *SN extraction*, we extract SNs at higher rate by repeatedly doing the following on a Skype client: 1. Extracting the SN addresses and ports from the XML file; 2. Flushing most of the SN addresses from the list - leaving only specific SNs 3. Restarting the Skype client and waiting until the list is refreshed with 200 SN addresses. Each such iteration, takes approximately 2-2.5 minutes. By implementing the described process at the same time on 20 computers, located at ETH and Israel, we gathered 10,000 SNs with unique IPs in less than 15 minutes.

The *SN extraction* process is a modification of the method used by Guha [9], designed to work at a faster rate. We estimate that we gathered a snapshot of around 20% of the active set of SNs since the common estimation is that there are around 45K active SNs at any give time [25]. Figure 1 (see line labeled "SN List Extraction") shows the percentage of SNs that also exist in the SN list at time t_i that appeared also in the original SN list at time t_0. After 2000 minutes, the percentage is negligible and is only 2.5%. Thus, the flushing phase at the *SN extraction* causes Skype to send almost entire new SN list.

In order to check the status of an SN (i.e., if the SN is up or down) we use a Skype application ping, i.e., we send to the SN the first UDP packet of the Skype login process (similar technique to [9]). We checked the 10,000 SNs, every 15 minutes, where in each such iteration, we pinged all the 10,000 SNs twice and waited for up to 5 minutes for the answer. An SN is considered to

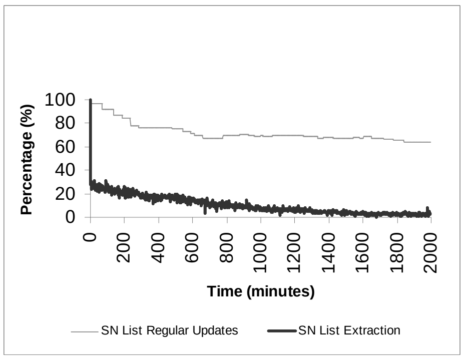


Fig. 1: The percentage of SNs at SN list at time t that appeared at the original SN list at time 0 in two scenarios: regular updates from Skype network and SN extraction

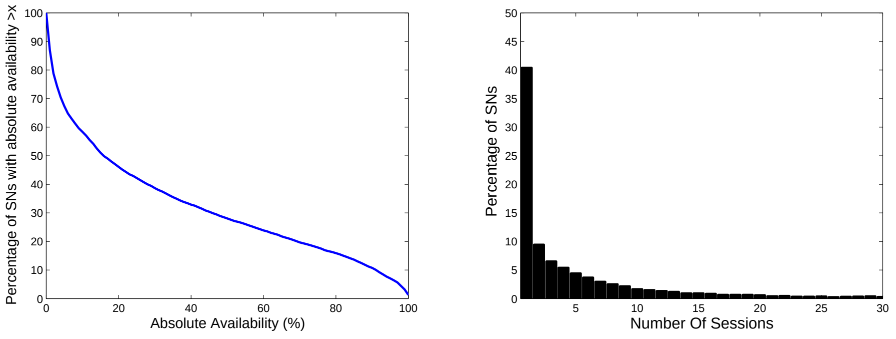
be up in iteration i if there was a response to at least one of the two pings in that iteration, otherwise the SN is considered down at iteration i. A *session* is defined as consecutive iterations where the SN is up. The SN life time is defined as the time elapsed between the start of the first session until the end of the last session. [4] We note that we verified that the Skype application ping is a good indicator of the fact that the client is still SN and is still alive. For dozens of SNs, we checked and verified that if the Skype application ping indicates that the SN is up, then we are able to connect to skype using this SN solely. [5]

Overall we did experiments over three months beginning on Apr 3. 2009. Our infrastructure was very stable and we conducted more than 9,000 iterations, and experienced connectivity problems only in 10 iterations. The stability and the large data set overcomes the known pitfalls in measuring the stability of SNs [8].

We note that there is a high correlation between the time the SN is up and the time the corresponding computer is up (and vise versa). Thus there is a good indication that usually the skype client is by default open at the client computer. We think that this is because the skype client is always on in order to be on standby to receive calls. The way we verified that a computer is on is by sending an ICMP ping to the computer. While most computers do not response at all to ICMP ping, since the ICMP is filtered in the host network for security reasons, a small subset of the SNs (17%) do respond to ICMP. We concentrated on this group and found that in those computers there is a decisive correlation between the responsiveness to the Skype application ping to the responsiveness to a regular ICMP ping (and vise versa).

[4] In order to avoid noises due to packet loss, we decided that a session ended only if at least in three consecutive iterations the SN was down. We found out that the results are not sensitive to this threshold.

[5] This was done by modifying the SN list at a client to hold only the examined SN and by filtering into the firewall any default hard coded skype servers



(a) (b)

Fig. 2: (a) CDF of the absolute availability percentage of SNs. (b) Histogram of number of sessions in the residual life time of SNs.

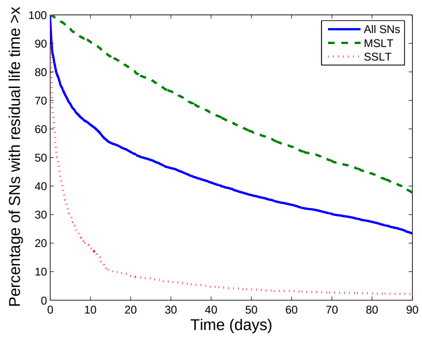
4 Availability and Life Cycle of SNs

In this section we shed light on the life cycle of SNs. Naturally, a P2P system wishes to choose SNs that are available most of the time. We start by checking the **absolute availability** of SN, which is defined as *the time the SN is up during the test.* Figure 2(a) shows the absolute availability of SNs and surprisingly the percentage of absolute availability is low: 50% of the SNs are available less than 18% of the test time, i.e. less than 16 days. In order to understand this low absolute availability we start to analyze the life of SNs. Due to the test methodology we can not measure the life time of SNs but only the residual life time (recall that those SNs were already alive for unknown time when we start to measure their activity). Our first step is to understand the number of sessions an SN has during its life. Figure 2(b) shows the histogram of the number of sessions in the residual life of SNs. It is obvious that the number of SNs with only one session seems to be an exceptionally large number of 40%.

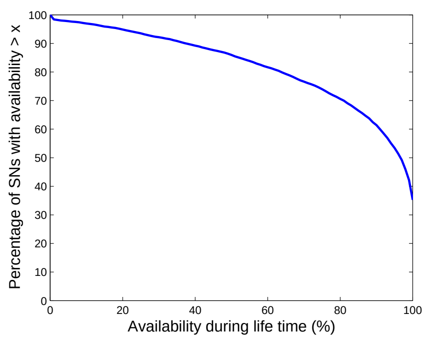
Motivated by this fact, and in order to further understand the behavior of SNs we define two groups of SNs: 1. The *Multiple Sessions in Life Time (MSLT)* SNs group - the group of SNs that have more than one session in their residual life time 2. Correspondingly, the *Single Session in Life Time (SSLT)* SNs group - the group of SNs that have exactly one session in their residual life time. [6]

Figure 3(a) shows the CDF of residual life time. One obvious outcome is that SNs that have only one session, have a very short residual life time. The median of residual life time at the SSLT group is 1.75 while the median of residual life time at the MSLT group is 67.5 days. SNs that were still alive when the experiment was ended (after 90 days) are clearly from the MSLT group: 38% of the MSLT SNs as opposed to 2.1% of the SSLT group. In the first 15 days

[6] We note that our definition of an SN session refers only to sessions which were made by the SN while holding the same IP+port.



(a)



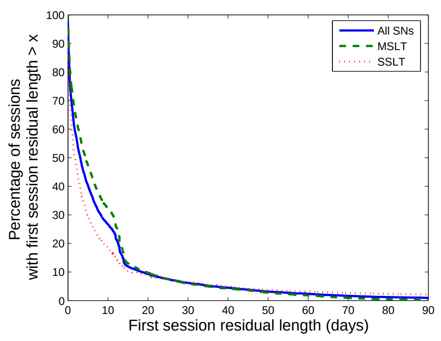
(b)

Fig. 3: (a) CDF of the residual life time. (b) CDF of the availability during life time of SNs.

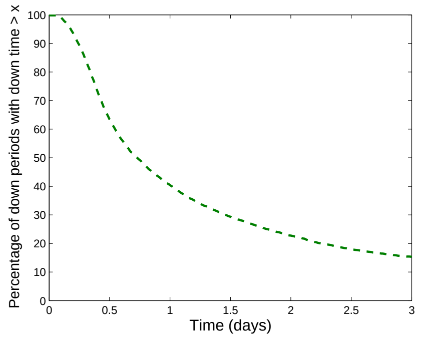
the number of SNs that died permanently is the highest, and most of them are from the SSLT group. From the graph shapes, it seems that the two groups have entirely different behavior, while the SSLT graph has exponential reduction in the first 15 days, the MSLT graph shows a steady linear reduction in all the test experiment period.

With this understanding we revisit the absolute availability definition and define new parameter the **availability during life** to be the *percentage of time a node is available during its residual life time.* Figure 3(b) shows that 90% of the nodes were available more than 37% of their life time. Note that, the SNs that were available 100% during their residual life, had only one session, i.e., they belongs to the SSLT group. Hence, we can now understand that the low measure of absolute availability of SN was mostly due to SNs that disappear permanently from the system, and not due to nodes that alternately leave and join the network.

Motivated to understand the life cycle of the SNs we continue and analyze the length of the residual session length and down time between sessions (which is naturally applicable only for the MSLT group). One may predict that the length of the session will be similar between the two groups: i.e., that the MSLT SNs would have multiple sessions but the length of the session would be the same. Figure 4(a), analyzes the first session residual length, and shows that this is not the case: the SSLT has also shortest residual session length of 1.75 days in the median as opposed of 4.35 days median for the MSLT group. We explain this phenomenon in Section 6 after discussing the roots of those two groups. Figure 4(b) shows that the down time is relatively low to the session time; where the median of downtime is 0.8 days while the median of residual session length is 3 days (for all SNs) and 4.35 for the SNs in the MSLT group.



(a)



(b)

Fig. 4: (a) CDF of residual length of the first session. (b) CDF of downtime.

5 Churn and Accessibility of the P2P System

In this section, we discuss the impact of the life cycle of SNs on the system stability. We also show the impact on the stability if we would choose SNs only from SSLT group or only from MSLT group.

The accessibility of the P2P system - This is a new metric we suggest (somewhat similar to the group availability parameter of [8]). Motivation wise, this parameter correlates to the ability of a client that has an SN list which is T time old to access the P2P network using one of the live SNs on its list. Specifically, let us take a snapshot of the M active SNs of the P2P system at time t_0, the accessibility at time T of the set M is the percentage of the M SNs that are up also at time $t_0 + T$. Hence, if a client has a list of SNs with K SNs which were obtained T time ago, and the accessibility of the P2P system after T time is p, then the probability that a client can connect to the P2P system is $1 - (1 - p)^K$. Using this definition if an SN fails (sometime before time T) and then recovers at time T, the SN is useable as a node that never failed.

The accessibility parameter quantifies the ability of a client to access the P2P network with an old SN list. The lower accessibility value the higher the rate which the P2P system needs to update the SN list to ensure that the client can connect the network.

We assume that the P2P system cannot rely on bootstrapping servers (if they exist), that are in the premises of the P2P company, since they are fixed and known and hence vulnerable to blocking attempts of the P2P services. There are various reasons to block P2P systems. In the case of a P2P worm [26], such as STORM, the motivation of blocking the attempt is clearly to mitigate the spread of the worm. In order to design resilient worm, the accessability to the SNs that appear in the infected message should be high also after a long time (since in some cases a long time might elapse between the infection by the worm to the actual time the worm executes).

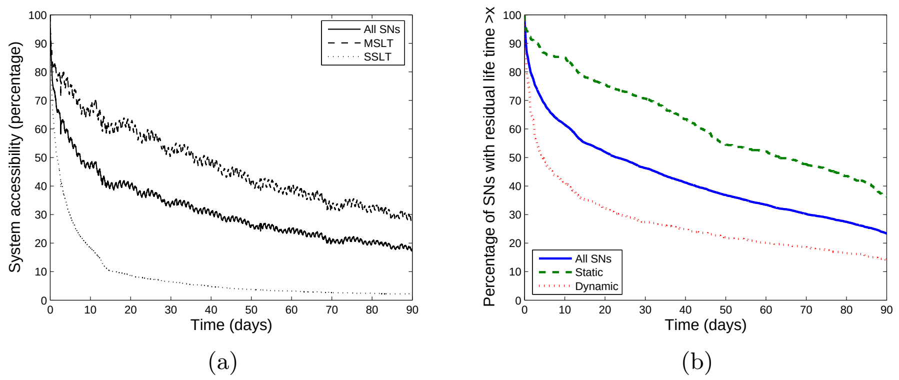


Fig. 5: (a) Accessibility of the SNs as function of time. (b) CDF of the residual life time of Static and Dynamic SNs.

Legitime P2P services, are also in danger of being filtered, especially at enterprizes, that are concerned with data leak using encrypted P2P services that offer file transfers (such as Skype). Without being drawn into the legality aspects, an ISP may wish to control the traffic of P2P services since the traffic consumes the ISP bandwidth. An ISP may also wish to filter or limit the rate of P2P services that compete with services offered by the ISP; for example, Skype may compete with ISP VoIP services.

In Figure 5(a) we show the percentage of accessibility to P2P network as a function of time elapsed from the beginning of our experiment. After 90 days of the test, the MSLT group has 38% accessibility, the SSLT group has accessability of 2.1% and the group of all the SNs has accessibility of 18%. I.e., choosing SNs only from the MSLT group would increase the acceptability by factor of 2.1. Note that there is no straightforward correlation between the accessibility of the system after T time and the life span and availability of the SN. Since if all the SNs had an availability of 50% and had a life span of 90 days it might still be that the accessibility of the system will be zero at fifty percentage of the time. In fact this can happen if all the SNs are correlated and in the same time zone: e.g, all of them are unavailable during the night. However, as Figure 5(a) shows this is not the case, and the graph shows only small waves, which we found were correlating to day and night zones in the USA. Overall the shape of the graph reassembles the graph of the residual life time (see Figure 2(b)). An analysis of the SNs origins according to countries, lead to the conclusion that this stability in accessibility is the direct outcome of the fact that the SNs are distributed over the whole world and over all time zones. We calculated the continents distribution of SNs that come from countries that contribute more than 1% to the total SNs and we have received that 38.24% are from North America, 29.92% from Europe and Africa, 16.02% from Asia and 15.82% that we did not classify. Hence we can conclude that the continent distribution of

SNs, guarantees that there is a high chance that a Skype client will be able to connect to the network even if it was not alive for weeks.

The churn of a P2P system - Motivation wise the churn measures the number of times an SN goes down and we need to replace the SN. Specifically, we assume for simplicity that the system maintains a fixed number of SNs, M SNs. The assumption is that when an SN fails, the system picks another one to replace it. In this definition when a node fails and then recovers, from the system's perspective the node is like a new fresh node. We define churn as the number of SN turnovers (where turnover is when SN went down and another node was assigned to replace it) over a period of time T divided by M. For example, a churn of 2 per day means that on average the identity of each SN is replaced twice a day. Churn significantly influences the stability of a P2P system. An SN that goes down, requires the system to transfer all its functions to another SN. For example, in Skype, if the SN relays calls, all the calls need to be transferred to another SN.

We can estimate the Churn rate from the first session length. Recalling that churn is defined as the number of times we replaced an SN in order to maintain a fixed number of live SNs. Hence, when the session ends, we need to replace the SN with a new one. Let X be the random variable of the first session length. Then we can estimate the churn as $T/E(X)$, where T is a given time unit. However, note that we measured in our experiment the residual first session length (as presented in Figure 4(a)), and not the actual first session length, since we start to measure the nodes at some random time of their life. Let X_r be the random variable of the residual life of the first session, then $E(X_r) = E(X * Y)$ where Y is a random variable that indicates a percentage of the session time that has already elapsed when we start to measure the node. We use here a simplified assumption that Y is uniformly distributed and hence $E(X) = 2E(X_r)$ [7]. Using this calculation we receive that the group of SSLT suffers from high churn of 0.35 turnovers per day compares to the group of MSLT with churn 0.22 turnovers per day. I.e., the SSLT churn is 1.66 higher than the MSLT churn. Moreover the churn of all the SNs group is 0.27, which means that choosing SNs only from the MSLT group will reduce the churn by 19%.

6 Dynamic addresses and the Correlation to SSLT group

Until now we have shown that there is a huge difference between the characteristics of SSLT and MSLT SNs (see summary of result at Table 1). We have also shown that if we choose the SNs only from the MSLT group it would improve the Churn and Accessibility of the P2P system dramatically. In this section we explain why these groups, SSLT and MSLT, are so different, and what the rational behind this partition into these two groups is.

[7] Due to the paradox of residual life [27]- this calculation is not entirely accurate however using a more complicated and accurate calculation would not change significantly the result.

	Residual life time	Residual first session length	System Churn	System Accessability
Total	22.7	3	0.27	18%
MSLT	67.7	4.35	0.22	38%
SSLT	1.75	1.75	0.35	2.1%

Table 1: Summary of the different characteristics of SSLT and MSLT SNs. The parameters are calculated using the 90 days of data. The residual life time, session length numbers are the median. The system accessability is calculated after the 90 days of the experiment.

	From All SNs	Static IP SNs	Dynamic IP SNs
Total	10,000	637	983
MSLT	59.7%	84.92%	38.55%
SSLT	40.3%	15.08%	61.45%
Ratio MSLT/SSLT	1.48	5.63	1/1.59

Table 2: Classification of the SNs according to their address type (Static/Dynamic)

We believe that the difference between SSLT and MSLT is inherent and it is related to the fact that some of the SNs belong to dynamic IP networks (usually residential users connected by cable, xDSL and so on..). In this case an SN in the SSLT group is mostly SN with a dynamic IP and hence died since the IP address of the SN was replaced. However, there is a good chance that this SN is alive but with a different IP address [8]. This can explain also the exponential reduction in the residual life of SSLT SNs in the first days (see Figure 3(a)), since dynamic IPs live for only a few days [28]. As opposed to SSLT group, the MSLT group is composed from SNs with static IPs, that after leaving the network (for example closing the computer), can return with the same IP.

In order to support our assumption, we classified the IP to Static and Dynamic IPs using reverse DNS (rDNS, similar to the method of [28]). An rDNS maps an IP to its host name (e.g., "ip-66-186-253-215.dynamic.eatel.net"). We classify the IPs by searching in the returned results for keywords such as "static" and "dynamic". Using this method, we were able to classify 637 static IPs and 983 dynamic IPs. Note, that the rDNS technique was able only to identify only 15% of the SNs but the classification is almost 100% accurate. The SNs that were classified using the rDNS technique are a good random sample of the SNs. Moreover, rDNS is consider to be the most accurate technique. Other solutions, e.g. Spamhaus are able to classify all the IPs but are known to be accurate only 70% of the time [28]. Using Spamhaus we reach a similar result.

[8] Dynamic IPs may occur also due to NAT or a cluster of proxies, however this is not relevant to Skype SN, since an SN can not be behind a firewall or NAT [9].

We present at Table 2 the correlation between the classification to SSLT and MSLT to Dynamic and Static IPs. It is clear that the vast majority, 84.92%, of the static IP SNs belongs to the MSLT group. With dynamic IPs group the result shows a weaker correlation with only 61.45% of dynamic IPs appeared in the SSLT group. Hence 38.55% of the dynamic IPs belong to the MSLT group. This is still a good indication that most of the dynamic IPs are from the SSLT group, if you take into account that the ratio between the number of SNs in the SNs group is 1.48 and for dynamic SNs group the ratio between MSLT to SSLT is 1/1.59.

We suspect that dynamic IPs that are in the MSLT group belong to a third group of IPs, to the "Sticky dynamic IP address" group. We believe that IPs that belong to this group are using DHCP. In the DHCP protocol [29], the ISP assigns its client an IP address and a lease time which determines the amount of time that subscriber can use this IP address. [9] In this period of time the client can also disconnect from the network and return to the same IP (hence the terminology "sticky dynamic IP address"). The lease time is usually a couple of days, since the ISP wishes to avoid load on the DHCP server and redundant traffic. We find support for our speculation in the fact that the Dynamic IPs that belong to the MSLT group had an average down time between sessions of 9 hours while static IPs that belong to MSLT had an average down time of 54 hours. We suspect that the dynamic IPs in the MSLT group had a relatively lower down time between sessions, since "Sticky dynamic IP address" can return to their original IP address only if the break is short and the lease time has not elapsed.

Another support for the correlation between the MSLT/SSLT groups to static/dynamic groups can be seen in the great similarity between the CDF graph of residual life time of the static/dynamic groups (see Figure 5(b)) to the CDF graph of residual life time of MSLT/SSLT groups (see Figure 3(a)).

The fact that MSLT/SSLT groups correlated to static/dynamic IP groups can explain the different residual life time and the different accessibility between the two groups. Dynamic IPs replace their IPs and hence live a shorter life time. However, one may wonder why there is also a difference in the first session length (see Figure 4(a)) which influences the Churn. Our speculation is that the root cause is the special type of clients that maintain static IPs. Static IPs are widely used for server applications or academic networks. Servers or academic networks need constant IPs since they need to be constantly up and available. A consequence of the required high availability is the relatively good infrastructure and hence the long session. We also observe, looking at the rDNS results, that there are a high number of university computers in the static group.

7 Conclusion and Discussion

Choosing which clients should become Super-Nodes in P2P networks is an important and crucial task for the stability of the network. As far as we know we

[9] During the lease time the subscriber can also renew the address lease time

are the first paper that shows the impact of choosing static IPs on the different aspects of network stability. The high stability of static IP is due to two reasons: the fact that the static IP does not change the IP address, and the fact that computer which is connected through a static IP connection is relatively more stable, since the computer is usually used as a server. The fact that static IPs are more stable can be used to choose stable SNs, and thus increase the stability of the P2P network. While it is a hard task to classify the type of address (static or dynamic) using the IP address alone, this is an easy task for the P2P application.

Specifically, the P2P application is usually designed in such a way that the application in the clients send from time to time keepalive messages to the central unit of the P2P network with some identifer of the client application. Using those keepalive messages the P2P network can detect the changes in the IP addresses of the computer. Change of IP can be due to the roaming of the computer (incase of laptop or smart phone device) or due to the fact that the IP address of the computer is a dynamic IP address. A thumb rule suggests that if the change is to another IP in the same subnetwork (usually the /24 or according to the BGP prefixes [30]) and the change is periodically then the change is due to the fact that this is dynamic IP [28]. Hence our paper observation on the rule of type of address (dynamic/static) is an operative guideline to the designers of P2P system.

Our measurements on the life cycle of SNs reveal that there is a set of nodes which is very stable; that its session duration is a couple of days and its life span is over 3 months (note that 38% of the nodes in the MSLT group live during the entire experiment which was 3 months). Those nodes can be used as an important building block in designing a more stable P2P network.

Acknowledgment

The authors would like to thank Amir Lev, Commtouch's CTO, Prof. Hanoch Levy from Tel-Aviv University and Omer Dekel, a graduate student at IDC, for their helpful suggestions.

References

1. L. Garces-Erice, E.W.Biersack, P. Felber, K. Ross, , and G. Urvoy-Keller, "Hierarchical peer-to-peer systems," in *ACM/IFIP International Conference on Parallel and Distributed Computing (Euro-Par)*, 2003.
2. F. Wang, J. Liu, and Y. Xiong, "Stable peers: Existence, importance, and application in peer-to-peer live video streaming," in *IEEE Infocom*, 2008.
3. S. Saroiu and S. Gummadi, P. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Multimedia Computing and Networking*, 2002.
4. S. Sen and J. Wang, "Analyzing peer-to-peer traffic across large networks," in *ACM SIGCOMM Internet Measurement Workshop*, 2002.
5. R. Bhagwan, S. Savage, , and G. Voelker., "Understanding availability," in *IPTPS*, 2003.

6. K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, , and J. Zahorjan, “Measurement, modeling, and analysis of a peer-to-peer file-sharing workload,” in *ACM SOSP*, 2003.
7. P. Godfrey, S. Shenker, and I. Stoica, “Minimizing churn in distributed systems,” in *ACM SIGCOMM*, 2006.
8. D. Stutzbach and R. Rejaie, “Characterizing churn in peer-to-peer networks,” Tech. Rep., May 2005.
9. S. Guha, N. Daswani, and R. Jain, “An experimental study of the skype peer-to-peer voip system,” in *IPTPS*, 2006.
10. S. A. Baset and H. G. Schulzrinne, “An analysis of the skype peer-to-peer internet telephony protocol,” in *IEEE INFOCOM*, 2006.
11. “kazaa. [Online] 2007,” http://kazaa.com.
12. “Joost ,” http://joost.com.
13. “ iMesh ,” http://www.imesh.com/ .
14. “ Morpheus discussion group ,” http://www.gnutellaforums.com/morpheus-windows/ .
15. K. F. Suh, D. R. Kurose, and D. J. Towsley, “Characterizing and detecting skype-relayed traffic,” in *IEEE INFOCOM*, 2006.
16. S. Ehlert, T. Magedanz, S. Petgang, and D. Sisalem, “Analysis and signature skype voip session traffic,” Tech. Rep. NGNI-SKYPE-06B, 2006.
17. “D. Fabrice. Skype uncovered, 2005.” http://www.ossir.org/windows/supports/-2005/2005-11-07/EADS-CCR_Fabrice_Skype.pdf.
18. “Recon 2006 - Fabrice Desclaux and Kostya Kortchinsky - Vanilla Skype (2006),” http://recon.cx/en/f/vskype-part1.pdf , http://recon.cx/en/f/vskype-part2.pdf.
19. P. Biondi and F. Desclaux, “Silver needle in the skype,” in *Blackhat*, 2006.
20. L. D. Cicco, S. Mascolo, and V. Palmisano, “An experimental investigation of the congestion control used by skype voip,” in *5th International Conference on Wired/Wireless Internet Communications*, 2007.
21. “Apoc Matrix , The SkypeLogger, December 2006,” http://www.epokh.org/drupy/files/Skype%20Logger.pdf.
22. K.-T. Chen, C.-Y. Huang, P. Huang, and C.-L. Lei., “Quantifying skype user satisfaction,” in *ACM SIGCOMM*, 2006.
23. T. Hofeld, A. Binzenhofer, M. Fiedler, and K. Tutschku, “Measurement and analysis of skype voip traffic in 3g umts systems,” in *The 4th InternationalWorkshop on Internet Performance, Simulation, Monitoring and Measurement*, 2006.
24. X. Wang, X. Chen, and S. Jajodia, “Tracking anonymous peer-to-peer voip calls on the internet,” in *ACM Conference on Computer and Communications Security*, 2005.
25. A. Bremler-Barr, O. Dekel, and H. Levy, “Harvesting skypesuper-nodes,” Tech. Rep., May 2008.
26. T. Holz, M. Steiner, F., D. E., Biersack, and F. Freiling, “Measurements and mitigation of peer to-peer-based botnets: A case study on storm worm,” in *LEET 08: The first USENIX Workshop on Large-Scale Exploits and Emergent Threats*.
27. L. Kleinrock, *Queueing Systems: Volume I Theory (Wiley Interscience, New York, 1975).*
28. Y. Xie, F. Yu, K. Achan, E. Gillum, M. Goldszmidt, and T. Wobber, “How dynamic are ip addresses,” in *ACM SIGCOMM*, 2007.
29. M. Khadilkar, N. Feamster, M. Sanders, , and R. Clark, “Usage-based dhcp lease time optimization,” in *Internet Measurement Conference*, 2007.
30. B. Krishnamurthy, “On network-aware clustering of web clients,” in *In Proceedings of ACM SIGCOMM*, 2000, pp. 97–110.