Reliability and Relay Selection in Peer-to-Peer Communication Systems

Salman Abdul Baset and Henning Schulzrinne
Department of Computer Science
Columbia University, New York, NY, USA
{salman,hgs}@cs.columbia.edu

ABSTRACT

The presence of restrictive NATs and firewalls prevent nodes from directly exchanging packets and thereby pose a problem for peer-to-peer (p2p) communication systems. Skype, a popular p2p VoIP application, addresses this problem by using another Skype client (relay) with unrestricted connectivity to relay the signaling and media traffic between session endpoints. This distributed technique for addressing connectivity issues raises challenging questions about the reliability and latency of relayed calls, relay selection techniques, and the interference of relayed calls with the applications running on relays – a phenomena we refer to as user-annoyance.

We discuss a framework to analyze the reliability in peer-to-peer communication systems, present a simple model to estimate the number of relays needed to maintain a desired reliability for the media sessions, and analyze two techniques to improve reliability of relayed calls. We present a distributed relay selection technique that leverages a two level hierarchical p2p network to find a relay in $O(1)$ hop. We augment our distributed relay selection technique to find a relay that minimizes call latency and user-annoyance. Our results indicate that for Skype node lifetimes, at least three relays are needed to achieve a 99.9% success rate for call duration of 60 mins (95^{th} percentile of Skype call durations).

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: [Reliability, availability, and serviceability]

General Terms

Design, Reliability, Measurement

Keywords

Reliability, P2P, VoIP, Relay

1. INTRODUCTION

Restrictive network address translators (NATs) and firewalls prevent hosts from directly exchanging packets. A recent survey of 1,630 NAT devices indicates that hosts behind ~25% of these devices cannot traverse the NATs using UDP or TCP implying that hosts behind two different such devices are not likely to directly exchange packets without an intermediary. Moreover, corporations are increasingly deploying firewalls to protect their networks from malicious traffic that originates both outside and inside their networks. The restrictive NAT and firewalls pose a problem for IP communication systems because they prevent the user agents from directly exchanging signaling and media traffic.

In a client-server (c/s) communication system, the caller user agent discovers the current network address of a callee user agent through a managed server and exchanges signaling information with the callee user agent to establish a media session. The media traffic flows directly between the user agents. To address the connectivity constraints due to restrictive NATs and firewalls, c/s systems such as Vonage [6] use managed servers for relaying media traffic between user agents with restrictive connectivity. In contrast, in a peer-to-peer (p2p) communication system, user agents collaborate to discover the network address of the callee and use peers with unrestricted connectivity and sufficient bandwidth to let the user agents with restrictive connectivity exchange signaling and media traffic [7]. Suh *et al.* [26] report hundreds of calls being relayed by a Skype relay.

The above characteristics of a p2p communication system pose unique challenges for a system designer. First, the lookup performance in p2p systems must be as effective as the lookup performance of client-server systems. Additionally, a media session may be prematurely terminated because a relay peer goes offline. This issue necessitates a formal analysis of the reliability of p2p communication systems and techniques to prevent dropped sessions. Moreover, as media sessions such as voice and video have a tight playout requirement, the network latency of a media session involving a relay peer should satisfy these tight requirements. Further, the relaying of a media session may interfere with other user applications and and impair their performance. A system designer must either provide incentives for users to run relay peers or design techniques that minimize the interference of relayed session with other user applications.

In this paper, we present a framework to analyze the reliability of peer-to-peer communication systems (Section 3). We then devise a simple analytical model that predicts the smallest number of relays needed to achieve the desired reliability for relayed media sessions (Section 4.1) and evaluate it on exponential, pareto, and Skype node lifetime. For a given node lifetime and call duration distribution, the model allows to determine the minimum number of relays so that the percentage of successful calls does not fall below a desired threshold (e.g., 99.9%). Such an analysis can help characterize the resources (relays) needed for improving the reliability of relayed calls. We then devise two techniques to prevent dropped sessions, selecting k relay peers at the beginning of a call with no-replacement and with-replacement and pre-

dict their reliability improvement using reliability theory in Section 4.2 and 4.3. In Section 4.4, we analyze the reliability improvement scheme used by Skype. Section 4.5 presents the experimental evaluation of the model and discussion.

In Section 5, we present a distributed technique to find a relay peer in $O(1)$ hop and compare the performance of this technique to a relay selection scheme that has global knowledge of all the relays in the p2p network. Instead of designing incentives for users to allow relaying of media sessions through their user agents, we aim to minimize the interference of relayed session with the user applications. To capture the impact of relayed media sessions on user applications, we introduce the notion of *user-annoyance* (Section 5.2). We augment our distributed search technique to select a relay that minimizes delay, user-annoyance, or both within a threshold. To the best of our knowledge, we are the first to address these issues in peer-to-peer communication systems. Our analysis and results are also applicable to media translation and conferencing in p2p communication systems.

2. PROBLEM SETTING

We consider a peer-to-peer communication system that has N participating nodes. A node is a machine with CPU, memory and disk and is connected to the Internet through a dialup, DSL, cable, fiber, or a wireless connection. Typically, a human user is associated with each node or a machine and runs a peer-to-peer communication application(s) and other applications. The p2p applications use any peer-to-peer protocol to form a p2p network. There are two types of nodes in a peer-to-peer communication network, peers and free-riders. In the literature, they are also referred to as super nodes and ordinary nodes [7, 19] or peers and clients [10]. A peer fully participates in the p2p network, collaborates with other peers to discover the current network address of the callee user agent, and can relay one or more media sessions. A free-rider does not collaborate in the discovery of the callee user agent and does not relay any media sessions. However, this collaboration is not always purposefully avoided. The presence of restrictive NATs and firewalls may hinder the participation of a node in the overlay, thereby forcing it to act as a free-rider. The need for relaying media sessions between caller and callee user agents arises precisely due to this reason. For ease of exposition, we refer to the caller and callee user agent as caller and callee, relay peer as relay, and voice session as a call. Unless stated otherwise, we refer to the p2p communication application as a p2p application.

3. RELIABILITY OF A P2P COMMUNICATION SYSTEM

Availability is the classical metric for modeling the reliability of a communication system and is typically measured by the number of nines after a decimal point. For example, a "3 nines" (99.9%) reliability means that the system is down only 0.1% of the time. For relayed calls, availability implies the ability of a p2p communication system to find a relay for establishing the call. However, this notion does not fully capture the reliability of relayed calls because in addition to relay search failure, calls can also fail due to relay churn as there is no guarantee about their uptime. Thus, a more accurate metric to determine the reliability of relayed calls is the number of dropped calls due to relay failure and inability to find a relay.

$$P_{succ} = P_{ss}F_{norelay} + P_{ss}\overline{F}_{norelay}P_{rs}P(R > D) \quad (1)$$

Equation (1) formalizes the notion of reliability or percentage of successful calls in a p2p communication system. The term to the immediate left of plus sign is the probability of successfully finding the network address of the callee user agent, P_{ss}, times the proportion of calls that do not need a relay, $F_{norelay}$. The term to the immediate right of plus sign is the probability of successfully finding the relay times the proportion of calls that need a relay, $\overline{F}_{norelay}$, times the probability that the residual lifetime of a relay, R, is greater than the call duration distribution D. This equation indicates that the proportion of successful calls can be increased by enhancing the performance of lookup schemes [21], by designing schemes that establish a media session between user agents in the presence of NAT and firewalls without requiring a relay [13], and by improving the success rate of distributed relay search and relay calls. We focus our attention on analyzing the reliability of relayed calls and relay search since other areas have seen related work [22].

4. MODELING RELIABILITY OF RELAYED CALLS

We present a simple model to calculate the minimum number of relays per call, k so that the success rate of relayed calls is above a desired reliability criteria such as 99.9% (Section 4.1), analyze two reliability improvement schemes, namely, *no-replacement* (Section 4.2) and *with-replacement* (Section 4.3), and present an evaluation of the model and reliability improvement schemes (Section 4.5). Our analysis assume that the nodes that need a relay to establish a call (ordinary nodes) can randomly select it from the set of all relays, that relays are plenty, and the system has reached stationarity. In Section 5.1, we discuss a distributed scheme to find a relay.

4.1 Number of Relays

Let X_i be a random variable (r.v) that denotes the lifetime of relay i, F_{X_i} be its CDF, and X_i be independent and identically distributed (i.i.d). Let R_i be a random variable that denotes the residual lifetime of relay i when it starts relaying the call and D denote the distribution of call duration. When a relay fails, the call it is relaying is immediately switched to a new relay j, having residual lifetime R_j. Since the new relay is selected immediately when the old relay fails, the residual lifetime of the relays used are also i.i.d. For simplicity, we assume that calls are not dropped during switch over to a new relay. Leonard *et al.* [18] note that if the system has reached stationarity, the CDF of residual lifetimes is given as:

$$F_R(x) = P(R_i < x) = \frac{1}{E[X_i]}\int_0^x (1 - F(z))dz \quad (2)$$

We are interested in determining the minimum relays per call k, so that the number of dropped calls due to relay failure is below a desired criteria, i.e.,

$$\text{Desired reliability} \leq P(\sum_{i=1}^{k} R_i > D) \quad (3)$$

LEMMA 1. *When X and D are exponentially distributed with parameters λ and ν, the r.h.s of (3) has a closed form solution:*

$$P(\sum_{i=1}^{k} R_i > D) = 1 - (\frac{\lambda}{\lambda+\nu})^k \quad (4)$$

Proof:
For exponential distribution, (2) can be solved to obtain $F_R(x)$ and its probability distribution function (pdf) $f_R(x)$, which are $1 - e^{-\lambda x}$ and $\lambda e^{-\lambda x}$, respectively. Using conditioning:

$$P(D < \sum_{i=1}^{k} R_i) = \int_0^\infty F(D < m) \times f(\sum_{i=1}^{k} R_i = m)dm$$

$f(\sum_{i=1}^{k} R_i = m)$ is a k-fold convolution of exponential r.v's which have a gamma pdf.

$$= \int_0^\infty (1 - e^{\nu m}) \times \frac{\lambda e^{-\lambda m}(\lambda m)^{k-1}}{(k-1)!} dm$$

$$= \int_0^\infty \frac{\lambda e^{-\lambda m}(\lambda m)^{k-1}}{(k-1)!} - \frac{\lambda e^{-(\lambda+\nu)m}(\lambda m)^{k-1}}{(k-1)!} dm \quad (5)$$

The left term of (5) is 1 since it is an integral of gamma pdf. Multiple and divide the right term by $(\lambda+\nu)^k$ and using $\Gamma(n) = \int_0^\infty e^{-x}x^{k-1}dx = (k-1)!$

$$= 1 - (\frac{\lambda}{\lambda+\nu})^k \quad (6)$$

For arbitrary lifetime and call distribution, the r.h.s of (3) is difficult to solve as convolution of k i.i.d random variables is non-trivial. Instead, we use the following approximation which replaces the sum of k r.v's with their maximum.

LEMMA 2. *The sum of* k *i.i.d r.v's R_i being greater than another r.v D is greater than or equal to one minus the k^{th} exponentiation of the probability of R being less than D.*

$$P(\sum_{i=1}^{k} R_i > D) \geq 1 - P(R < D)^k \quad (R_i \text{ are i.i.d}) \quad (7)$$

Proof:

$$P(\sum_{i=1}^{k} R_i < D) \leq P(\max(R_1, \ldots, R_k) < D)$$

$$P(\max R_i < D) = P(R_1 < D, \ldots, R_k < D)$$

$$= P(R < D)^k \quad \text{since } R_i \text{ are i.i.d}$$

$$P(\sum_{i=1}^{k} R_i > D) \geq 1 - P(R < D)^k$$

Observe that if node lifetimes are exponentially distributed, the equality holds in (7) holds and (4) is obtained. For non-exponential node lifetimes, the k^{th} exponentiation decreases much faster than the sum and intuitively, the bound is loose for large values of k. However, the relative error of the bound depends on the lifetime and call duration distributions. Next, we examine the relative error of (7) for pareto distribution since the measurement studies of Skype node lifetimes suggest using heavy tailed distributions as an approximation [15] and pareto is the most natural choice for such an approximation.

4.1.1 Pareto node lifetimes

The CDF of pareto lifetimes is $F(x) = 1 - (\frac{x}{b})^{-a}$, where a is the shape parameter and b is the scale parameter. For our analysis, we use the shifted pareto distribution $F(x) = 1-(1+\frac{x}{b})^{-a}$ with mean $\frac{b}{a-1}$ [18], because without the shift, a node is guaranteed to be up for b units of time. Clearly, the mean of this distribution is only defined for $a > 1$ where as variance is only defined for $a > 2$ which prevents the calculation of an an exact analytical formula for sum of k pareto i.i.d r.v's. Zaliapin *et al.* [29] describe methods for approximating the upper quantile (0.98), lower quantile (0.02), and median of sum of k i.i.d r.v's. Their results indicate that although replacing the sum with the maximum can reasonably approximate the quantiles around median, such an approximation is poor for the lower and upper quantiles and for large values of k (e.g., > 10). The CDF of residuals of pareto lifetimes is $F(x) = 1 - (1 + \frac{x}{b})^{1-a}$ [18]. Although, the approximation results by Zaliapin can be extended to the sum of pareto residuals for arbitrary values of a, b, and k, such an effort is beyond the scope of this paper. Further, the utility of precise approximation may be limited due to the difficulty in estimating the pareto parameters. Also, real node lifetimes do not follow a strict pareto distribution and incorporate effects such as diurnal variations [15,17]. Therefore, to obtain a bound on the minimum number of relays to achieve desired reliability, we approximate the sum of k pareto residuals with their maximum, but note that in doing so, it is necessary to get an estimate of the relative error of such an approximation to determine its usefulness.

In Table 1, we show the simulated values of the sum of two and four pareto residual R_i being less than exponentially distributed call holding times D and the relative error of the approximation (maximum of R_i being less than D) with respect to the simulated values. The simulated results are an average over 10^7 runs. The parameters a and b were chosen so that the mean of the distribution was five and one hour, respectively. The choice of mean uptime of five hours approximately reflects the median of the observed Skype node lifetimes [15, 17], where as mean node lifetime of one hour is for a relatively less stable system. The top two values in the fourth column are zero because sum of four R_i was never observed to be smaller than D (with mean of 2.5 and 5 minutes) in 10^7 runs. Observe that the relative error is low ($< 0.2\%$) when the value of the simulated sum of R_i r.v's being less than D is above 2% where as the relative error increases for simulated values smaller than 2%. This result is consistent with [29] which notes that using the maximum of k pareto r.v's instead of their sum is not a good approximation for lower quantiles (< 0.02). However, note that although the relative error increases as D decreases and the number of summands k increase, we are only interested in the smallest value of k for which the call success rate is just above 99.9% and not an arbitrary large value of k. In general, the approximation can be applied to determine the smallest value of k that meets the desired reliability criteria, as long as the relative error remains low (e.g., $< 1\%$).

Next, we present two schemes for preventing failure of relayed media sessions due to relay churn.

	a=2,b=5 (mean lifetime=5 hours)				a=3,b=2 (mean lifetime=1 hour)			
	k=2		k=4		k=2		k=4	
call duration	sim (%)	rel-e (%)	sim (%)	rel-e (%)	sim (%)	rel-e (%)	sim (%)	rel-e (%)
2.5	0.0074	8.5755	0	0.00	0.1544	0.2171	0.0003	21.3205
5	0.0251	3.6121	0	0.00	0.5517	0.2131	0.0027	6.9055
10	0.0961	1.7193	8e-5	20.0925	1.8179	0.1980	0.0319	3.8110
20	0.3553	1.3791	0.0011	14.7570	5.2869	0.1456	0.2772	0.2958
30	0.7171	0.4476	0.0053	1.9231	9.0853	0.0737	0.8292	0.2894
40	1.1567	0.4465	0.0137	1.7594	12.867	0.0233	1.6608	0.2589
50	1.6537	0.4349	0.0265	1.1979	16.464	0.0061	2.7106	0.0885
60	2.1895	0.1096	0.0482	0.8299	19.836	0.0303	3.9368	0.0585

Table 1: Simulated values of $P(\sum_{i=1}^{k=2} R_i < D)$ and $P(\sum_{i=1}^{k=4} R_i < D)$ for pareto lifetimes are shown in the 'sim' column. The values indicate the percentage of dropped relay calls in 10^7 runs. The relative error of the approximation $P(R < D)^{k=2}$ and $P(R < D)^{k=4}$ with respect to the simulated values is shown in the 'rel-e' column. Call duration is exponentially distributed.

4.2 No-replacement Scheme

In the no-replacement scheme, k relays are selected at the beginning of the call with one relay acting as primary and $k-1$ acting as backup. If the primary relay fails, the call is switched to a backup relay. We assume that calls are not dropped during switch over. A call fails when all k relays fail. Let R_i be a random variable that denotes the residual lifetime of the relay i when it is drafted as a relay and D be a random variable that denotes call duration. We are interested in the probability that at least one of the relay, that were selected when call was established, is online before the call completes:

$$P(\max(R_1, \ldots, R_k) > D)$$
$$= 1 - \int_0^\infty P(R < z)^k P(D = z) dz \quad (R_i \text{ are i.i.d}) \quad (8)$$

We solved (8) to determine the proportion of successful relay calls using two or three relays when node lifetimes are exponentially distributed, and the corresponding expressions are $1 - \frac{2\nu}{\lambda+\nu} + \frac{\nu}{2\lambda+\nu}$ and $(\frac{1}{2\lambda+\nu} - \frac{1}{3\lambda+\nu})\frac{6\lambda^2}{\lambda+\nu}$, respectively. For pareto node lifetimes, we numerically solved (8) to obtain the proportion of successful calls using two or three relays.

How many relays? As might be expected, the proportional increase in the reliability decreases with selecting more relays at the start of the call. For example, when node lifetimes are exponentially distributed, the MTTF of a 2-relay, 3-relay, and 4-relay schemes are $\frac{3}{2\lambda}$, $\frac{11}{6\lambda}$, and $\frac{25}{12\lambda}$, respectively. The proportional increase in MTTF is 50%, 22%, and 13%, respectively. Clearly, this is a case of diminishing returns. Further, maintaining numerous backup relays exclusively for every call when relays are not plenty is likely to result in a poor performance from the perspective of successful call establishment for relayed calls, reliability, delay, and user annoyance.

4.3 With-replacement scheme

This scheme is similar to the no-replacement scheme in that k relays are selected at the beginning of a call, and a call is switched to a backup relay if the primary relay fails. However, when a caller or callee detects that one of the k relays has failed, it launches a search to replace the failed relay. Suppose it takes μ time units to detect that a relay has failed and find a new relay. If node lifetime and search time

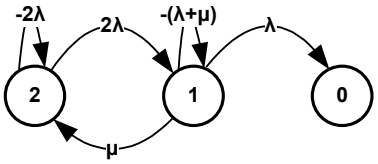

Figure 1: Markov chain for a 2-relay with-replacement scheme.

are exponentially distributed, a Markov chain can be used to evaluate the reliability of this scheme [9]. For a single backup relay, the Markov chain is shown in Figure 1. In the reliability literature, this scheme is referred to as 1-out-of-2 active redundancy with constant failure rate λ and constant repair rate μ [9]. This chain can be solved to obtain MTTF, i.e., the time it spends in states (2) and (1), when two and one relays are operational. The failure rate is the reciprocal of MTTF, i.e.,

$$\frac{1}{\lambda_{WR}} = MTTF = \frac{3\lambda + \mu}{2\lambda^2} \quad (9)$$

The subscript WR denotes with-replacement. For $\lambda << \mu$, this scheme approximately behaves like a one relay scheme with constant failure rate λ_{WR} (Birolini [9, page 190]). Let R_{WR} be a random variable that denotes the reliability of this scheme. Since its failure rate is constant, its CDF is $R_{WR}(t) = e^{-\lambda_{WR}(t)}$. When call duration is exponentially distributed with parameter ν, probability that a call completes before the two relays fail and a search for the replacement relay also fails is:

$$P(R_{WR} > D) = \frac{\nu}{\nu + \lambda_{WR}} \quad (10)$$

When the node failure rates are not constant, either non-homogeneous poisson processes may be used to model the reliability of this scheme or node lifetime can be split into periods where failure rate is constant. However, the difficulty in using such analysis lies in the fact that for heavy tailed distributions, the shape parameter a is often not accurately known. Therefore, we leave such analysis for future work.

4.4 Reliability of Relayed Calls in Skype

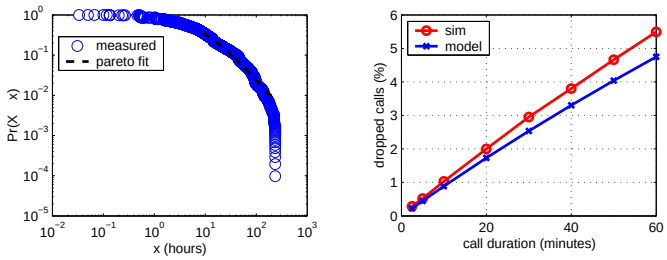

Figure 2: (Left) CCDF of the node lifetimes and the pareto fit for Skype data set (right) percentage of dropped calls through simulations on the Skype data set and using a pareto model when only one relay is used.

We performed experiments to determine if the Skype application employs a no-replacement or a with-replacement scheme. We blocked direct traffic between two machines running in our lab using NetPeeker [3] and then ran Skype applications on them and established a call. Since the traffic was blocked between the machines, the Skype applications were forced to use a relay to exchange signaling and media traffic. Using NetPeeker [3], we blocked the media traffic between caller machine and the relay, which is similar to emulating a relay failure. Within 2-4 seconds, the Skype applications chose a new media relay. We then immediately blocked traffic between this new relay and the caller Skype application which resulted into the call getting disconnected. The experiment shows that when a call is established that requires a relay, the Skype application chooses a backup relay at the start of the call. When both relays fail simultaneously, the call is disconnected.

To determine if a Skype application searches for a new relay when the primary relay fails and the call is shifted to the backup relay, we gradually increased the time between primary and backup relay failure from 30 s to two minutes. Our experiments indicate, that a Skype application waits for more than a minute before searching for a new relay. Thus, it employs a 'periodic-recovery' scheme for replacing a failed relay instead of a 'reactive-recovery' scheme. We periodically failed the primary relay every 90 s for a call lasting 15 minutes and found that the Skype application was able to find a backup relay and the call did not get disconnected.

All the experiments were performed during the first week of December 2009 and more than 70 calls were established over a period of seven days.

4.5 Evaluation and Discussion

We evaluate the analytical model for the number of relays, and reliability improving techniques using simulations. We wrote an event driven simulator in which nodes can form an overlay network using Chord. We use a relay selector which randomly selects a relay from the pool of online relays having sufficient network capacity. The inter-arrival time between requests for relayed calls is exponentially distributed and its mean is adjusted over the course of the simulation so that the cumulative load of relayed calls follow a target utilization (such as 40%). Thus, in our simulations, the relayed calls only fail due to relay failure. We run the simulation for 10 days of simulated time and repeat the experiments until 10^7 call attempts have been made. The warm up period is excluded from the reported results.

We use three node lifetime data sets. The first two data sets contain synthetically generated exponential and pareto node uptime and downtime with a mean of 300 minutes. The pareto parameters a and b were chosen as 2 and 5, respectively. The third data set contains the uptime and downtime of 4,000 Skype nodes measured for 25 days [15]. The uptime of Skype nodes was measured by sending a specially crafted Skype message to these nodes every 30 minutes. We randomly selected 1,740 nodes from this data set of 4,000 nodes because this is the maximum number of end nodes for which all pair ping latency data is available [16]. We use this data for designing a distributed relay search mechanism that minimizes latency of the relayed calls in Section 5.3.

All 1,740 nodes can potentially provide the relay service. The median and mean uptime of these 1,740 nodes was 256 and 711 minutes, respectively. The pareto parameters, a and b, computed using the method of maximum likelihood and Kolmogorov-Smirnov statistic, are 1.4916 and 8.9833, respectively. Figure 2 (left) shows the CCDF of Skype node lifetimes and the pareto fit indicated by a dashed straight line. Towards the end of the tail, the measured lifetimes exhibit a knee of the curve. This happens because the node lifetimes do not strictly exhibit a pareto behavior and the measurement is stopped after T time units. For the Skype data set, Figure 2 (right) shows the percentage of dropped calls when a call is assigned to one relay through simulations and those predicted by the model $P(R < D)$. The relative error with respect to simulations was less than 15%. Wang [28] suggested that there is an inherent inaccuracy in computing the exact parameters of the node lifetime distribution when they are sampled every T time units. We note that such a bias depends on the ratio of the mean node lifetime and the sampling interval: the higher the ratio, the lesser the inaccuracy and vice versa. Nevertheless, we note that when the real lifetime data is used for churn simulations, such a bias will always be present.

A key consideration is to realistically set the upload and download bandwidth of a relay peer since it cannot relay an arbitrary number of calls. Dischinger *et al.* [12] have measured the upload and download bandwidth for a range of broadband hosts and we set the relay bandwidths according to their reported distribution. We assume that a relay call needs an uplink and downlink bandwidth of 128 kb/s (using the G.711 codec). Modern codecs such as SILK [5] which has a bit-rate between 4-40 kb/s can bring down the required bandwidth at a relay to 8-80.

Figure 3 shows the number of relays for exponential, pareto, and Skype node lifetimes for a range of exponentially distributed call holding times. Guha [15] showed that 95% of Skype relayed calls last less than an hour. The approximation from (7) is used to calculate number of relays when pareto distribution is used to model node lifetimes. For pareto node lifetimes (second row in the figure) and call duration of 60 minutes, the relative error of the approximation was less than 1%. The results from the simulation show that for the Skype data set and for call durations of 60 minutes or less, three relays are sufficient to achieve a call success rate of 99.9%. Observe that modeling the Skype node lifetimes as exponential and pareto resulted in a minimum relay prediction of three relays which matches the simulations. For call

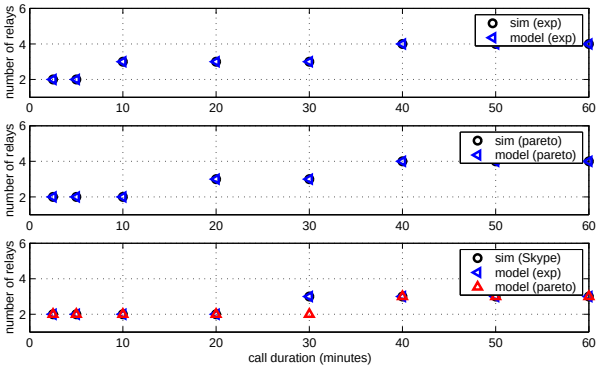

Figure 3: Number of relays for exponential (top) pareto (middle) and Skype (bottom) node lifetime data set to maintain call success rate of 99.9%. The mean node lifetime for exponential and pareto distributions was 300 minutes.

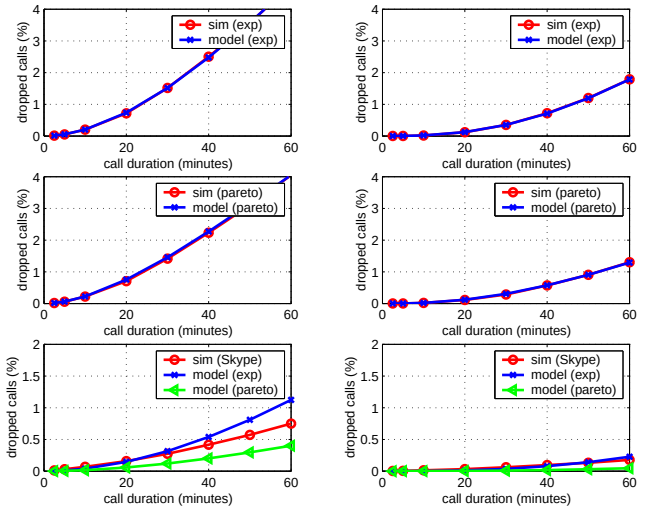

Figure 4: Proportional of failed calls using simulations and model for exponential (top), pareto (middle), and Skype (bottom) node lifetimes. The figures on the left and right are for a 2-relay and 3-relay no-replacement scheme, respectively.

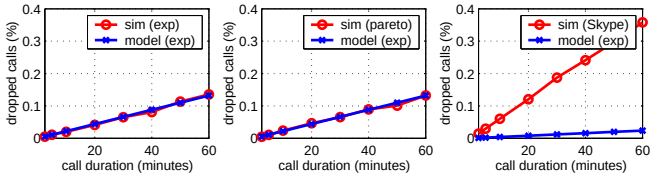

Figure 5: Proportion of failed calls using simulations and Markov model for 2-relay with-replacement scheme for exponential (left), pareto (middle), and Skype (right) node lifetimes.

duration of 30 minutes, the pareto model under predicts the number of relays. However, this is expected as Skype node lifetimes do not exactly follow the pareto model (Figure 2). Also, for the results shown, note that although only three or four relays or less are needed to achieve call drop rate of 0.1% or less for call duration of 60 minutes, the number can be higher when node lifetimes are smaller. As an example, when the node lifetimes are exponential with a mean of one hour, at least ten relays per call are needed to achieve a success rate of 99.9% for mean call duration of 60 minutes.

Figure 4 shows the reliability of a 2-relay and 3-relay no-replacement scheme for exponential, pareto, and Skype node lifetime data sets computed using (8). As expected, there is a good match between analytically computed (using (8)) and simulated call success rates for exponential and pareto node lifetimes. For the Skype data set, the simulations show that a 2-relay scheme achieves a 99.9% success rate for call durations of 10 minutes or less where as for call duration of 60 minutes, the success rate is 99.25%. For 2-relay no-replacement scheme, using exponential and pareto node lifetimes to model Skype node lifetimes results in over predicting and under predicting the number of dropped calls by approximately a factor of two, respectively.

Figure 5 shows the reliability of a 2-relay with-replacement scheme for exponential, pareto, and Skype node lifetime data sets. The time to detect if a relay has failed and consequently to search a new relay is exponentially distributed with a mean of 60 s. As expected, the Markov model accurately predicts the call drop rate when node lifetimes are exponential. The results also indicate that the Markov model may be a reasonable approximation for pareto node lifetimes. For Skype data set and for call duration of 60 minutes, this scheme achieves a call success rate of 99.65%, an improvement of 0.3% over a 2-relay no-replacement scheme. The improvement is small because node lifetimes have a large mean (711 minutes). When node lifetimes have a small mean, it may be necessary to incorporate a with-replacement scheme to avoid dropped calls. Since Skype employs a 2-relay with replacement scheme with a relay search time of approximately 60 s, the results from our simulations indicate that the drop rate of relayed calls is likely to be small. However, Skype's relay mechanism is not completely random and is biased towards low latency and high bandwidth relays. Such a bias may result in higher drop rates for relayed calls [14]. Nevertheless, an implication of the results is that for Skype node lifetimes, simple schemes for reliability improvement such as two relay no-replacement and with-replacement give reasonable reliability performance thereby obviating the need for a sophisticated reliability improvement scheme.

4.5.1 *Practical implications of these schemes*

In a k-relay no-replacement scheme, both caller and callee exchange information about k relays at the time of call establishment. After a call has been established, they must periodically check the liveness of all k relays. The liveness period should be adjusted so that when the primary relay fails, there is a high likelihood that the new relay to be incorporated is alive. However, the reliability returns of maintaining a large number of backup relays at the start of the call are diminishing, especially under high churn. For this reason, a with-replacement scheme is attractive. Such a scheme can potentially start with 2 or 3 relays, and find a re-

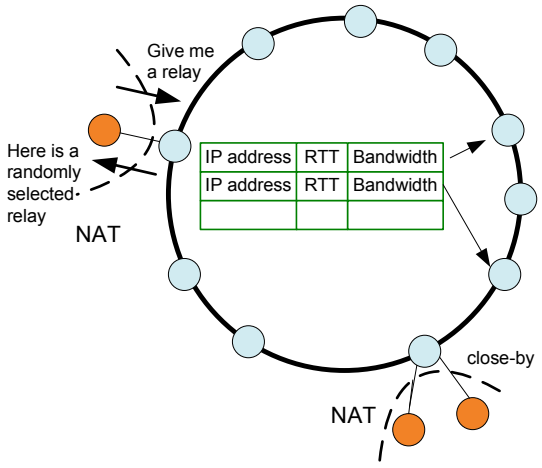

Figure 6: Two-tiered overlay implementing a *local-random* scheme for relay selection.

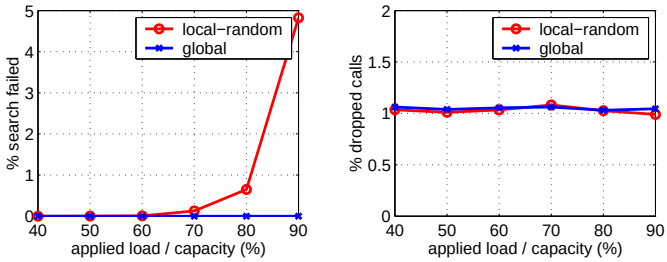

Figure 7: Performance of local-random scheme vs. global scheme as a function of system load (left graph). Percentage of dropped calls when one relays fails (right graph).

placement for a failed relay. However, the caller and callee must exchange information about the new relay in subsequent signaling messages. For a no-replacement scheme, no such exchange is required.

4.5.2 *Other reasons for call failure*

Relay failure is not the only reason why relayed calls may fail. Such calls can also fail during call switching. Also, since nodes may use silence suppression, it may take more time to correctly distinguish between silence periods and a failed relay because the frequency of heart-beat messages is likely to be lower than real-time voice or video packets. If a search for a relay is launched at the time when all relays fail, the caller and callee can perceive a silence gap in the conversation. If the duration of the perceived gap is long, the call participants may simply terminate the call.

5. RELAY SELECTION

In this section, we devise distributed techniques to find a relay that address several practical issues. The first issue is that the distributed relay search must find a relay in a timely manner to minimize the call establishment time and to quickly recover from relay churn. Also, the relaying of media session can interfere with the user applications and impair their performance. It is important to select relays in a way that minimize this interference. Besides minimizing interference, latency and increasing reliability are key objectives for relayed calls. Addressing all these factors is a multi-objective optimization problem which is NP-hard.

In Section 5.1, we devise a distributed relay selection technique that can find a relay in $O(1)$ hops and compare its performance to a scheme that randomly selects a relay from the global pool of all relays. Section 5.2 introduces the notion of user annoyance. In Section 5.3, we augment the distributed relay selection scheme to devise heuristics for finding a relay that, for a relayed call, minimizes user annoyance or latency or both, and evaluate their performance.

5.1 Distributed Relay Selection

We devise a relay selection scheme where a node requesting a relay can find a relay in $O(1)$ hop. As mentioned earlier, quickly finding a relay is necessary to reduce call establishment time and recover from relay churn. The key idea to accomplish this goal is to construct a two tier peer-to-peer network. All peers in the top tier provide routing services and can also potentially provide relay services. The peers form the top tier network using any structured or unstructured p2p protocols. Each peer maintains a data structure called a routing table to maintain connectivity with other peers in the overlay. Each entry in this table contains the network address and round-trip time of a reachable peer in the overlay. As part of keep-alive messages to check the liveness of entries in its routing table, a peer also exchanges information with its routing table entries on how many relay calls they can support, their uptime and the time for last user keyboard or mouse activity.

The nodes in the lower tier are connected to peer(s) in the top tier that are close by in terms of network latency and may need a relay peer for establishing a media session. A node in need of a relay sends a request to its connected peer which consults its routing table and returns to the requesting node a set of available relays. If none of the peers in the routing table can fulfill the relay request, the peer forwards the request to a randomly selected peer in its routing table, which in turn consults its routing table for available relay peers. The number of forwarding hops is bounded by a constant. If the number of relay requests are low and uniformly distributed across all peers, this scheme is likely to find a relay in $O(1)$ hops. We refer to this scheme as *local-random* scheme because it selects a relay by leveraging the local overlay view of a peer. This scheme is in contrast to a global random scheme, which has knowledge of all relays in the system and randomly picks a relay from this global pool. Figure 6 shows an illustration of this scheme.

We evaluate the performance of this scheme through simulations and use Chord [25] as the overlay protocol. Each Chord peer maintains a randomized routing table [14] instead of a deterministic table. This is so because Godfrey *et al.* [14] showed that for the same churn rates, randomized scheme for populating routing tables has a better performance than a deterministic one. Intuitively, local-random scheme may have poor performance when relay requests are concentrated on a few peers. However, this issue is addressed by a peer forwarding a relay request to a randomly selected peer in its routing table.

The metric for evaluating the performance of this scheme is its ability to find a relay compared to a scheme with global

knowledge of all relays for an increasing number of relay requests. The inability to find a relay impacts the success rate of relayed calls (equation (1)). The relay search is likely to fail when the number of relay requests is close to or exceeds the network capacity of the peers. If few relays are relaying calls, then local-random scheme is likely to find a relay. However, this may not be the case when the number of relay requests is close to the capacity of the system. Figure 7 plots the percentage of calls that fail to find a single relay. For the results shown, the local-random scheme did not forward the relay request to any peers. The x-axis is the ratio of the applied load to the total relay capacity of all relays. The figure shows that the performance of local-random scheme is poor when there are few relays that can relay the calls. However, it gives comparable performance in terms of percentage of dropped calls due to relay failure even under heavy relay request load.

5.2 User Annoyance

A key difference between p2p file-sharing and communication systems is in their approach to free-riders. The tit-for-tat mechanism in BitTorrent-like filesharing mechanisms aims to minimize the impact of free-riders who are not willing to share or are behind restrictive NAT and firewalls. Such nodes can only download files at a reduced rate [2]. Reducing rate may not be an option in p2p communication networks because it can affect the quality of audio, video, and conference calls. Thus, in contrast to a p2p file-sharing system, a p2p communication system must provide acceptable service to nodes behind restrictive NATs and firewalls. This key requirement means that nodes with unrestricted connectivity must relay calls for nodes with restrictive network connectivity and the relayed calls may interfere with user applications running on these altruistic peers. We refer to such interference as 'user annoyance'.

We focus on characterizing the user annoyance and augmenting the relay selection scheme to minimize user annoyance. User annoyance for relayed calls can also be reduced by providing incentives. However, in a system where proportion of relayed calls is much smaller than the number of available relays, it may be possible to avoid peers where a relay call is likely to cause a high interference with the user applications, and thus bypassing the issue of providing incentives.

The question is how to measure user annoyance. Since relay jobs are network centric and since it is difficult to accurately estimate the perceivable impact of the relay jobs on user applications, we use the spare network capacity to estimate user annoyance. This simplistic measure may not accurately measure user annoyance; however, it is more practical than the other approaches. The higher the spare network capacity, the smaller the likelihood of annoyance of a user whose machine is used as a relay. A peer can periodically perform its uplink and downlink capacity measurements (say every 30 minutes) and by determining the current network usage, gauge its spare network capacity which it can then advertise to peers in its routing table. We use this technique in our PlanetLab implementation (Section 5.4).

5.2.1 Estimating Spare Network Capacity

Measuring user annoyance requires estimating of the capacity of the network link. Unlike CPU, memory, and disk, it is non-trivial to estimate the network capacity. To an extent, this depends on the type of network link. On point-to-point dialup connections, the maximum link speed is typically determined by the speed of the modem. As DSL and cable Internet penetrates homes and the use of WiFi routers at home becomes common, a device no longer directly connects to the ISP in a way similar to dialup; rather, a device connects to a WiFi router which connects to the DSL or cable modem, which in turn is connected to the ISP. Using the link speed of the connected WiFi link will highly overestimate the machine-to-ISP link capacity.

We suggest three approaches for determining the machine-to-ISP link capacity in the presence of intermediate devices such as WiFi routers, and cable or DSL modems. The first approach uses the fact that link capacity is agreed upon between ISP and customer when the latter purchases a broadband plan. The idea is to design protocols which allows ISP to pass this link capacity to the cable or DSL modem which in turn passes this information to downstream devices such as WiFi routers or laptops. This idea can be implemented as a DHCP option, for example. The problem with this approach is that ISPs typically perform statistical multiplexing on multiple flows, and the instantaneous capacity of the link may be less than the purchased capacity. Also, this technique requires changing the already deployed cable/DSL modems and WiFi routers, which is a non-trivial task. Nevertheless, it is a solution that does not require p2p applications to perform any network capacity measurements. In the second approach, a p2p application can perform measurements to estimate the link capacity by sending a train of packets to other peers in the p2p network using tools such as LinkWidth [11]. Third, an operating system or the p2p application can keep track of the maximum data rate seen on the link within a recent time window and use it as an estimate of link capacity. However, this approach heavily depends on the network usage of the machine. We use the second approach in our PlanetLab implementation.

5.3 Heuristics

Besides minimizing user annoyance, it is necessary to minimize the delay of a relayed call and increase its reliability. In essence, this is a multi-objective optimization problem. We devise heuristics to optimize these metrics and evaluate their performance.

In Section 5.1, we constructed a two tier overlay network and peers in the top tier maintain information about the round-trip time, spare network capacity, and uptime of the nodes in their routing table. Peers can exchange this information as part of keep-alive messages. A node searching for a relay then sends a request to its connected peer which applies the heuristics and returns a set of candidate nodes.

Below, we discuss the heuristics for selecting a relay peer from a candidate set returned by the local-random scheme.

- *Random:* Select a random node.
- *NetMax:* Select a node with max. spare network capacity.
- *MinDelay:* Select a node that has the smallest RTT.
- *Threshold:* Select a node that does not add more than 200 ms of on top of the direct network delay between caller and callee, and has maximum spare bandwidth. If no candidate meets the criteria, randomly select one.

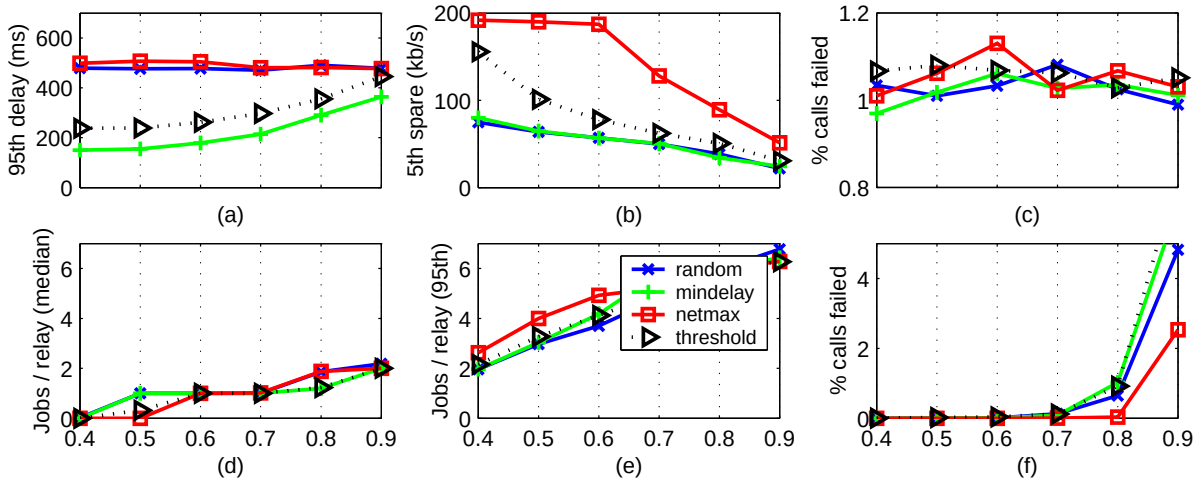

Figure 8: The x-axis represents the ratio of bandwidth consumption of total number of calls in the system to the total network capacity of all nodes. (a) 95th delay (ms) of completed calls (b) 5th percentile of spare network capacity (c) percentage of failed calls due to relay churn (d)(e) median and 95th percentile of number of jobs per relay (f) percentage of calls that fail to find a relay.

Figure 8 shows results for these heuristics. The results were obtained through simulations on a 1,740 Chord network, with node lifetimes taken from the Skype data set as described in Section 4.5. We assume that the network latency between the clients and their connected peers is very small(close to zero). This assumption is reasonable because clients connect to minimum latency peers to use overlay services. The heuristics are evaluated according to several metrics. The first metric is the 95th percentile of the total delay of a relayed call minus the direct latency between session peers. The second metric is the median and 95th percentile of number of jobs per relay. The third metric is the 5th percentile of absolute spare capacity on relay nodes. The fourth metric is the percentage of calls that fail due to relay failure. The last metric is the percentage of calls that cannot find a relay.

The results shows that *MinDelay* heuristic gives the best delay performance (Figure 8(a)). *NetMax* heuristic ensures that relays with large spare network capacity are preferred over relays with small spare capacity and achieves the best performance for user annoyance. However, this has a consequence that more calls can be assigned to high capacity nodes, making these calls more vulnerable to relay failure (Figure 8(c)). The *Threshold* approach gives the best performance in terms of minimizing latency and user annoyance. The *Threshold* scheme has a slightly high call drop rate due to failed relays but this can be improved by biasing relay selection towards idle nodes, e.g., machines with no keyboard or mouse activity within a time period. All heuristics have similar performance in terms of their ability to find a relay under increasing load.

As mentioned in Section 5.2, spare network capacity is a simplistic measure to estimate user annoyance. In addition to spare network capacity, machine idle time is a useful measure for relay selection. The idea is to select a relay with spare capacity that has been idle for sometime. The use of idle time as a relay selection metric is motivated by SETI@home project [4]. SETI@home runs compute jobs as a screen saver on idle machines that are distributed around the world. Using this approach in a p2p communication network, peers participating in the top-level hierarchy inform peers in their routing table how long they have been idle and whether they are in the screen saver mode. A node in need of a relay then selects a peer that meets the delay constraint, has been idle, and has the maximum spare capacity.

Figure 7 showed that search for relays start to fail when the requests for relay calls are close to or exceed the total network capacity of the system. This is unacceptable for an overlay provider like Skype. The only solution for the overlay provider is to provision the p2p applications with centralized media relay servers. When nodes establishing a media session fail to find a relay peer, they send a request to the media relay server to relay the media session. Such a hybrid solution is necessary for a commercial p2p VoIP provider.

5.4 PlanetLab Deployment

To examine the feasibility of relay selection schemes, we have implemented the *Random* and *Threshold* scheme in our OpenVoIP [8] system. OpenVoIP is a two-level hierarchical overlay network deployed on PlanetLab that uses the Kademlia DHT [23]. We have successfully scaled the top-level network to 1,000 peers that run on 500 PlanetLab machines. Each peer in the top-level network fully participates in the overlay and can act as a relay peer using TURN protocol [24]. Further, each peer periodically performs uplink and downlink capacity measurements and shares this information with its routing table nodes. In addition, a peer also shares its uptime with its routing table nodes. We have integrated p2p functionality with an open source SIP phone. This P2PSIP phone fully participates in the overlay if it is not behind a NAT or a firewall. Otherwise, it participates

as a client. When two P2PSIP phones behind a restrictive NAT cannot establish a media session directly, they use a peer in the top-level hierarchy to relay the media session.

We have implemented the *Random* and *Threshold* scheme for relay selection. Our implementation of the *Threshold* scheme uses delay and spare network capacity metric. We do not make use of a SETI@home like technique for determining whether a machine is idle as PlanetLab machines are not user desktop machines. The results for the *Threshold* scheme indicate that relay selection is biased towards nodes with maximum spare network capacity and low latency. We note that these relayed calls are real voice calls between two SIP user agents and are not emulated.

6. RELATED WORK

There has been extensive research on constructing proximity aware DHTs [22] and to minimize the impact of churn on DHT routing [14]. Ren *et al.* [20] showed through measurements that many relay peer selections in Skype are sub optimal, waiting time to select a peer can be quite long, and there are a large number of unnecessary probes. They designed an autonomous system aware p2p protocol (ASAP), which considers autonomous systems into peer relay selection. Their approach suffers from three limitations. First, when using DHTs, the network address of all relay peers within the same AS can get stored on a single node, creating a single point of failure. Second, their techniques do not incorporate interference of a relay session with the user applications. This is critical because users will not altruistically run a p2p application if it actively interferes with their applications. Finally, they provide no guidance on how many relay peers are needed to achieve desired reliability. Leonard *et al.* [18] analyze node connectivity in DHTs for exponential and pareto residual lifetimes. However, our focus is on charaterizing the reliability of relayed calls. Godfrey *et al.* [14] analyzed the impact of churn on the DHT routing performance and suggested techniques to minimize such impact. Our relay selection techniques uses their random selection approach. However, it is imperative to explicitly devise schemes to prevent dropped calls. Tan *et al.* [27] present analysis to improve the reliability of DHT-based multicast by improving its delivery ratio. Delivery ratio is not an appropriate metric to for analyzing reliability in peer-to-peer communication systems.

Connectivity issues due to NAT and firewalls also arise in p2p file sharing networks such as Kazaa [19] and BitTorrent [1]. BitTorrent allows nodes behind restrictive NAT and firewalls to download file chunks, albeit at a lower rate. To improve the download rate, BitTorrent FAQ recommends users to configure the 'port forwarding' feature of NATs [2]. Lowering rate is not an option in p2p communication networks because it can impact the quality of a call. Further, a user of the p2p communication may find it difficult to configure the NAT device and may abandon the p2p application in favor of a configuration-less communication application.

7. CONCLUSION AND ONGOING WORK

We have formalized the notion of reliability in peer-to-peer communication systems and designed a simple analytical model that predicts the reliability of relayed calls as a function of node lifetime and call duration distributions. Our analysis shows that for call duration of 30 minutes or less, three relays are sufficient to achieve a 99.9% call success rate for Skype node lifetimes. We have presented two techniques for relay selection, namely, no-replacement and with-replacement, and used reliability theory to analyze them. We have observed that Skype follows a 2-relay with-replacement scheme, and it uses periodic recovery to replace a failed relay, and the search period is more than a minute. Our results indicate that exponential distribution, despite its limitations, is useful in analyzing the reliability of relayed calls.

We introduced the notion of user-annoyance which measures the interference of a p2p application with other applications running on a machine. We have devised a distributed technique to find a relay in $O(1)$ hop. We augment this technique to find a relay that minimizes latency and user-annoyance. Finally, we have explored the feasibility of our relay selection schemes on a 1,000 node peer-to-peer communication system deployed on PlanetLab. In the future, we will extend our reliability analysis to p2p audio and video conferencing.

8. REFERENCES

[1] BitTorrent [accessed March 2010]. `http://www.bittorrent.com/`.

[2] BitTorrent FAQ [accessed March 2010]. `http://dessent.net/btfaq/\#ports`.

[3] NetPeeker [accessed March 2010]. `http://www.net-peeker.com/`.

[4] SETI@home [accessed March 2010]. `http://setiathome.ssl.berkeley.edu/`.

[5] Skype Silk codec [accessed March 2010]. `https://developer.skype.com/silk/`.

[6] Vonage [accessed March 2010]. `http://www.vonage.com/`.

[7] S. Baset and H. Schulzrinne. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol. In *Proc. of IEEE INFOCOM*, Barcelona, Spain, April 2006.

[8] S. A. Baset and H. Schulzrinne. OpenVoIP: An Open Peer-to-Peer VoIP and IM System. In *Proc. of SIGCOMM (demo)*, Seattle, WA, USA, September 2008.

[9] A. Birolini. *Reliability Engineering: Theory and Practice*. Springer-Verlag, 2004.

[10] D. Bryan, P. Matthews, E. Shim, D. Willis, and S. Dawkings. Concepts and Terminology for Peer-to-Peer SIP. Internet draft (work-in-progress), July 2008.

[11] S. Chakravarty, A. Stavrou, and A. Keromytis. LinkWidth: A Method to Measure Link Capacity and Available Bandwidth using Single-End Probes. Technical Report (cucs-002-08), Department of Computer Science, Columbia University, January 2008.

[12] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu. Characterizing Residential Broadband Networks. In *Proc. of IMC*, San Diego, California, USA, 2007.

[13] B. Ford, P. Srisuresh, and D. Kegel. Peer-to-Peer Communication Across Network Address Translators. In *Proc. of USENIX Tech. Conf.*, Anaheim, CA, USA, 2005.

[14] P. B. Godfrey, S. Shenker, and I. Stoica. Minimizing Churn in Distributed Systems. In *Proc. of SIGCOMM*, Pisa, Italy, 2006.

[15] S. Guha, N. Daswani, and R. Jain. An Experimental Study of the Skype Peer-to-Peer VoIP System. In *Proc. of IPTPS*, February 2006.

[16] K. P. Gummadi, S. Saroiu, and S. D. Gribble. King: Estimating Latency Between Arbitrary Internet End Hosts. *SIGCOMM Comput. Commun. Rev.*, 32(3):11–11, 2002.

[17] W. Kho, S. Baset, and H. Schulzrinne. Skype Relay Calls: Measurements and Experiments. In *Proc. of IEEE Global Internet Symposium*, Phoenix, AZ, USA, 2008.

[18] D. Leonard, V. Rai, and D. Loguinov. On Lifetime-based Node Failure and Stochastic Resilience of Decentralized Peer-to-Peer Networks. In *Proc. of SIGMETRICS*, Banf, Alberta, Canada, June 2005.

[19] J. Liang, R. Kumar, and K. Ross. Understanding Kazaa, 2004.

[20] S. Ren, L. Guo, and X. Zhang. ASAP: an AS-Aware Peer-Relay Protocol for High Quality VoIP. In *Proc. of ICDCS*, Lisbon, Portugal, 2006.

[21] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling Churn in a DHT. In *Proc. of USENIX Tech. Conf.*, Anaheim, CA, USA, 2004.

[22] S. C. Rhea. *OpenDHT: A Public DHT Service*. PhD thesis, Berkeley, CA, USA, 2005.

[23] J. Risson and T. Moors. Survey of Research towards Robust Peer-to-Peer Networks: Search Methods. RFC 4981, September 2007.

[24] J. Rosenberg, R. Mahy, and P. Matthews. Traversal Using Relays around NAT (TURN). Internet draft (work-in-progress), February 2009.

[25] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. *IEEE ToN*, February 2003.

[26] K. Suh, D. R. Figuieredo, J. Kurose, and D. Towsley. Characterizing and Detecting Relayed Traffic: A Case Study using Skype. In *Proc. of IEEE INFOCOM*, Barcelona, Spain, April 2006.

[27] G. Tan and S. A. Jarvis. Stochastic Analysis and Improvement of the Reliability of DHT-Based Multicast. In *Proc. of IEEE Infocom*, May 2007.

[28] X. Wang, Z. Yao, and D. Loguinov. Residual-Based Estimation of Peer and Link Lifetimes in P2P Networks. *IEEE/ACM Transactions on Networking*, 17(3):726–739, 2009.

[29] I. V. Zaliapin, Y. Y. Kagan, and F. P. Schoenberg. Approximating the Distribution of Pareto Sums. *Pure and Applied Geophysics*, May 2005.