ASAP: an AS-Aware Peer-Relay Protocol for High Quality VoIP

Shansi Ren, Lei Guo, and Xiaodong Zhang
Department of Computer Science and Engineering
The Ohio State University
{sren, lguo, zhang}@cse.ohio-state.edu

Abstract

Peer-to-peer (P2P) technology has been successfully applied in Internet telephony or Voice over Internet Protocol (VoIP), such as the Skype system, where P2P is used for both searching clients and relaying voice packets. Selecting one or multiple suitable peers to relay voice packets is a critical factor for the quality, scalability and cost of a VoIP system. In this paper, we first present two sets of intensive Internet measurement results to confirm the benefits gained by peer relays in VoIP, and to investigate the performance of the Skype system. We obtain the following results: (1) many relay peer selections are suboptimal; (2) the waiting time to select a relay node can be quite long; and (3) there are a large number of unnecessary probes, resulting in heavy network traffic to limit scalability of the VoIP system. Our further analysis shows that two main reasons cause these problems. First, the peer selections do not take Autonomous System (AS) topology into consideration, and second, the complex communication relationships among peers are not well utilized. Motivated by our measurements and analysis, we propose an AS-aware peer-relay protocol called ASAP. Our objective is to significantly improve VoIP quality and system scalability with low overhead. Our intensive evaluation by trace-driven simulation shows ASAP is highly effective and easy to implement on the Internet for building large and scalable VoIP systems.

1 Introduction

With the continuous increase of Internet bandwidth and rapid advancement of P2P applications, VoIP technology has become a communication alternative for many Internet users. We envision that Internet telephony will soon become a popular voice communication vehicle due to its low cost and convenience to many Internet end users. VoIP technology has been investigated from different perspectives recently ([17, 19, 20]).

P2P technology has quickly emerged in Internet telephony, where P2P is used for both searching clients and relaying voice packets. Little work has been done on peer node selections to relay voice packets, which is a critical factor for the quality, scalability and cost of a VoIP system. There are some overlay routing projects, represented by MIT's Resilient Overlay Network (RON) [4], and University of Washington's Scalable One-hop Source Routing (SOSR) [11], which are proposed to mitigate Internet path failure through one intermediary node overlay routing. However, they are not specifically designed for VoIP systems, thus, are not suitable to VoIP peer relay selections. To the best of our knowledge, Skype is the first and only commercial software that has used P2P technology in both user search and voice packet relay. Millions of people are enthusiastically using the Skype VoIP system in many applications.

We will address several important technical issues of P2P supported VoIP systems in this paper.

• Having conducted intensive Internet measurements based on Border Gateway Protocol (BGP) routing tables and other dynamic information, we show the benefits gained and important roles played by the relaying peers in a VoIP system. This measurement study establishes a technical foundation for building a large and scalable peer-relayed VoIP system.

• We have looked into the Skype routing path selections by running extensive experiments via its conversation sessions. We have identified several performance limits of the Skype system: (1) many relay peer selections are suboptimal; (2) the waiting time to select a relay node can be quite long; and (3) a large number of unnecessary probes are generated, resulting in heavy network traffic to limit scalability of the VoIP system. With these limits, Skype may not guarantee a stable VoIP quality and a high scalability to a large number of clients. In fact, Skype users, including ourselves, have experienced some quality and scalability difficulties from time to time in practice. To our knowledge, these kinds of measurements have not been systematically reported in the research literature.

• Our further analysis shows two main reasons behind our findings in Skype. First, the peer selections do not take the Autonomous System (AS) into consideration, and second, complex communication relationships among peers are not well utilized in peer selections.

• Motivated by our measurements and analysis, we propose an AS-aware peer-relay protocol called ASAP. Our trace-driven simulation shows that ASAP significantly improves VoIP quality and system scalability with low overhead. It is also easy to implement ASAP on the Internet for building large and scalable VoIP systems.

2 VoIP Application and its Quality Requirement

Based on the underlying transmission media, VoIP applications are generally operated in a pure or hybrid environment. *Hybrid VoIP applications* partly use the public telephone switch network for local accesses, and partly use IP networks for wide area data transmission. *Pure VoIP applications*, such as Skype, solely rely on IP networks. On the other hand, the VoIP communication process consists of signaling and voice packet transmission. In this paper, we focus on the voice packet transmission of pure VoIP applications. VoIP is a real-time and interactive Internet application, and generally has the following quality requirements.

• *VoIP Speech Quality Requirement.* Mean Opinion Score (MOS) is a subjective metric widely used to evaluate human feeling speech quality and is given on a scale of 1-5, as defined in [1]. High MOS means high speech quality, and a MOS below 3.6 likely causes listeners' dissatisfaction. For a VoIP application, its speech quality is affected by many factors, such as codec algorithms, playout buffer size, error concealment mechanisms, packet delay and loss. Among these factors, packet delay and loss are network factors. High MOS demands short end-to-end latency and low end-to-end packet loss rate.

• *Short End-to-End Latency Requirement.* The International Telecommunication Union (ITU) G.114 [3] recommends 150 ms as the upper limit for one-way delay for most interactive applications. Therefore, in VoIP applications, 150 ms is also the one-way delay upper bound for normal human hearing.

• *Low End-to-End Packet Loss Rate Requirement.* The VoIP application also demands low packet loss rate. In [17], with Nortel Networks contributed results, the authors observed that for codecs including G.711, G.729, G.729A, and G.723.1, without packet loss concealment, MOS drops by roughly one unit every 1% of packet loss.

3 Several Fundamental Issues of Overlay Routing for VoIP

An effective routing protocol for low latency is an important technical foundation to ensure the quality of VoIP. In this section, we present an experimental study on direct IP routing and peer relay routing to examine whether they can meet the quality requirement of VoIP communication. Fig. 1 shows the experimental procedures of our methodology, which will be further explained in the following subsections. [1]

3.1 Collecting and Grouping P2P IPs

The Internet consists of many Autonomous Systems (ASes), each of which is administrated by a single organization. An AS can enforce its own routing policies, and inter-AS routing on the Internet is specified by the Border Gateway Protocol (BGP). Because of this structure, nodes within one AS are normally close to each other physically.

We have run the Gnutella network crawler in [13] and have collected 269,413 distinct IP addresses of Gnutella peers on the Internet. We have collected a large number of BGP routing table entries and BGP updates with timestamp of 2005-09-26 (US Eastern Time) from RouteViews (http://www.routeviews.org/), RIPE RIS (http://www.ripe.net/projects/ris/rawdata.html), and China CERNET (http://bgpview.6test.edu.cn/). From these BGP routing table entries and updates, we build an IP prefix to origin AS mapping table and extract the AS-AS connection relationships. Using the constructed mapping table, we group IPs with the same origin AS or with the same longest matched prefix into one cluster at the AS level or at the IP prefix level, respectively. In our grouping results, out of 269,413 distinct IP addresses, there are 103,625 IP addresses matching 7,171 IP prefixes, and belonging to 1,461 ASes. It has been justified in [14] that by grouping hosts based on IP prefixes, we can create clusters in which hosts are close to each other. Then we randomly choose one IP out of each cluster as the cluster delegate. A latency measurement between each pair of cluster delegates will provide us with a basic routing benchmark.

3.2 Routing Round-Trip-Time (RTT) Measurements

We have used the tool King [12] to estimate the latency between any two online P2P end hosts. King mainly relies on two observations: (1) most end hosts in the Internet are located close to their Domain Name System (DNS) name

[1] Relevant programs and tools we have written and used, and detailed measurement results can be found at http://www.cse.ohio-state.edu/∼sren/VoIP-Peer-Relay/.

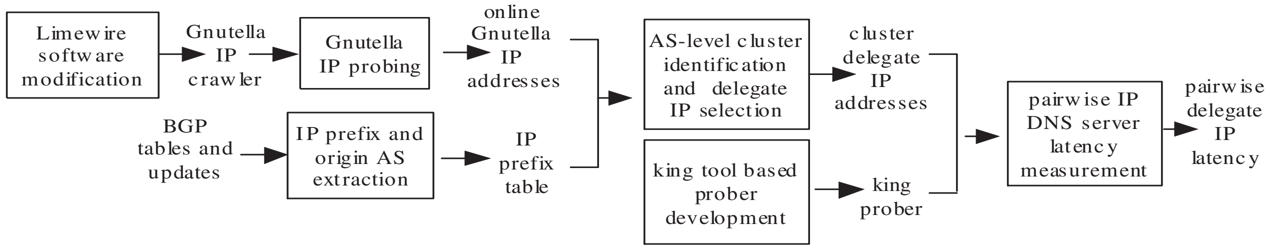

Limewire software modification
Gnutella IP crawler
Gnutella IP probing
online Gnutella IP addresses
BGP tables and updates
IP prefix and origin AS extraction
IP prefix table
AS-level cluster identification and delegate IP selection
king tool based prober development
cluster delegate IP addresses
king prober
pairwise IP DNS server latency measurement
pairwise delegate IP latency

Figure 1. All pairwise cluster delegate IP latency measurement procedure.

servers; and (2) recursive DNS queries can be used to measure RTTs between pairs of DNS servers. We have further automated the King tool to conduct a large scale pairwise IP RTT measurements.

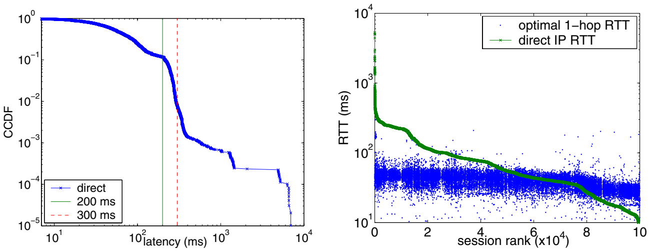
Using the King tool on our end hosts, we have measured RTTs of all pairs of delegate IPs. Note that in our measurements, there is a fraction of recursive DNS queries that are not responded to. The number of delegate IP pairs is 2,130,140, to which we have obtained 1,498,749 responses.

We have conducted experiments to emulate the scenario that a host B relays packets between two end hosts A and C. Host A continuously sends messages to host B at a rate close to 100 Kbps. The relay delay is the duration starting from the time that a packet arrives at B's network adaptor queue, gets copied to B's memory, and ending at the time that this packet arrives at B's network adaptor's queue for transmission. In thousands of relay delay experiments in our departmental 100 Mbps network, the average relay delay is around 12 ms. In our paper, we conservatively use 20 ms as the packet relay delay, and 40 ms as the round-trip relay delay to accommodate other potential delay.

3.3 Benefits of Overlay Routing

We have conducted all pairs of delegate IP RTT experiments by randomly choosing two IPs in our collected Gnutella IP pool to represent the communication between them, each of which we call a *session*. We randomly generate 10^5 such sessions. For each pair of delegate IPs of hosts A and B among the collected online IP pool, we iterate through every other delegate IP C, and use the sum of the RTT of A-C, the RTT of C-B, and twice of relay delay to estimate the RTT of the one-hop overlay path A-C-B. We have iterated through every possible one-hop relay node C in our collected peer population to obtain the optimal one-hop overlay routing path RTT.

The distribution of direct IP routing latency based on DNS RTT measurements is shown in Fig. 2(a). It shows that in our randomly generated 10^5 sessions, there are about 10^3 sessions with direct IP routing RTTs greater than 300 ms, and there are 10^4 sessions with direct IP routing RTTs greater than 200 ms. The remaining sessions all have direct IP routing latency smaller than 200 ms. There are about 10 sessions whose RTTs are greater than 5 seconds.



(a) Direct IP routing (b) Direct and optimal 1-hop routing

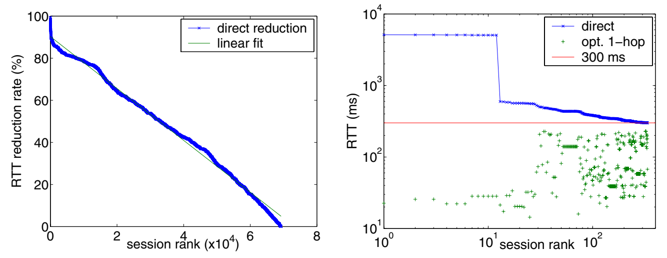
Figure 2. RTT distribution.

Fig. 2(b) shows the RTTs of direct IP routing between the two end hosts and the corresponding RTTs of optimal one-hop relay. There are about 60% of the sessions whose optimal one-hop RTTs are shorter than their corresponding direct IP routing RTTs. Most optimal one-hop RTTs are below 100 ms.

Fig. 3(a) shows the RTT reduction rate of optimal one-hop peer relay for those sessions whose direct IP routing RTTs are longer than their corresponding optimal one-hop RTTs. The RTT reduction rate r is defined as $r = \frac{\textit{direct RTT - optimal 1-hop RTT}}{\textit{direct RTT}}$. As shown in the figure, the reduction rate is evenly distributed across these sessions.

As mentioned earlier, an RTT above 300 ms results in unsatisfactory user experience of VoIP applications. Fig. 3(b) shows the RTT of direct IP routing and corresponding optimal one-hop peer relay for sessions with direct IP routing RTTs exceeding the 300 ms threshold. As shown in the figure, for those sessions with direct IP routing RTTs above 300 ms, the RTTs of corresponding optimal one-hop relays are always smaller than 300 ms, meaning one-hop relay can satisfy the VoIP RTT quality requirement in these cases.

In summary, there are a non-trivial fraction of sessions (varying from 1% to 10% depending on the RTT quality requirement) with direct IP routing RTTs above the RTT threshold for quality VoIP communication. For these sessions, it is always possible to find one-hop overlay relay paths to reduce the RTTs below the threshold. Peer relay plays an important and critical role in improving the quality



(a) Optimal 1-hop RTTs reduction ratio.

(b) Direct and optimal 1-hop RTTs for latent sessions.

Figure 3. RTT reduction.

for Internet applications with stringent latency requirements such as VoIP.

IP routing on the Internet is dependent on the provider-customer and peer-peer commercial contractual relationships between neighboring ASes or Internet Service Providers (ISPs) [9]. Usually a provider AS transits traffic for a customer AS, while a customer AS does not transit traffic for its provider AS. Constrained by this rule, an Internet AS-level routing path usually has the *valley-free* property [9]. Because each AS can enforce its own routing policy, the direct IP routing path between two end hosts is not necessarily the optimal one among all possible routing paths between them, including overlay routing paths. Under the following two conditions, overlay routing paths can be faster than the direct IP routing paths.

(1) An AS in a direct routing path is congested or failed. As shown on the left of Fig. 4, we consider the routing paths between two end hosts in AS A and AS C. Their direct IP routing path is A-D-G-H-F-C, in which AS H is congested. While the one-hop overlay routing path between AS A and AS C through AS B is A-D-G-E-B-E-F-C, which does not contain the congested AS H on the direct IP routing path. Even worse, the direct routing path may contain failed ASes, while overlay routing can bypass them. In these cases, overlay routing latency can be shorter than the direct IP routing path.

(2) Multi-homed customer ASes can further improve overlay routing. A multi-homed customer AS connecting to multiple upstream provider ISPs can act as the intermediary relay to transit traffic for its provider ISPs, and this one-hop relay path can have shorter AS hops than that of the direct IP routing path. Consider the annotated AS graph on the right of Fig. 4, in which AS B has multi-homed connections to two providers AS D and AS E. For the two end hosts in AS A and AS C, respectively, the direct IP routing path between them is A-D-F-H-I-G-E-C, which has 7 AS hops, while its overlay routing path through AS B is A-D-B-E-C, which has only 4 AS hops and is 3 AS hops shorter. Because the path latency is correlated to the AS hops on this path [2], despite of the relay delay at AS B, the RTT of overlay routing is highly likely to be shorter than that of direct IP routing.

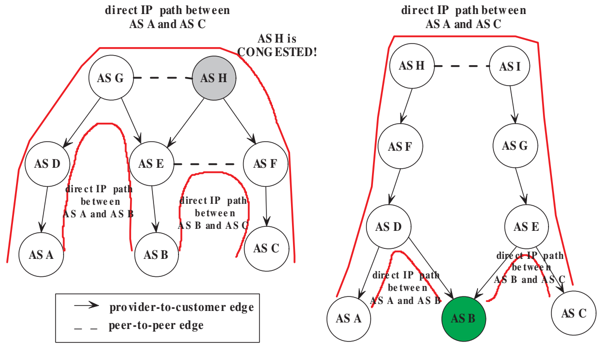


Figure 4. Two scenarios that overlay routing is faster than direct IP routing.

4 Related Work on Overlay Relay Node Selections

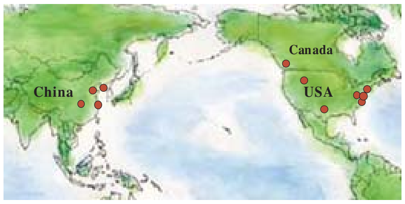
Since overlay routing can greatly improve VoIP quality when direct IP routing cannot satisfy the requirement, quickly choosing appropriate relay nodes with low overhead is a critical problem. We have looked into the suitability of existing peer node selection methods for VoIP.

Currently representative overlay relay node selection methods include RON [4], and SOSR [11]. A RON-like relay node selection method is proposed in [19]. These methods focus on utilizing one-hop intermediary node to mitigate direct IP routing path failure problems. For this reason, when these methods are used to find relay nodes for VoIP applications, the paths may not meet the VoIP quality requirements. Furthermore, RON needs dedicated nodes to act as relay, and needs to timely measure all pairwise latencies among those dedicated nodes, and thus, is difficult to scale. While SOSR randomly chooses one hop intermediaries, it cannot guarantee to find a short one-hop routing path with a moderate number of probings.

Being slightly relevant but different from the problem of relay node selection to meet VoIP quality requirement, routing underlays are proposed [18] to optimize overlay routing based on probing. On the other hand, an earliest-divergence (ED) heuristic is proposed to find independent routing paths between two end hosts [8]. When used in VoIP applications, ED cannot guarantee to find good relay nodes to satisfy the VoIP quality requirements.

5 Experiments and Analysis of Skype

Skype (http://www.skype.com) is the most popular P2P-based commercial VoIP system, which has more than 3 million online users at any time on the Internet. Skype utilizes peers for both user search and voice packet relay, and can dynamically switch to better paths upon quality degradation of the current path [6]. Although measurement study [10]



site	location
1-6	Williamsburg, VA
7	Reston, VA
8	Beltsville, MD
9	Jersey City, NJ
10	Austin, TX
11	Bozeman, MT
12	Vancouver, Canada
13	Jingzhou, China
14-15	Shanghai, China
16	Beijing, China
17	Dalian, China

Figure 5. Skype sites in the measurement experiments.

session no.	1	2	3	4	5	6	7
caller-callee	3-5	1-11	1-7	1-14	1-3	1-16	1-15
session no.	8	9	10	11	12	13	14
caller-callee	1-15	1-9	1-17	1-13	1-12	6-8	2-10

Table 1. 14 example Skype calling sessions.

characterizes dynamic behaviors of Skype supernodes, due to the fact that Skype uses a close standard and encrypts its data packets, its overlay routing path selection method is unknown to the public.

We have recruited a number of volunteers and collected extensive Skype communication workload between Williamsburg, Virginia, USA and 11 locations across China and North America. Fig. 5 shows the summary of end host locations in our experiments. All voice communications in our experiments are performed under Skype software Windows version 1.3.0.57, from 2005-05-22 to 2005-07-30. During each voice communication session, we use the WinDump software (http://www.winpcap.org/windump/) to collect Skype packets on both end hosts. We have developed a trace analyzer to study the collected data using the "pcap" library. We have selected 14 representative sessions out of our collected Skype traces, and use them to study the Skype routing mechanism.

5.1 Skype Node Probing and Major Relay Path Selections

We analyze Skype packet headers collected at the two end hosts in a session to check if they share common destination IP addresses reached from their voice data ports. We found that a Skype session can choose several peer nodes at start-up time, and test their corresponding relay path latencies before a few fast and stable relay nodes are constantly used to transmit voice packets, which we call *major relay nodes*. Their corresponding relay paths are called *major paths*. Major relay nodes and major paths are the critical components for VoIP.

In our measurements, we found some Skype VoIP sessions may use different major paths for forward and backward directions, which we call *asymmetric* sessions. For example, two Skype sessions that both have direct IP routing RTTs of 246 ms use direct IP routing paths as major paths for the forward direction, while using one-hop relay paths as major paths for the backward direction. For the remaining *symmetric* sessions, major paths in 4 sessions use direct IP routing, major paths in 7 sessions use one-hop relay, and 1 session uses two-hop relay. Furthermore, in all 14 sessions, the major paths carry more than 90% of the total transmitted voice data packets.

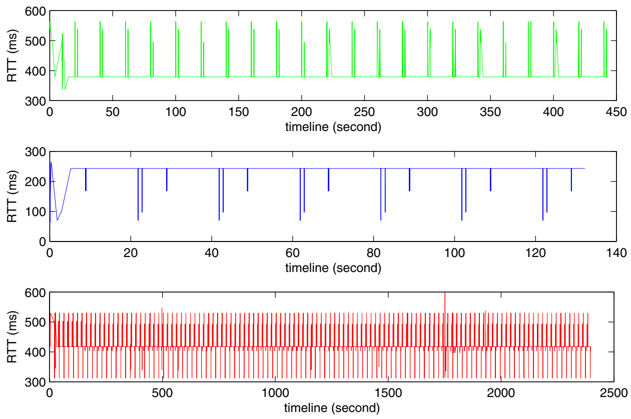


Figure 6. Relay path time-series distribution of sessions 4, 9, and 10.

Having thoroughly analyzed our captured traces, we are able to present several limits of Skype caused by its inefficient relay node selection methods.

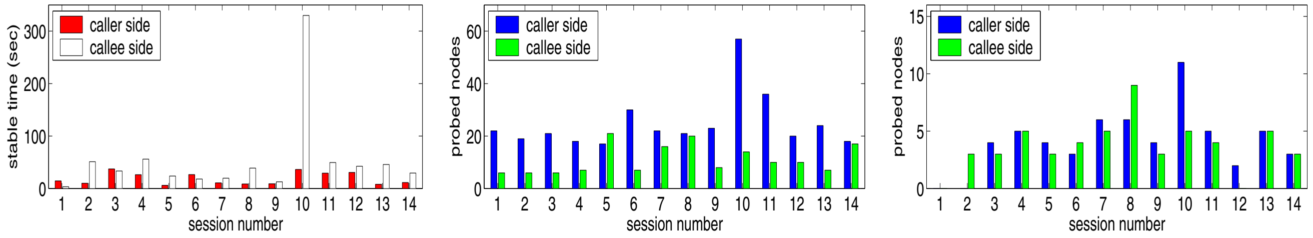
Limit 1: Long VoIP Latency due to Improper Relay Node Selections

We use the tool "King" to measure the latency between the end hosts and the intermediary relay nodes, and add the relay delay of 40 ms to estimate the RTTs of Skype one-hop routing paths.

Fig. 6 shows the probed path RTT distribution in three problematic sessions. As we can see, the major path RTTs of session 4 and session 10 are above 350 ms, which means that the quality of these two sessions is unsatisfactory. Although the major path RTT of session 9 is close to 250 ms, which is below the 300 ms RTT requirement, it probed relay paths with lower RTTs but did not use them later. Session 10 uses two-hop relays, and we calculate the RTT of its major path. Based on our measurement results in Section 3, it is possible to find relay paths whose RTTs are below 300 ms. However, the Skype did not find them for packet relay, though there are 3 million Skype online users at any time. From the time-series latency distributions, we found that Skype sessions probe peer nodes without sufficient prior knowledge of latencies of the probed paths.

Limit 2: Probing Multiple Latent Nodes in the Same AS

For each Skype VoIP session, We use "traceroute" on one end host to check the paths to its probed relay nodes. Through our trace analysis, we found that Skype does not consider the underlying AS topology in relay node selec-



(a) Stabilization time of 14 sessions.

(b) Total number of probed nodes.

(c) Probed nodes after stabilization time.

Figure 7. Stabilization time and probing overhead.

relay node	DNS zone name	relay path RTT
85.64.x.x	barak-online.net	360 ms
85.65.x.x	barak-online.net	359 ms

Table 2. Two relay nodes in session 8.

tions, so that some relay nodes are located in the same AS. The relay paths of these nodes to the two end hosts in a calling session share many common physical links. Once one relay path cannot meet the VoIP one-way latency requirement of 150 ms, the other relay paths are unlikely to meet the requirement as well. Furthermore, once one relay path is lossy, the other paths are likely to be lossy as well [5]. Thus, they should not be probed at the same time.

Table 2 shows two relay nodes probed in session 8. Both paths have RTTs close to 360 ms, and the two relay nodes are located in the same AS.

Limit 3: Taking a Long Time to Find Major Relays

Our trace analysis shows that it takes Skype many probes to find the major relay nodes, which causes a long delay and a non-negligible amount of overhead traffic. We define the duration from session start to the time when major relay nodes are constantly used as its *stabilization time*. Fig. 7 (a) shows that many sessions have long stabilization time. Particularly, in session 10, in the reverse direction from Dalian, China to Williamsburg, USA, the stabilization time is as high as 329 seconds.

A long stabilization time is caused by continuous switches among multiple relay nodes in the session, and the calling quality cannot be guaranteed at the beginning of the session. We call this phenomenon *relay bounce*.

Limit 4: Generating Non-Negligible Overhead

A large amount of unnecessary probes and non-negligible overhead may limit Skype's scalability. The overhead of relay node probing traffic is proportional to the number of probed nodes. We calculate the number of nodes that are probed and the overhead traffic generated in these sessions.

We measure the number of relay nodes these calling sessions have used by checking the port number used for voice packet transmission at two end hosts. Each such a relay node represents a probed routing path. Fig. 7 (b) shows the number of relay nodes each session has probed. We find that many sessions have probed more than 20 nodes before the major relay node is selected. In sessions 10 and 11, 59 and 37 relay nodes are probed, and their direct RTTs measured with "ping" is 238 ms and 355 ms, respectively.

Fig. 7 (c) shows the number of probed nodes to transmit voice data packets after the stabilization time in all 14 calling sessions. From this figure, most sessions have probed 3-6 relay nodes after the stabilization time. This means that the network condition still changes dynamically after the stabilization time.

6 ASAP: An AS-Aware Fast and Low-Overhead Peer Relay Selection Protocol

We have identified two main reasons behind the limits of Skype: (1) relay node selections are AS-unaware; (2) complex communication relationships among peers defined by certain properties of the Internet are not well utilized. Motivated by our measurement studies presented above, we propose an AS-aware peer relay protocol called ASAP to aim at high VoIP quality and high system scalability with low overhead. Our protocol design and algorithm analysis are based on the following Internet properties. (1) In general, peer nodes with the same IP prefix are relatively close to each other [14]. We call the collection of all peer nodes with the same IP prefix an *IP prefix cluster*, or in short, a *cluster*. The direct IP routing latency between two peers in two different clusters can be estimated by the direct IP routing latency between any pair of nodes in their corresponding clusters. (2) With publicly available BGP tables and updates, an up-to-date annotated AS graph can be built ([7, 9]). (3) The number of AS hops and the latency of a direct IP routing path are correlated, and paths with longer AS hops are likely to have longer latency [2]. (4) An Internet AS-level direct IP routing path usually has the valley-free property [9].

6.1 ASAP System Structure and Node Operations

To ease presentation, a node or an end host denotes a computer in the VoIP system, while an AS node denotes

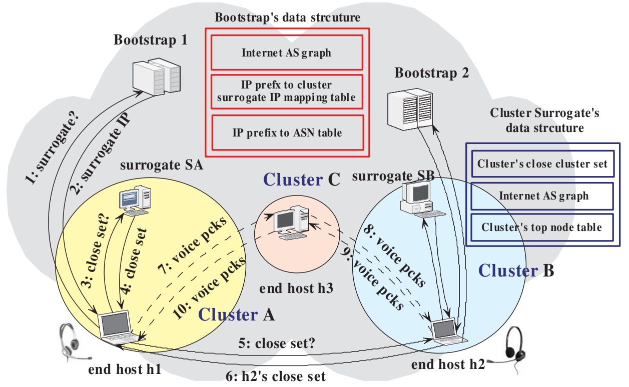


Figure 8. ASAP system structure and end hosts relay node selection process.

a node on the annotated AS graph. We define the following three types of nodes in our ASAP protocol. (1) *Bootstraps*. They are normally system's powerful, dedicated, and always-on servers used to process VoIP user login and nodes' join requests. (2) *Cluster surrogates*. These nodes are powerful and stable with high bandwidth network connections within a cluster. (3) *Normal end hosts*. End hosts can initiate and accept VoIP calling sessions, and can relay voice packets for other calling sessions. Note that except for bootstrap nodes, which are provided for dedicated and always-on service that is part of the system infrastructure, the other two types of nodes are both VoIP peer nodes. Therefore, this system structure is highly cost-effective. Every VoIP peer node is a normal node. A powerful peer node can be a cluster surrogate as well. The ASAP system structure is illustrated in Fig. 8.

Bootstrap nodes play critical roles in any P2P system where important information is stored. In our ASAP system, they provide following functions and service to make the entire system informative and intelligent. (1) Build an annotated up-to-date AS graph. (2) Build an IP prefix to cluster surrogate IP mapping table and an IP prefix to AS number (**ASN**) mapping table. Upon the join request of a new node, translate the node IP to its ASN and its cluster surrogate IP, return the ASN and the cluster surrogate IP to the new node. Note that an AS can have multiple IP prefixes. (3) Disseminate the AS graph to surrogates, so that every surrogate has an up-to-date AS graph. (4) Select new surrogates for clusters upon surrogate failures.

Surrogate nodes volunteer themselves to provide the following service. (1) Maintain the list of IP addresses of all end hosts in their clusters. (2) Periodically contact bootstrap nodes to retrieve the up-to-date annotated AS graph. (3) Periodically run the **construct-close-cluster-set()** algorithm to find close cluster sets for their clusters. The *close cluster set* of a cluster c consists of those clusters whose end hosts have short direct IP routing latencies to any end host in c. (4) Process close cluster set requests from other end hosts in their clusters. (5) Accept nodal information of other end hosts in their clusters. If there are better end hosts, recommend the better end hosts to be new surrogates, become normal end hosts, and notify bootstraps and other end hosts in their clusters of the changes.

```
construct-close-cluster-set($k$, $latT$, $lossT$)
    close cluster table $T = \phi$;
    working queue $Q = \phi$;
    add $a$ to $Q$'s tail;
    while $Q \neq \phi$ {
        remove path $P$ from head of $Q$;
        get last AS node $l$ of $P$;
        for each $l$'s neighbor node $v$ not on $P$ {
            concatenate $P$ and edge $l$-$v$ to get new path $P1$;
            if valley-free($P1$) and hops($P1$) $\leq k$ and $v$ has surrogates
                for each surrogate $cs$ of $v$
                    if ($d$= lat($s$, $cs$)) $< latT$ and ($r$ = loss($s$, $cs$)) $< lossT$ {
                        add tuple ($v$, $cs$, $d$, $r$) to $T$;
                        append $P1$ to tail of $Q$;
                    }
        }
    }
    sort $T$ by latency field;

    return $T$;
```

Figure 9. Surrogate close clusters creation.

End hosts are millions of callers/callees in ASAP and they have the following light duties. (1) Get their ASNs and the surrogate IP addresses of their clusters from bootstraps. (2) Become surrogates in their clusters, if they are the only nodes in their clusters. (3) Periodically publish their nodal information to their surrogates. (4) Run **select-close-relay()** algorithm when they initiate VoIP calls.

Note that nodal information includes bandwidth, continuous online time, node processing power, and other related information.

6.2 Close Cluster Sets Construction and Close Relay Nodes Selection

Fig. 9 shows the **construct-close-cluster-set()** algorithm pseudocode that runs on a cluster surrogate s to construct the one-hop close cluster set for s. Starting from the AS node of s on the AS graph, s does a breadth-first search under valley-free constraints by considering the RTT and loss rate requirements. In this algorithm, k is the number of hops to stop the breadth first search, $latT$ and $lossT$ are the latency and loss rate thresholds to stop path expansion, and lat() and loss() are two functions that get the RTT latency

```
**select-close-relay**($sizeT$)
  one-hop relay set $OS = \phi$;
  two-hop relay set $TS = \phi$;

  obtain $h1$'s close cluster set $S1$ and $h2$'s close cluster set $S2$;
  common set $CS = S1 \cap S2$;
  for each cluster surrogate $r$ in $CS$
    if relaylat($h1$-$r$-$h2$) $< latT$
      for each $ip$ in cluster of $r$
        add $ip$ to $OS$;

  if size($OS$) $< sizeT$
    for each cluster surrogate $r1$ in $OS$ {
      $h1$ obtain $r1$'s close cluster set $OS1$;
      for each cluster surrogate $r2$ in $OS1$
        if $r2$ in $S2$ and relaylat($h1$-$r1$-$r2$-$h2$) $< latT$
          for each $ip1$ in cluster of $r1$ and each $ip2$ in cluster of $r2$
            add $ip1$-$ip2$ to $TS$;
    }

  return $OS$ and $TS$;
```

Figure 10. Session close relays selection.

and the packet loss rate between the two cluster surrogates, respectively. For a direct IP routing path, lat() and loss() can be done by using simple system utilities, such as "ping". $latT$ can be set close to 300 ms, since the one-way delay upper limit of a path is 150 ms to avoid user dissatisfaction. For all sessions with direct IP routing RTTs below 300 ms in Section 3, our experiments show that more than 90% of the sessions with direct IP routing RTTs below 300 ms have no more than 4 AS hops. Therefore, we can set k to 4 in practice, since it is reasonably accurate to infer AS paths by computing the shortest AS hops paths [16].

Fig. 8 illustrates the communication process of two end hosts $h1$ and $h2$ that are engaged in a calling session in ASAP. When the host $h1$ in cluster A joins ASAP, $h1$ first sends a "join" request to a bootstrap (step 1), and the bootstrap returns the IP address of $h1$'s surrogate (step 2). Then $h1$ contacts its surrogate (step 3) to get the close cluster set of its cluster A (step 4). When $h1$ initiates a VoIP calling session to another end host $h2$ in cluster B, $h1$ measures its direct IP routing RTT to $h2$ by using system utilities such as "ping". If $h1$ finds the direct IP routing RTT is above the latency threshold, such as 300 ms, it runs the **select-close-relay()** algorithm in Fig. 10 to obtain close relay nodes. $h1$ contacts $h2$ (step 5) to get $h2$'s close relay nodes (step 6). Finally, by comprehensively considering factors including traffic load conditions and reliabilities of the close relay nodes as well as RTTs and packet loss rates of the relay paths corresponding to these close relay nodes, the two end hosts $h1$ and $h2$ pick the most suitable relay nodes for voice communication. Techniques such as path diversity ([15, 19]), and path switching [20] can be used in combination with ASAP to transmit voice packets between them. In the same figure, $h3$ is selected as relay. Voice packets are transmitted between $h1$ and $h2$ through node $h3$ (steps 7-10).

Fig. 10 shows the pseudocode of the **select-close-relay()** algorithm. It finds one-hop and two-hop relay nodes for a session by intersecting close cluster sets. In this algorithm, relaylat() is a function to get the RTT of an one-hop or two-hop relay path, and can be done by summing up the RTTs of all direct IP routing paths on this relay path and all relay delays on the intermediary relay nodes.

6.3 ASAP System Traffic Load Discussion

On the ASAP bootstrap nodes, the storage required to save an Internet AS graph is small. It takes only 800 KB to store the 2005-09-26 AS graphs, and less than 8,000 ASes contain online end hosts, as shown in Section 3. Furthermore, BGP routing tables do not change frequently. By deploying multiple powerful bootstrap nodes in the system, the traffic load on each bootstrap should be moderate.

Our measurement results show that 90% of the clusters contain no more than 100 online end hosts. In ASAP, every surrogate is the most powerful and reliable VoIP end host in its cluster. Thus, it should not be a problem for most surrogates to handle close cluster set requests from other end hosts in their clusters. For a few large clusters containing close to 1,000 online end hosts, we can select multiple surrogates in them to share the possible heavy load due to the requests from the large number of end hosts.

7 ASAP System Evaluation

7.1 Metrics, Different Routing Methods and Datasets

VoIP user satisfaction demands RTT latency be below 300 ms and MOS be above 3.6. Our evaluation is based on the following metrics: (1) the number of relay paths satisfying the RTT requirements, which we call *quality paths*; (2) the shortest RTTs and the highest MOSs of these quality paths; and (3) the overhead, measured by the number of generated messages to find the quality path relay nodes.

In our evaluation, we consider the following five relay node selection methods. [2] (1) **DEDI** uses dedicated relay nodes (RON-like). (2) **RAND** randomly selects relay nodes (SOSR-like). (3) **MIX** is a combination of RAND and DEDI. Both dedicated nodes and randomly selected

[2] Code we have written for these methods and their corresponding results are available at http://www.cse.ohio-state.edu/∼sren/VoIP-Peer-Relay/.

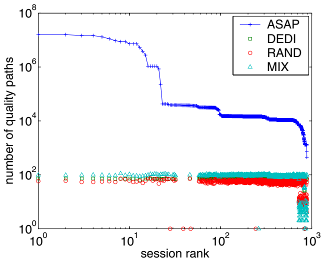


Figure 11. Session quality paths.

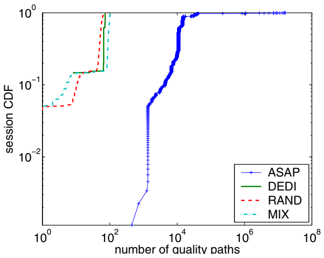


Figure 12. Quality path CDF.

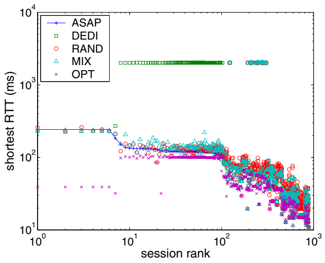


Figure 13. Session shortest RTTs.

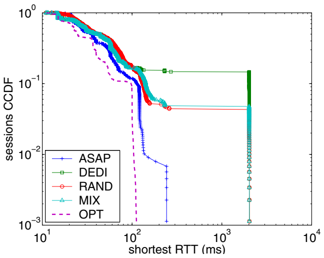


Figure 14. Shortest RTT CCDF.

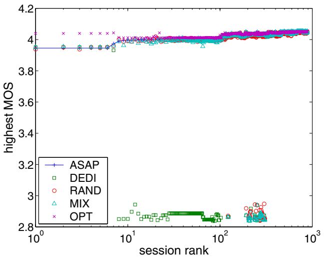


Figure 15. Session highest MOSs.

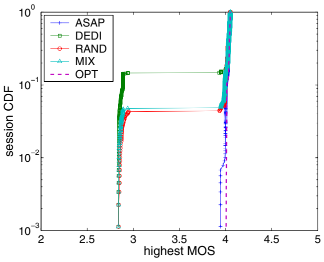


Figure 16. Highest MOS CDF.

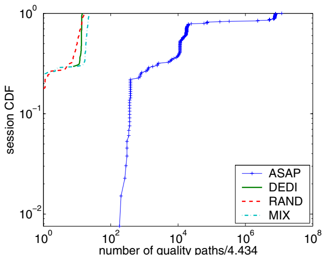


Figure 17. Quality path CDF.

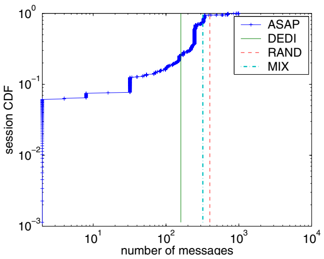


Figure 18. Overhead CDF.

peer nodes are used as relay. (4) **ASAP** selects relay nodes using our AS-aware method. (5) **OPT** always chooses relay nodes that give the shortest overlay routing latency. This is an offline method with all latency data on hand through one-hop and two-hop relay paths iterations.

With the BGP tables and updates we collected (see descriptions in Section 3), we construct annotated AS graphs using the inferring AS relationships algorithm in [9], which contains 20,955 AS nodes and 56,907 AS links. With RTT data we measured in Section 3, we randomly generate 100,000 pairs of peers from our collected Gnutella IP address pool to represent 100,000 VoIP calling sessions, among which there are about 1,000 sessions having their direct IP routing RTTs above 300 ms. Focusing on these sessions, we study the distribution of the metrics under the aforementioned relay node selection methods, including our ASAP method. In these methods, we have set the node relay delay as 20 ms for the same reason as in Section 3.

In our evaluation, DEDI probes 80 nodes in 80 clusters with the largest connection degrees, RAND randomly selects 200 nodes. Correspondingly, MIX probes 160 nodes, including 40 dedicated nodes and 120 randomly probed nodes. We set $sizeTh$ in **select-close-relay**() of ASAP to 300 to start the two-hop relay node selections. Note that in this section, 103,625 IPs are used in Fig. 17, while 23,366 IPs are used in all other figures.

7.2 Comparing ASAP with DEDI, RAND, and MIX

Number of Quality Paths. Fig. 11 and Fig. 12 show that the number of quality paths found by ASAP is much greater than that found by DEDI, RAND, and MIX. It can be seen that by using DEDI, RAND, and MIX, all sessions can find no more than 500 quality paths. While by using ASAP, 90% of the sessions can find more than 10^4 quality paths.

Shortest RTTs. Fig. 13 and and Fig. 14 show that the shortest RTT paths found by ASAP are comparable to those found by the OPT method. In ASAP and OPT, all sessions have shortest RTTs below 115 ms. While in DEDI, RAND, and MIX, more than 5% sessions have shortest RTTs above 1 second.

Highest MOSs. The MOS quality metric can be quantitatively characterized with the end-to-end delay and packet loss rate under the ITU-E-Model when fixing other non-network factors [20]. By fixing the codec as G.729A+VAD, given the RTT and packet loss rate of a path, we use ITU-E-Model to compute its MOS. Based on the real loss rate examples shown in [20], we assume that each path has an average packet loss rate of 0.5%. Fig. 15 and Fig. 16 show that in ASAP and OPT, all sessions have their highest MOSs above 3.85. In contrast, in DEDI, RAND, and MIX, there are about 3% of the sessions have highest MOSs below 2.9, which are unsatisfactory.

7.3 The Scalability of ASAP and its Cost Analysis

We assume that the number of calling sessions in a VoIP system is roughly proportional to the population of its online end hosts. For a given relay node selection method, under different host populations, if the number of quality paths it found divided by the population remains relatively stable, we say this method is *scalable*. Fig. 17 shows the CDF of the number of quality paths of sessions for the four relay node selection methods in a VoIP system with 103,625 online nodes. In Fig. 17, the number of quality paths is divided by 4.434, since 103,625/23,366 = 4.434. We find that for ASAP, the curves of the number of quality paths divided by 4.434 in Fig. 17 have almost the same shape as that of the ASAP curves of the number of quality paths in Fig. 12. In contrast, in Fig. 17, all sessions can find less than 30 quality paths by using the DEDI, RAND, and MIX methods, while in Fig. 12, 90% of the sessions can find more than 90 quality paths. This reflects that ASAP is highly scalable, while DEDI, RAND, and MIX are not.

In our ASAP algorithm, one-hop relay node selection only needs 2 messages, while two-hop relay node selection messages are dependent on the close cluster set size of end hosts. Fig. 18 compares the overheads of these methods. Methods DEDI, RAND, and MIX all probe the fixed number of nodes. On the other hand, the number of messages probed by the ASAP method changes from session to session. With the ASAP method, more than 80% of the sessions generate no more than 300 messages. A very small fraction of sessions with a large number of quality paths generate relatively large number of messages. To further reduce the overhead for these sessions, the end host can choose a fraction of candidate relay nodes to probe in ASAP.

8 Conclusion

In this study, we have confirmed the benefits of peer relay for VoIP applications, and have identified several limits of Skype by extensive measurements on the Internet. Taking the Internet AS topology into consideration, we have proposed an AS-aware protocol called ASAP. We have shown that ASAP is highly efficient and scalable, outperforming all the existing peer selection methods and being close to the performance of an optimal off-line selection method. We plan to implement ASAP on the PlanetLab testbed for further experiments and evaluation in a global Internet environment.

Acknowledgments

We thank William L. Bynum and Junwen Lai for reading the paper and their comments, and thank anonymous reviewers for their constructive suggestions. This work is partially supported by the National Science Foundation under grants CNS-0098055, CNS-0405909, and CNS-0509054/0509061.

References

[1] Methods for subjective determination of transmission quality. ITU-T Recommendation P.800, August 1996.

[2] A passive system for server selection within mirrored resource environments using as path length heuristics. AppliedTheory Communications, Inc., June 1999.

[3] One way transmission time. ITU-T Recommendation G.114, May 2000.

[4] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proceedings of ACM SOSP'01*.

[5] D. Andersen, A. Snoeren, and H. Balakrishnan. Best-path vs. multi-path overlay routing. In *Proceedings of ACM IMC'03*.

[6] S. A. Baset and H. Schulzrinne. An analysis of the skype peer-to-peer internet telephony protocol. In *Proceedings of IEEE INFOCOM'06*.

[7] G. Battista, M. Patrignani, and M. Pizzonia. Computing the types of the relationships between autonomous systems. In *Proceedings of IEEE INFOCOM'03*.

[8] T. Fei, S. Tao, L. Gao, and R. Guerin. How to select a good alternate path in large peer-to-peer systems. In *Proceedings of IEEE INFOCOM'2006*.

[9] L. Gao. On inferring autonomous system relationships in the internet. *IEEE/ACM Trans. Netw.*, 9(6), 2001.

[10] S. Guha, N. Daswani, and R. Jain. An experimental study of the skype peer-to-peer voip system. In *Proceedings of IPTPS'2006*.

[11] K. Gummadi, H. Madhyastha, S. Gribble, H. Levy, and D. Wetherall. Improving the reliability of internet paths with one-hop source routing. In *Proceedings of USENIX OSDI'04*.

[12] K. Gummadi, S. Saroiu, and S. Gribble. King: Estimating latency between arbitrary internet end hosts. In *Proceedings of ACM IMW'02*.

[13] L. Guo, S. Jiang, L. Xiao, and X. Zhang. Fast and low cost search schemes by exploiting localities in p2p networks. *J. Parallel Distrib. Comput.*, 65(6), 2005.

[14] B. Krishnamurthy and J. Wang. On network-aware clustering of web clients. In *Proceedings of ACM SIGCOMM'00*.

[15] Y. Liang, E. Steinbach, and B. Girod. Real-time voice communication over the internet using packet path diversity. In *Proceedings of ACM Multimedia'01*.

[16] Z. Mao, L. Qiu, J. Wang, and Y. Zhang. On as-level path inference. In *Proceedings of ACM SIGMETRICS'05*.

[17] A. Markopoulou, F. Tobagi, and M. Karam. Assessing the quality of voice communications over internet backbones. *IEEE/ACM Trans. Netw.*, 11(5), 2003.

[18] A. Nakao, L. Peterson, and A. Bavier. A routing underlay for overlay networks. In *Proceedings of ACM SIGCOMM'03*.

[19] T. Nguyen and A. Zakhor. Path diversity with forward error correction (pdf) system for packet switched networks. In *Proceedings of IEEE INFOCOM'03*.

[20] S. Tao, K. Xu, A. Estepa, T. Fei, L. Gao, R. Guerin, J. Kurose, D. Towsley, and Z. Zhang. Improving voip quality through path switching. In *Proceedings of IEEE INFOCOM'05*.