A Method to Efficiently Integrate Internet Telephony Call Signaling with Dynamic Resource Negotiation

Yu GE, Winston SEAH

Networking Department, Communications and Devices Division, Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613
Email: {geyu, winston}@i2r.a-star.edu.sg

Abstract

Internet telephony imposes Quality of Service (QoS) requirements in terms of delay, jitter and loss ratio to provide reliable, high-quality multimedia services. While the Internet has historically offered a best effort service, it is hard to satisfy the required QoS of multimedia applications. In order to provide a tight QoS guarantee to telephony services, priority must be offered to real-time flows. Dynamic resource negotiations are required at the beginning of sessions besides long term service contracts. Most of the researchers propose to integrate call signaling with resource negotiation in a sequential manner [1–3]. This method can provide QoS at the highest level, but it results in high call setup delay and high probability of connection failure. Parallel integration is an alternative method that can mitigate these drawbacks. In this paper, we proposed a detail solution based on the parallel approach to establish resource negotiation simultaneously with call signaling. This approach can apply to different QoS models and mechanisms such as Integrated Services (IntServ), Differentiated Services (DiffServ) and Multi-protocol Label Switching (MPLS). This paper presents the architecture and system design of the parallel signaling approach. Threshold quality control scheme is applied to guarantee that the service is above the acceptable level before a telephone session begins.

Key words: SIP; QoS; RSVP; Internet telephony; call setup delay

1 Introduction

Internet Telephony is the transmission of real-time, two-way, interactive multimedia traffic over IP-based networks, conveying both the user data and control information over the Internet [4]. Currently, there are two call control standards for Internet Telephony: the International Telecommunication Union (ITU) Recommendation H.323 [5] and the Internet Engineering Task Force (IETF) standard Session Initiation Protocol (SIP) [6]. We choose SIP in our work because SIP is widely accepted by the Internet society [1] and 3GPP reported lower production and operation cost with SIP. As a call control protocol, SIP creates, modifies and terminates sessions between Internet end systems, as well as controls access to telephony specific services.

The best effort service of the current Internet has no guarantee to transmission timeliness. In Internet Telephony services, QoS is essential in two phases: call setup phase and real-time media exchange phase. In the telephony application, end-to-end audio transmission delays must be small enough so as not to interfere with normal voice conversations. Resource Reservation Protocol (RSVP) is a widely accepted IETF standard to do resource negotiation between end users and network components [7], to make sure that enough transport resources are provided to deliver multimedia flows promptly. To make IP Telephony services acceptable to users, the performance of IP Telephony signaling must be comparable to that of the current Switched Circuit Networks (SCNs) [8]. Call Setup Delay (CSD), also called post-dial delay, is a key and easily discernible QoS parameter [9], which is defined as the interval between the user dialling the last digit and receiving positive confirmation from the network. An excessive delay may lead the caller to abandon the call.

In the generic Internet telephony solutions, call signaling and resource negotiation are handled by separate protocols. However, these two protocols need to be coordinated during call setup to ensure media quality. A number of studies on the integration of call signaling with resource negotiation can be found in the literature [1–3].

[1] Cisco IP Phone 7960 and BTS 10200 Softswitch, eStara Softphone and Pingtel SIPfoundry products, etc.

Most of them propose to combine SIP with RSVP in a sequential manner. In this method, the subscribers first set up a call connection, then make the resource negotiation. Only if the resources in both transmission directions are available for the session, the subscribers initiate another signaling phase to activate the call connection, such as alerting the users and opening sound device ports. This method can provide QoS at the highest level, but it may result in high call setup delay and high probability of connection failure. Parallel integration of call signaling with resource negotiation is an alternative method that can mitigate these drawbacks. Although [3] suggested that the parallel integration can also be possible, it does not give any solution and detail analysis based on it. In this paper, we proposed a detail solution based on the parallel approach to negotiate resources together with call signaling. This paper presents the principle and architecture of the parallel signaling approach. We also apply threshold quality control scheme to guarantee that the service is above the acceptable level before a telephone session begins. The remainder of the paper is organized as follows. Section 2 outlines related work. In Section 3, we describe the architecture and system design of our new approach. Section 4 evaluates the performance of new approach using mathematical analysis and simulation. Finally, Section 5 concludes the paper.

2 Related Work

Most schemes for the integration of call signaling with resource negotiation make SIP and RSVP work sequentially. As a call signaling protocol in Internet Telephony, SIP has these functional entities: User Agent (UA), proxy server, redirect server and registrar. SIP defines a series of methods: INVITE, ACK, BYE, OPTIONS, CANCEL and REGISTER to process calls. A UA initiates a session with an INVITE request, which may traverse several servers, to the called party. A redirect server returns the current location of the callee, and the caller contacts the callee directly. Instead of returning the location information of the callee, a proxy server relays the INVITE request to the corresponding party on behalf of the caller. To terminate the call, either of the call parties issues a BYE message.

Resource Reservation Protocol (RSVP) is a proposed IETF standard for requesting and negotiating resource allocations between end users and networks or among network components. RSVP is broadly accepted in different QoS mechanisms, such as IntServ, DiffServ and MPLS. Also, many commercial routers provide support to RSVP [2] . As a signaling protocol, RSVP has different functionalities in different QoS mechanisms. In IntServ, RSVP is used for specifying resource requirements of real-time flows. A sender that wishes to initiate a session issues a PATH message to the corresponding receiver, containing traffic parameters and QoS requirement of the sending application. The receiver then generates a RESV message to request the resources in each node along the path. The intermediate nodes may accept or reject the request when receiving RESV. If the sender successfully receives the RESV message, meaning that the end-to-end resources have been reserved for the flow, the sender starts to transfer data. In DiffServ, the Expedited Forwarding (EF) service is suitable for Internet Telephony and videoconferencing. However, EF services are expensive and ISPs are more likely to support dynamic Service Level Agreements (SLAs), which allow customers to request EF services on demand without subscribing to them. In the dynamic SLA negotiation, it is the sender who requests resources through RSVP messages instead of the receiver in IntServ. Admission control is needed in each domain from the source to the destination during the process of the dynamic SLAs. When a PATH message reaches a domain, the Bandwidth Broker (BB) representing this domain processes this message, accepts or rejects the request and forwards the PATH message to the receiver. When receiving RESV message from the receiver, domain BBs configure domain components and allocate resources accordingly. In MPLS, an RSVP extended standard - RSVP-TE [10] is used to setup dedicated paths for flows. A multimedia flow can be assigned a specific prioritized Label Switching Path, through which such packets will be fast switched. In this paper, we exemplify the parallel approach with DiffServ QoS mechanism for simplicity. In fact, this approach can apply to different QoS models.

The sequential approach of integrating call signaling with resource negotiation includes two phases in call setup. Phase one consists of call signaling that do not alert the users and bi-directional end-to-end resource negotiation. The second phase involves the alerting of the user after the network resources are available for the session. In this case, the phone does not ring, until the required resources confirm to be available [1, 3]. The signaling flow of the sequential approach is depicted in Fig. 1. The sequential approach requires minor modifications to classical SIP, mainly the extension of not alerting the users upon receiving the first INVITE. However, it substantially increases the number of the messages exchanged for the call setup. The Call Setup Delay (CSD) will be significantly longer, including two stages of call signaling and bi-directional resource negotiations, especially when the RSVP messages are lost, and the default refresh timer for both messages is 30s [7, pp. 57–58]. In this case, the call setup delay will be totally different from that in the existing telephony service, which is about 3 to 5 second with Signaling System No.7 (SS7) [11, pp. 167]. In addition, large quantity of call requests may be blocked in a congested network, making

[2] There are some listed commercial routers supporting RSVP such as Cisco IOS Release 12.0 Quality of Service Solutions, Intel Express 8100 Series Routers, Foundry Networks NetIron Metro Routers, NEC IP Switch Core Router CX5220 and Redback Networks SmartEdge Routers, etc.

subscribers impatient.

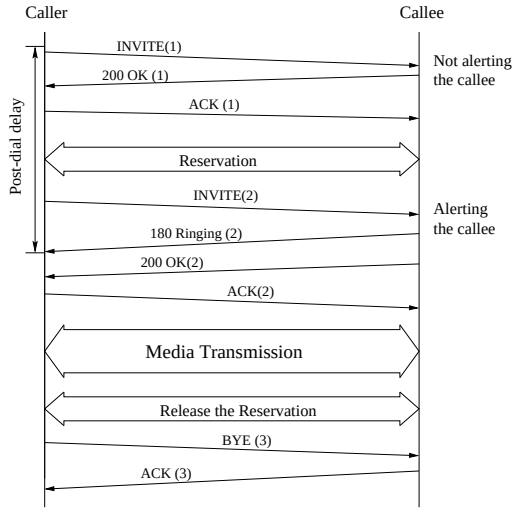


Fig. 1. Integration of Call Signaling with Resource Negotiation in Sequential Approach

3 An Efficient Parallel Integration Approach

This approach aims to reduce CSD to make it comparable to the existing telephony system. To achieve this target, the number of exchanged messages must be kept to a minimum and the signaling process must be simplified, while meeting the quality requirement of telephone users. In the SIP operation, since SIP messages may traverse multiple proxies, the exact route of SIP messages is likely to be different from the route that the real-time packets may take. However, the two kinds of messages are forwarded from the same source domain to the same destination domain, transiting the same intermediate domains, based on existing underlying routing mechanisms. Call signaling messages potentially can carry resource negotiation information in the domain-based resource allocations. To minimize the number of exchanged messages and simplify the process, it is possible to merge the information of call signaling and dynamic resource negotiations in some way. In the following sections, we describe our proposed architecture, in which the resource request information can be encapsulated in call signaling messages.

3.1 Architecture and System Design of the New Approach

The architecture of the new approach is depicted in Fig. 2. In this approach, a domain Server Group (SG) provides advanced services such as billing, resource management, presence and authentication. The server groups are similar to policy servers in COPS (Common Open Policy Service) [12], except that they use different protocols to communicate between policy clients

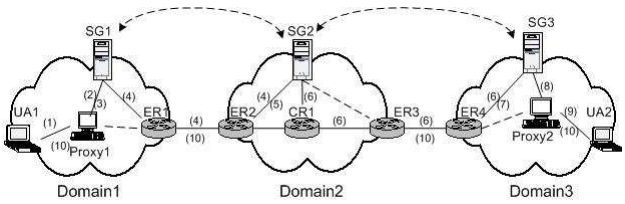


Fig. 2. The Architecture of New Approach

and servers. An SG communicates with the domain SIP proxy server through SIP messages, while a COPS policy server uses COPS protocol to talk to its policy client. In our architecture, we use the server group instead of COPS policy server because the system can have flexible implementation and be compatible with the existing network configurations. The COPS standard assumes that at least one policy server exists in each controlled administrative domain; other policy based management systems also adopt such a configuration [13, 14]. Therefore, it is valid for us to assume that each administrative domain is equipped with at least one server group. In this architecture, as call signaling will go from caller domain to callee domain depending on specific routing protocols, and media packets are also forwarded through the same sequence of domains. Thus, resource negotiation information can be encapsulated in call signaling messages to minimize the number of exchanged messages. In the operation of the new approach, it is necessary to identify the call signaling messages which encapsulate reservation information, by setting the "protocol type" field in the IP header to a specific number. The general packet format is shown in Fig. 3.

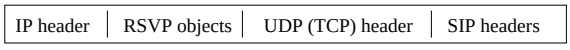


Fig. 3. The New Packet Format

A typical call setup sequence in the parallel approach is shown in Fig. 2. In this example, a user UA1 in Domain1 wishes to make a call to a user UA2 in Domain3. UA1 requests EF service in this call through dynamic SLA.

Session Initiation Phase:

(1) UA1 sends a call request, encapsulating resource request information (RSVP PATH objects) from the UA1 to UA2, to its local proxy server Proxy1.
(2) After Proxy1 gets the IP address of remote SIP server Proxy2 from Domain Name System (DNS), Proxy1 knows the destination domain subnet. Proxy1 then forwards the call signaling requests to the domain server group SG1. SG1 authenticates whether the user can access the resources.
(3) SG1 informs Proxy1 the authentication result. If the user is valid, Proxy1 sends "100 Trying" message to UA1; otherwise, sends "Request Failure 4xx" message to stop the session.

(4) SG1 starts to negotiate resources for this session with next domain server group SG2 with the call request message.
(5) SG2 checks if the Domain2 does not have enough resource to serve the session, SG2 denies the request. This reject information sends to SG1, either using RSVP messages or proprietary protocols. SG1 signals request failure using SIP "606 Not Acceptable" message, which is forwarded by Proxy1 to UA1 to stop the session.
(6) If Domain2 has enough resource, SG2 forwards the request to destination domain server group SG3. If there are more than one intermediate domains, the process repeats. SG3 checks the resources availability of Domain3.
(7) If there is not enough resource, SG3 rejects the request. The reject procedure is the same as SG2 reject procedure.
(8) If enough resource can be provided to serve the session, SG3 forwards the call signaling message to SIP proxy server Proxy2 in Domain3.
(9) Proxy2 then forwards the call request to callee UA2.
(10) If UA2 is available to receive the call, a "180 Ringing" message is sent back through the reverse path, carrying the resource request information from UA2 to UA1 for SG3, SG2 and SG1 to process.
(11) If UA1 receives the "180 Ringing" message, the admission controls are successful in both directions, and UA1 sends an acknowledgement (PRACK) to alert the callee.
(12) Both UA1 and UA2 then send RESV messages for SGs to configure the classification and policing rules on their domain routers. The interval between the callee hearing the ring and picking up the phone is usually long enough for exchanging bi-directional RSVP RESV messages [15], thus resource negotiations can be made in this period.
(13) When callee picks up the call, a "200 OK" messages is sent to UA1; UA1 acknowledges and completes the call signaling. Fig. 4 depicts the message flow of the new proposal.

When the UAC of the caller creates a call invitation, it asks for reliable delivery of provisional response (180 Ringing) for this invitation through inserting a Require header field, in order to ensure the bi-directional resources are available for this session [16]. After the UAC receives the 180 response message, it acknowledges the UAS with a PRACK message to trigger the ringing of the called party.

Session Modification/Refreshment Phase:
RSVP-capable components maintain soft states in the resource management. If a state times out without refreshment, the resource will not be used for the session and released to others. During the call, there are three actions (events) in end systems related to the renegotiation of resources:

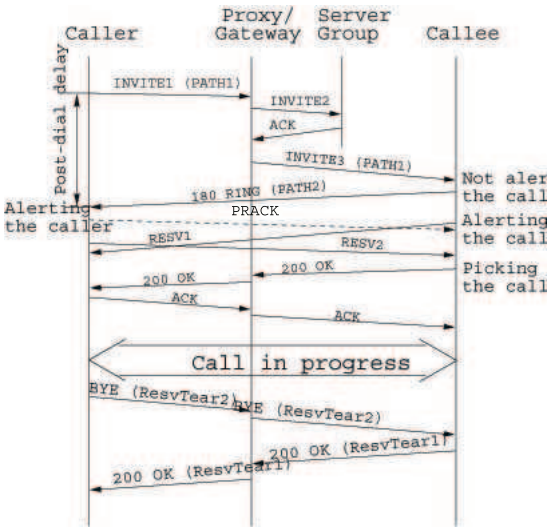


Fig. 4. Message Flow of The Integration in Parallel Way

- Resource Timeout. In this event, the RSVP soft state times out; media format and participants remain unchanged. RSVP PATH messages are needed to refresh the resource request.
- Media Modified. This event happens when any UA changes its media format and/or adds/cancels participants during active sessions, invoked by an SIP re-INVITE/UPDATE [17] message. The re-INVITE/UPDATE message encapsulates RSVP PATH information to refresh the resource request.
- Session Terminated. A session is terminated when either party sends a BYE message. UAs may choose to explicitly end the usage of resources through encapsulating ResvTear information into BYE messages, as shown in Fig. 4, or implicitly terminate when timeout.

Fig. 5 depicts the functional modules in SIP UA and proxy. In the host running the SIP client, the control plane contains the SIP application process and RSVP resource negotiation process. When a UA initiates a call, the SIP application process signals the RSVP process the resource requirement of the SIP session. The RSVP process prepares RSVP objects according to the application profile and delivers the objects to the SIP process. The inter-process signals can be proprietary and self-defined. The SIP process sends SIP messages and RSVP objects to the application-oriented transmission module, which composes control packets using the format as shown in Fig. 3. Then the packets are delivered to traffic control module to be transmitted to the network. It can be seen from Fig. 5 that different types of messages such as SIP, RSVP and multimedia (e.g., RTP) are handled by different modules before they arrive at the traffic control module to be transmitted into network.

As shown in Fig. 5, SIP proxy has a logical interface connected to SIP UA and a logical interface to the domain SG to signal resource requests and authentication

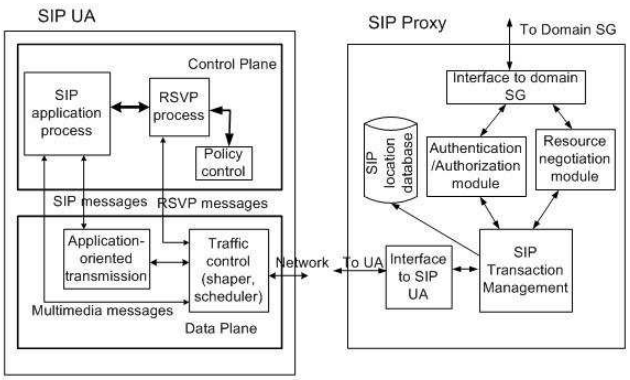


Fig. 5. SIP Components Functional Modules

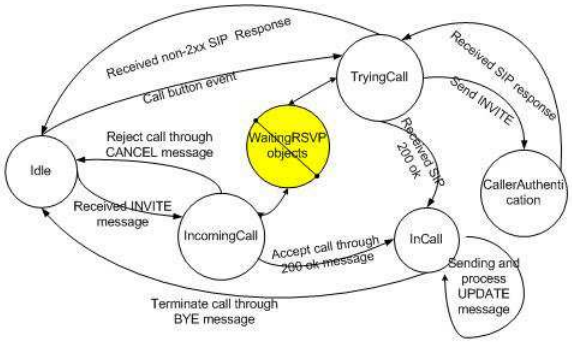


Fig. 6. The State Machine of UA in Parallel Scheme

information. The core component is the SIP transaction management module, which maintains SIP session transactions. Each SIP proxy transaction involves sending resource requests to the domain SG for the SG to negotiate between end-to-end domains, and feeding back request results to users.

In the parallel proposal, the state machine of SIP UAs retains most of the design of the classical SIP application process, adding only the "WaitingRSVPObjects" state as shown in Fig. 6. The "WaitingRSVPObjects" state transits from "IncomingCall" or "TryingCall" state, when the SIP process needs RSVP objects generated by RSVP process for further processing. When an SIP UA is in the "WaitingRSVPObjects state, there are inter-process communications between the SIP and RSVP processes. In Fig. 6, the line across the "WaitingRSVPObjects" state circle indicates that when the procedure is complete, the state can only transit back to its previous state and cannot cross the "WaitingRSVPObjects" state to a new state.

The functional modules of the domain Server Groups (SGs) are depicted in Fig. 7. An SG has an interface to adjacent domain SGs in order to negotiate inter-domain resource allocations. The interface to intra-domain routers manages to collect resource utilization information and send traffic control information to

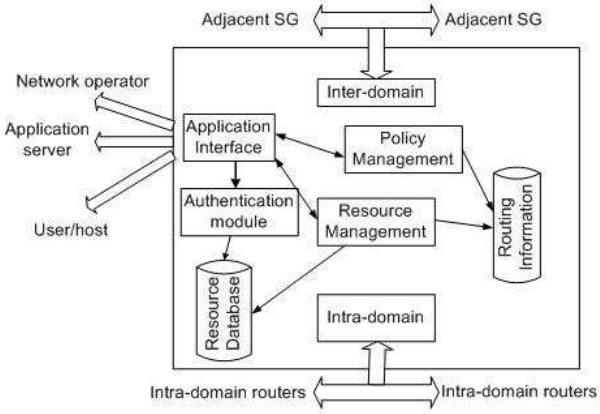


Fig. 7. Domain Server Group Functional Modules

these routers. The application interface is designed to get requests or instructions from operators, application servers and individual users. In our application scenario, the application interface in caller domain receives dynamic resource request from SIP components, refers to authentication and management modules, and negotiates resource availability through the inter-domain interface. When a transit domain SG receives a resource request from an adjacent domain, this SG decides to accept or reject the request according to resource availability of its domain and service contracts between domains. If a request is accepted, an SG configures its intra-domain routers to serve the requesting flows accordingly through the intra-domain interface. In the SG transactions, the message format follows Fig. 3, and the SGs only encode/decode and process RSVP objects.

3.2 Call Defect in Internet Telephony

Internet telephony subscribers do expect that the connection can be made once the destination phone rings. Blocking a call after ringing the destination is considered a "call defect", which is a critical problem in telephony services. In Fig. 4, when the callee is alerted, admission control in both directions are already successful. If the callee picks up the call, EF services are offered by all transit domains to this session. However, if admission control fails in either direction, the callee will not be alerted and the session terminates with the reason "resource not available"; thus, call defect never happens in this approach.

3.3 User Mobility Handling

If the callee has moved when the INVITE arrives at his home domain, the signaling path may differ from that of the real-time data traffic. The former may be triangular, passing by the home domain, while the latter will go directly from the caller to the callee. We use SIP personal mobility support scheme to merge the two paths.

As described in [6], each time the user moves, it registers its current location with servers in the home and foreign domains. When the INVITE arrives at his home domain, the server in the home domain operates in the "redirect mode", allowing the caller to contact the callee directly. The scheme is similar to route optimization in Mobile IP [18] and makes it highly scalable because the home domain server does not have to participate in the whole transaction. Fig. 8 depicts the operation. We have proposed a DNS based scheme to support personal mobility in Internet Telephony, which can also be used in the parallel approach [19].

3.4 Threshold Quality Control in New Approach

We propose the use of the Threshold Quality Control Scheme (TQCS) because to general customers a successful connection is more important than obtaining the best service quality. To these customers, a blocked call is more annoying than a relatively poorer quality call and they may also be willing to proceed with a lower quality service. For the same reason, a low bit rate codec is developed to enable graceful speech quality degradation in the case of lost frames [20]. In this context, a threshold quality control scheme is proposed in our approach.

In this scheme, there is a lower limit rate, which maps to the *threshold quality* that user selects in User Interface (UI), and an upper limit rate, which maps to telephone user's *preferred quality* indicated in UI. When the user initiates a call, the SIP application requests network resources with the lower limit rate that meets the user's *threshold quality*. If the session is successfully set up, the application probes more resources for the session through mid-point sampling algorithm.

We use an example to describe TQCS. In this example, a user runs an SIP application to initiate a videoconferencing and the user's system supports MPEG-2, a media compression scheme standardized as ISO Recommendation 13818. MPEG-2 has scalability options where users with different requirements can access the same video, without having multiple encoded copies of

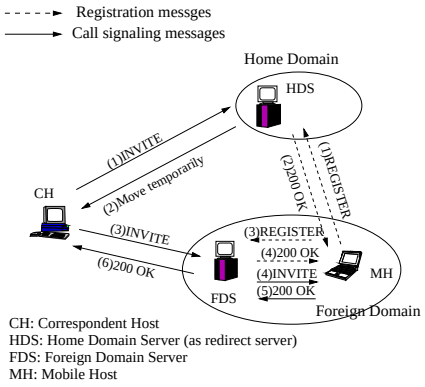


Fig. 8. The Registration and Mobility Operation of SIP

the same video. MPEG-2 media stream is composed of a base stream and enhanced stream. The SIP application provides an option to let the user select from an array of possible levels of media quality, each has a different enhanced stream; the total corresponding bit rates are: 4 Mbps, 20 Mbps, 80 Mbps, and 100 Mbps. This information can be wrapped and mapped to higher layer description such as "Basic Quality", "Medium Quality", "High Quality" and "Premium Quality" in radio buttons, even a slider to show different quality level, to be displayed in the application UI. In such an SIP UI, the user indicates the service level that he/she prefers to use, and also the threshold quality he/she accepts. These two values are mapped to the upper and lower limit in our algorithm.

When the sender issues an INVITE1(PATH1) message, the INVITE1 contains items of the Session Description Protocol (SDP) [21], which facilitates the advertisement of sessions and communicates the relevant session setup information to prospective participants, thus to support the negotiation of session content or media encoding. SDP includes the type of media (video, audio, etc.), the transport protocol (RTP/UDP/IP, H.320, etc.) and the format of the media (H.261 video, MPEG video, etc.). The proposed bandwidth to be used by the session or media is also specified.

After the user indicates his/her preference and threshold quality through UI, the SIP UA requests the resource that meets the threshold quality in the call signaling. If the domain SG collectively determines that unidirectional resources are insufficient to provide the threshold quality, call signaling procedure is blocked and error messages are returned to the caller. Otherwise, the called party receives the call signaling message and issues a 180 Ring(PATH2) message, also requesting end-to-end resources matching threshold quality. If the 180 RING(PATH2) reaches the caller, the threshold quality requirement is met for the session. Since the threshold quality is based on user's configuration in the UI, the call can be processed at an acceptable level to the user. When a session is successfully set up, each UA tries to probe for more resources to be used in this session using the mid-point sampling algorithm. In this algorithm, the UA requests the resources that meet the user's preference (upper limit rate). If such request succeeds, the session will use the user's preferred data rate; if it fails, the UA probes the mid-point of the lower and upper limit. If this fails again, resource request goes to the mid-point in the lower part. However, if it succeeds, the request goes to the middle point in the upper part, until four requests for different rates are reached during the call. Considering the tradeoff between probe granularity and overhead, the iteration limit is set as four to provide eight different layer coding. The current granularity of different media layer coding is below six. Therefore, the granularity provided by four iterations of probe should be enough to cater for the media adaptability granularity.

In the example of MPEG-2 with four different layer coding, three iterations are enough. In our previous example, MPEG-2 supports different bit rates of 4 Mbps, 20 Mbps, 80 Mbps, and 100 Mbps, and a user prefers 100 Mbps but accepts 4 Mbps transmission rate. If the call is connected, 4 Mbps data rate is guaranteed. The UA then requests the higher limit rate of 100 Mbps. If it succeeds, the session uses 100 Mbps enhanced transmission; if it fails, the UA probes for 52 Mbps. If succeeds, the UA tries to probe again until four iterations are reached. If the network can only guarantee 52 Mbps finally, the UA encodes the media up to 80 Mbps (one of the data rate of MPEG-2 layer coding). In this session, 52 Mbps of media stream is marked as *in-profile* and provided guaranteed transmission, which includes base stream and partial enhanced stream, while the remaining 28 Mbps (80 Mbps - 52 Mbps) of enhanced stream is marked as *out-profile* to be handled with lower priority by the network. Fig. 9 illustrates possible bandwidth probe results in the example of MPEG-2, where the shadowed blocks are MPEG-2 layer coding data rates and the light blocks are possible probed data rates in four iterations; the numbers insider blocks indicate data rates and the number above each blocks are their iteration numbers. For example, the block with number "3" above and with "76" inside means that in iteration 3 one possible data rate is 76 Mbps.

As per Fig. 5, when applying TQCS to the system, the SIP application process initiates RSVP module to probe for more resources. If higher transmission rate can be used, the SIP process sends UPDATE message to its peer UA process to modify the media parameter to achieve better quality. The existing dialog state will not be changed, as shown in Fig. 6.

Applying the TQCS and mid-point sampling algorithm, users have high probability of session connection and the network has higher resource utilization, if the system supports media with scalability option. In the service providers' point of view, these schemes are likely to provide better link utilization in the network. However, there is also a higher load in the network resource management system.

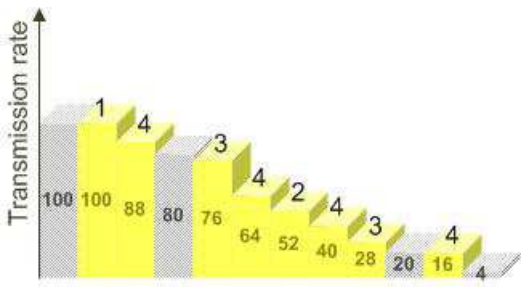


Fig. 9. An Example of Mid-point Sampling Algorithm

4 Performance Evaluation

4.1 Theoretical Analysis

An exact analysis of Call Setup Delay (CSD) appears to be very difficult due to the flexibility of SIP and underlying traffic control mechanisms. For example, SIP supports different transport protocols, packet retransmissions may be handled by TCP session or SIP transaction layer [6, pp. 91], resulting in different delays in retransmission. Traffic control policies applying to different control messages may be different in intermediate routers. Routers may give different priorities to different control packets (according to protocol ID). Furthermore, if policies are different domain-by-domain, the analysis is even more difficult. To make the analysis focused and tractable, we adopt the First-In-First-Out (FIFO) policy and use UDP as transport protocol to simplify the analysis, because it would be adequate to differentiate the performance between the parallel and sequential approaches. CSD includes the time to process all requests and responses in transit nodes and end systems. In the call setup period with QoS signaling, some inevitable delays would be incurred, e.g., service access latency, resource negotiation among domains and the propagation delay between end systems. However, some other delays can be reduced. The queueing delay will be longer when increasing the total number of exchanged messages during call setup. Our approach can reduce total queuing delay by decreasing the number of messages. We based the approximate analysis of the call setup performance on the network configuration shown in Fig. 2 and message flows shown in Figs. 1 and 4.

Our analysis includes the simple scenario, where the packet losses are not considered, and the scenario where the packet losses are considered.

4.1.1 Analysis without Packet Loss

We first define the notations used in the analysis. T_{quei} is the queuing delay of a single message in the i-th router. We assume every signaling packet experiences the same queueing delay in a specific router i. The assumption is valid because a signaling period is normally within a few seconds and the router utilization should not change much. T_{pthi} and T_{rsvi} are the transmission delays for the i-th router to forward RSVP PATH and RESV objects, respectively. T_{trsi} is the transmission time of a single SIP message in the i-th router. T_{pr} is the processing delay for a receiver to process PATH objects and generate RESV objects. T_{svc} is the service access time, which includes the delay experienced in the SIP proxy server and domain SGs to handle the service transactions. M is the average number of routers between two call parties. T_{cc} is the end-to-end propagation delay.

Let T_p be the CSD in the parallel approach; it can be

expressed as

$$T_p = 2\sum_{i=1}^{M}(T_{pthi} + T_{trsi} + T_{quei}) + T_{svc} + 2T_{cc} \quad (1)$$

In the parallel approach, there are two signaling messages incurring call setup delay. Both messages contain two parts: SIP headers and RSVP PATH objects. The delay of a signaling packet in a router i is $(T_{pthi} + T_{trsi} + T_{quei})$, containing data transmission delay of RSVP PATH objects and SIP headers, as well as queueing delay of the packet. The packets need to access the services in the SIP proxy and domain SG, incurring T_{svc}. There are two-way end-to-end propagation delays $(2T_{cc})$ between two call parties.

Let T_s be the CSD in the sequential approach, it can be expressed as

$$\begin{aligned} T_s &= 2\sum_{i=1}^{M}(T_{pthi} + T_{quei}) + 2\sum_{i=1}^{M}(T_{rsvi} + T_{quei}) \\ &\quad + 2T_{pr} + T_{svc} + 5\sum_{i=1}^{M}(T_{quei} + T_{trsi}) + 7T_{cc} \\ &= [2\sum_{i=1}^{M}(T_{pthi} + T_{trsi}) + T_{svc} + 2T_{cc}] \\ &\quad + 2\sum_{i=1}^{M} T_{rsvi} + 2T_{pr} + 9\sum_{i=1}^{M} T_{quei} \\ &\quad + 3\sum_{i=1}^{M} T_{trsi} + 5T_{cc} \end{aligned} \quad (2)$$

Taking the same computation method as the parallel approach, there are seven message exchanges during the signaling incurring call setup delay. In router i, an SIP message experiences delay as $(T_{quei} + T_{trsi})$; a RSVP PATH and RESV message's delays are $(T_{pthi}+T_{quei})$ and $(T_{rsvi} + T_{quei})$, respectively. The delay of T_{pr} is needed for an end host to process PATH objects and generate RESV objects. Also, the service time T_{svc} and propagation delay are contained in call setup delay.

Combining (1) with (2) yields

$$\begin{aligned} T_s &= T_p + 2\sum_{i=1}^{M} T_{rsvi} + 2T_{pr} + 7\sum_{i=1}^{M} T_{quei} \\ &\quad + 3\sum_{i=1}^{M} T_{trsi} + 5T_{cc} \end{aligned} \quad (3)$$

Dividing (3) by T_p, we obtain

$$\begin{aligned} \frac{T_s}{T_p} &= 1 + \frac{1}{T_p}(2\sum_{i=1}^{M} T_{rsvi} + 2T_{pr} + 7\sum_{i=1}^{M} T_{quei} \\ &\quad + 3\sum_{i=1}^{M} T_{trsi} + 5T_{cc}) \end{aligned} \quad (4)$$

The second term in (4) is always greater than zero; hence

$$\frac{T_s}{T_p} > 1 \quad (5)$$

From (5), we conclude that $T_s > T_p$ and therefore the parallel approach can reduce call setup delay as compared to the sequential approach.

4.1.2 Analysis with Packet Loss

During call setup, signaling messages are likely lost in congested networks. Therefore, it is necessary to theoretically analyze the delay in packet loss scenario. The CSDs under loss scenarios are analyzed in both parallel and sequential approaches. The proofs can be found in the appendix.

Let T_{com} be the compensation delay that is the packet retransmission time when a packet is lost, p_l be the packet loss probability in the networks, T_{ret1} be the default refresh timer for SIP messages and T_{ret2} be the default refresh timer for RSVP messages. Other definitions remain unchanged.

The mean of CSD in parallel approach can be written as

$$\tilde{T_p} = \sum_i p_i T_{pi} \quad (6)$$

where T_{pi} is the CSD when i packets are lost. In Fig. 4, the INVITE2 and ACK between Proxy and Server Group will be transmitted within the same domain, so we assume that they will not get lost. Therefore, the CSD only considers two packet losses. In the parallel approach, the compensation delay T_{com} includes the refresh timer value and retransmission delay of a packet, so its terms are the same as (1) except the timer value. T_{com} can be written as:

$$\begin{aligned} T_{com} &= \sum_{i=1}^{M}(T_{pthi} + T_{trsi} + T_{quei}) + T_{svc} + T_{cc} \\ &\quad + T_{ret1} \end{aligned} \quad (7)$$

The mean of CSD in parallel approach can be expressed as:

$$\begin{aligned} \tilde{T_p} &= T_p(1 - 2p_l + {p_l}^2) + (T_p + T_{com}) \\ &\quad (2p_l - 2{p_l}^2) + (T_p + 2T_{com}){p_l}^2 \\ &= T_p + 2p_l\sum_{i=1}^{M} T_{pthi} + 2p_l\sum_{i=1}^{M} T_{trsi} \\ &\quad + 2p_l\sum_{i=1}^{M} T_{quei} + 2p_l T_{svc} + 2p_l T_{cc} + 2p_l T_{ret1} \end{aligned} \quad (8)$$

where T_p is the CSD when no packet is lost.

The mean of CSD in the sequential approach $(\tilde{T_s})$ can be written as

$$\tilde{T_s} = \sum_i p_i T_{si} \quad (9)$$

where T_{si} is the CSD when i packets are lost. According to Fig. 1, the CSD only considers four SIP packet losses and four RSVP packet losses. Our analysis divides the losses into two parts, i.e., SIP losses and RSVP losses, then integrates them later.

The analysis of the sequential approach is similar to that of the parallel approach above. Let T_{ssi} (T_{sri}) be the

latency when i SIP (RSVP) packets are lost. (9) can be expressed as

$$\tilde{T_s} = \tilde{T_{ss}} + \tilde{T_{sr}} = \sum_i p_i T_{ssi} + \sum_i p_i T_{sri} \quad (10)$$

where $\tilde{T_{ss}}$ is the mean of T_{ssi} and $\tilde{T_{sr}}$ is the mean of T_{sri}.

The mean of CSD in the sequential approach is:

$$\begin{aligned}\tilde{T_s} &= \tilde{T_{ss}} + \tilde{T_{sr}} \\ &= T_p + (2+2p_l)\sum_{i=1}^{M} T_{rsvi} + 2T_{pr} + (5 \\ &\quad + 8p_l)\sum_{i=1}^{M} T_{quei} + (3+4p_l)\sum_{i=1}^{M} T_{trsi} \\ &\quad + (5+6p_l)T_{cc} + 4p_l T_{ret1} + 2p_l \sum_{i=1}^{M} T_{pthi} \\ &\quad + 2p_l T_{ret2} \end{aligned} \quad (11)$$

Comparing (8) with (11) yields

$$\begin{aligned}\tilde{T_s} - \tilde{T_p} = (2+2p_l)\sum_{i=1}^{M} T_{rsvi} + (5+6p_l) \\ \sum_{i=1}^{M} T_{quei} + (3+2p_l)\sum_{i=1}^{M} T_{trsi} \\ +(5+4p_l)T_{cc} + 2T_{pr} + 2p_l T_{ret1} \\ +2p_l T_{ret2} - 2p_l T_{svc}\end{aligned} \quad (12)$$

Dividing (12) by $(T_s - T_p)$ yields

$$\begin{aligned}\frac{\tilde{T_s}-\tilde{T_p}}{T_s-T_p} = 1 + \frac{1}{T_s-T_p}[2p_l \sum_{i=1}^{M} T_{rsvi} + 6p_l \sum_{i=1}^{M} T_{quei} \\ +2p_l \sum_{i=1}^{M} T_{trsi} + 4p_l T_{cc} \\ +2p_l(T_{ret1} + T_{ret2} - T_{svc})]\end{aligned} \quad (13)$$

where T_s and T_p are the CSDs in the sequential and parallel approaches when no packet is lost. The default refresh timer values for SIP messages (T_{ret1}) and RSVP messages (T_{ret2}) are 500ms [6, pp. 90] and 30s [7, pp. 57], respectively. Service access time (T_{svc}) nearly equals to the node latency in a server ($T_{quei} + T_{trsi}$). Therefore, the second term in (13) is always greater than zero; hence

$$\frac{\tilde{T_s}-\tilde{T_p}}{T_s-T_p} > 1 \quad (14)$$

From (14), we conclude that ($\tilde{T_s}$ - $\tilde{T_p}$) > (T_s - T_p). Therefore, the CSD differentiation between the parallel and sequential approaches in packet loss scenario is greater than that in non-loss scenario. The parallel approach has better performance in terms of CSD than the sequential approach in heavily loaded networks.

4.2 Simulation

We further verify the performance with simulation studies. The simulation tool used is Network Simulator - 2 (NS - 2, version 2.1b6) [22]. In our simulations, we study the establishing of call connections using the two approaches under different scenarios.

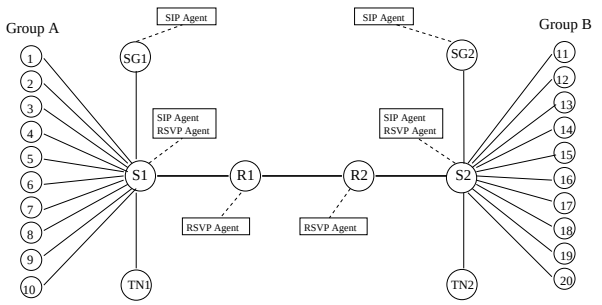
The purpose of this simulation study is to compare the performance of the parallel approach and sequential approach under a multi-user environment. The topology used in the simulations is shown in Fig. 10.

4.2.1 Simulation Scenario 1

To simulate the multi-user environment, we assume ten pairs of telephone users. The users (1-10) in group A try to make call connections to the users (11-20) in group B, using the parallel and sequential signaling approaches, separately. The ten users in group A send out the call invitations to group B users in turn, with the interval of 50 seconds. One user in group A and another user in group B comprise a call connection pair. After each call connection is successful, the user then begins to send out Constant Bit Rate (CBR) traffic with transmission rate of 64 kbps. We choose this rate because it is the typical real-time voice traffic rate in communication systems. In the simulation, we assume the call will last 1000 seconds.

In real world, there are other traffic passing through the shared link together with telephony traffic. In order to simulate call connection performance under multi-user and loaded environment, we use two traffic generator nodes (TN1, TN2), that issue and receive UDP packets as background traffic during the simulation period. The background traffic model is CBR, with a packet size of 500 bytes. To simulate different congestion levels in shared links, we adjust the packet sending interval from 0.012s to 0.004s, giving a background traffic rate that varies from 330 kbps to 1000 kbps. The bandwidths of the shared links are 1 Mbps each and the link capacity between the traffic nodes and the cross nodes (e.g., S1 and S2) is 10 Mbps each. In Fig. 10, the TN1 to S1 link and the TN2 to S2 link have bandwidths of 10 Mbps each. The two links have larger capacity because we need to make S1 and S2 perform packet scheduling and packet discarding functions under an overload condition.

Since the purpose of our proposal is to enhance the call



TN1, TN2: Traffic Node 1 & Traffic Node 2
S1, S2: SIP Server 1 & SIP Server 2
SG1, SG2: Server Group 1 & Server Group 2
R1, R2: Router 1 & Router 2
Link Bandwidth: 1Mb/s
Link Delay: 0.2s

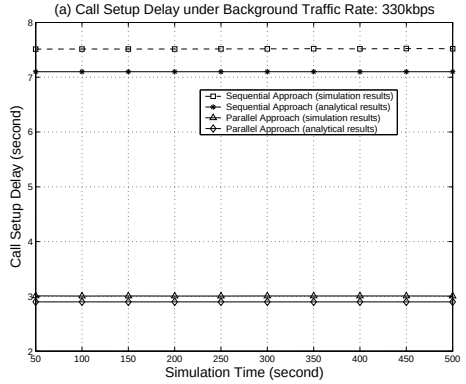
Fig. 10. Multi-user Simulation Scenario

signaling performance in QoS signaling, it is important to evaluate the voice quality after the call is connected by both the call parties. During the simulation, we trace the bandwidth utilization of each of the call flows and background traffic flow. The received bytes are recorded in each of the traffic sinks every 5s, i.e., a sampling window of 5s. Then we calculate the average rate in this 5 second period and assume this result to be the rate of this period. If a voice flow can be transferred in full bandwidth, it should have a 64 kbps transmission rate. The following experiments target at comparing call setup delays and voice qualities between the two schemes. We also provide the analytical results of the call setup delays based on the same network configurations as in the simulation. As shown in Figs. 11(a), 12(a) and 13(a), the analysis is accurate and provides the lower bound on call setup delay for both approaches.

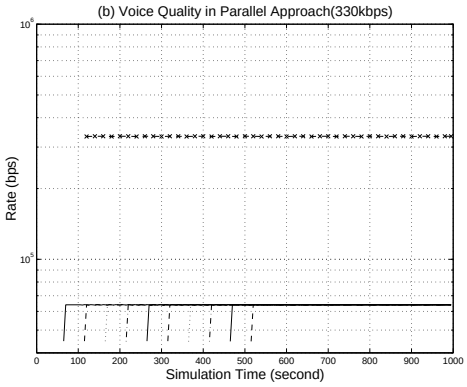
We first set Background Traffic Rate (BTR) at 330 kbps. Because the ten telephone calls will need a total bandwidth of 640 kbps to fully transfer their voice traffic and the link capacity is 1 Mbps, the total bandwidth utilization is 970 kbps, which is less than the link capacity. In this condition, the links provide the transmission rate according to the traffic requirement. Fig. 11(a) depicts the call setup delay comparison in the two approaches under this condition. Since the shared link is not saturated, all the flows transmit at their full rates. Fig. 11(b) and 11(c) depict the bandwidth utilization of voice flows under BTR of 330 kbps. In each of these two figures, the curve which is marked with "x" is the bandwidth utilization of background traffic and the remaining curves are the bandwidth utilization of the voice data flows. The other groups of figures that depict the bandwidth utilization of flows have the same identification method. From the Figs. 11(b) and 11(c), the bandwidth utilizations of both approaches are nearly the same. This result is obtained because under such network condition, the resource requests for both approaches can be acquired, thus the bandwidth utilization patterns are nearly the same.

Then, we increase the background traffic to 500 kbps. When the ten calls are in progress, total traffic in this link will be 1140 kbps, thus, causing congestion in the shared link. The shared links are not saturated until the ninth call begins because when the eighth call begins, the total bandwidth occupation is 1012 kbps. Call setup delays will increase from the ninth call connection. The results are given in Fig. 12(a). Because telephony flows have higher priority, the background traffic is degraded consequently. Fig. 12(b) and 12(c) illustrate the voice qualities under 500 kbps of BTR. Also, the similar utilization patterns from these two figures are obtained because resource requests in both approaches can be acquired under such network status.

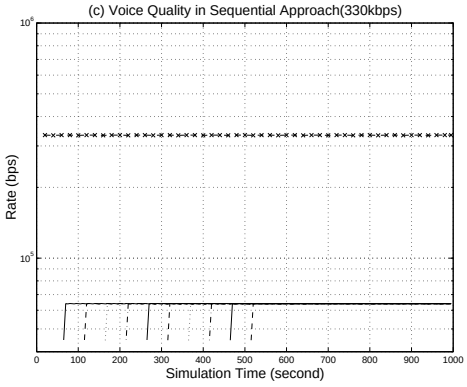
When BTR reaches 1000 kbps, it occupies the whole link, so the shared link is congested after the simulation starts.



(a) Call Setup Delay Comparison with Two Approaches

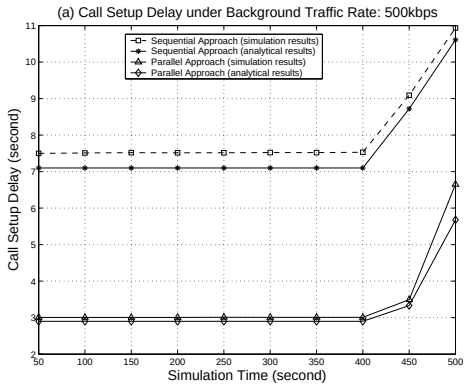


(b) Bandwidth Utilization under Parallel Approach

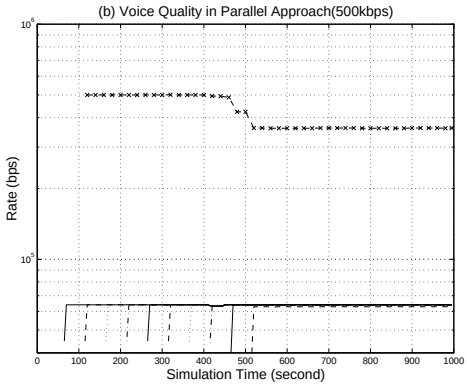


(c) Bandwidth Utilization under Sequential Approach

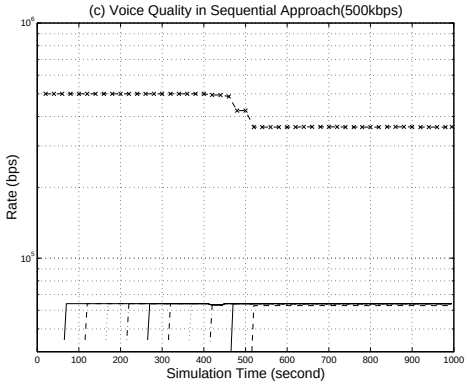
Fig. 11. Simulation under Background Traffic Rate: 330kbps



(a) Call Setup Delay Comparison with Two Approaches



(b) Bandwidth Utilization under Parallel Approach



(c) Bandwidth Utilization under Sequential Approach

Fig. 12. Simulation under Background Traffic Rate: 500kbps

In both the proposals, call setup delay has increased significantly. Fig. 13(a) depicts the call setup delay comparison of the two approaches. In the simulation, a call connection might fail due to message lost or resource unavailability. In the parallel approach, through using the TQCS, the failure of bandwidth request for the preferred media does not immediately result in the failure of this call connection. However, the unsuccessful resource request in the sequential approach will terminate the call connection immediately. Therefore, call connections using the parallel approach can be successful with a higher probability. From Fig. 13(b) and Fig. 13(c), we can see that four call connections fail in the sequential approach, but the successful connections can roughly transmit the real-time flows over the required bandwidths. While using the parallel signaling approach, the connection setup success ratio is higher with the trade-off of some call flows being transmitted at lower rates. This trade-off comparison in congested network can be illustrated in Table 1.

4.2.2 Simulation Scenario 2

Scenario 1 simulates call connections between different pairs of users through shared links. However, the interval between each two call requests is fixed (50s), and each call duration is fixed (1000s). Therefore, this scenario is different from the real telephone usage in communication systems. We now randomize our simulation to provide a better abstraction of real networks.

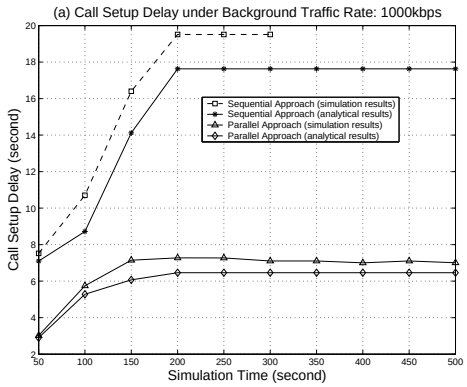
The topology and link parameters (delay, capacity) in simulation scenario 2 are the same as scenario 1, as shown in Fig. 10. In this simulation, each of the users in group A initiates a session to a randomly selected idle user from group B. Each session starts at a randomly selected time, which is uniformly distributed between 0s and 1000s, and lasts a randomly selected duration, which is uniformly distributed with an average value of 180s. After a session is completed, the idle caller in group A will initiate another call connection to a randomly selected idle user from group B. The intervals between the sessions initiated by the same caller are also randomly selected, which has a uniform distribution between 0s and 1200s. Fig. 14 shows the distribution of the sessions over the simulation period.

With per call bandwidth utilization remaining at 64 kbps, we begin with 700 kbps of BTR and no congestion happens under this condition. The call setup delay comparison can be seen in Fig. 15(a).

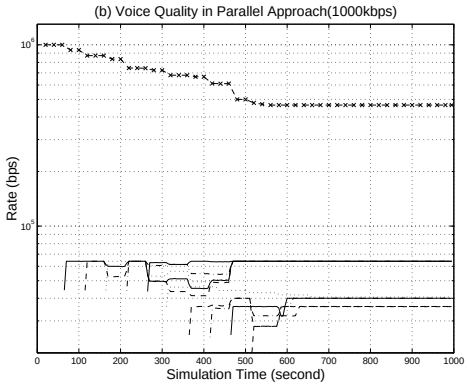
The BTR is then increased to 900 kbps. Congestion

Table 1
The Trade-off Comparison in the Two Approaches

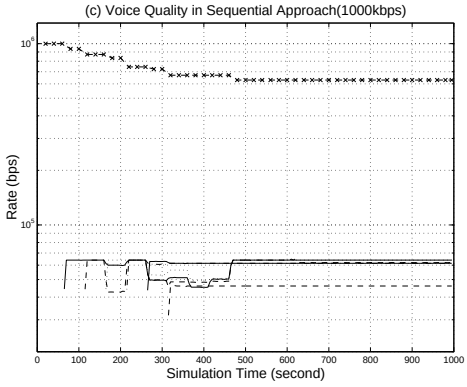
	Sequential	Parallel
Media transmission quality	Good	Acceptable
Connection rejected probability	High	Low



(a) Call Setup Delay Comparison with Two Approaches



(b) Bandwidth Utilization under Parallel Approach



(c) Bandwidth Utilization under Sequential Approach

Fig. 13. Simulation under Background Traffic Rate: 1000kbps

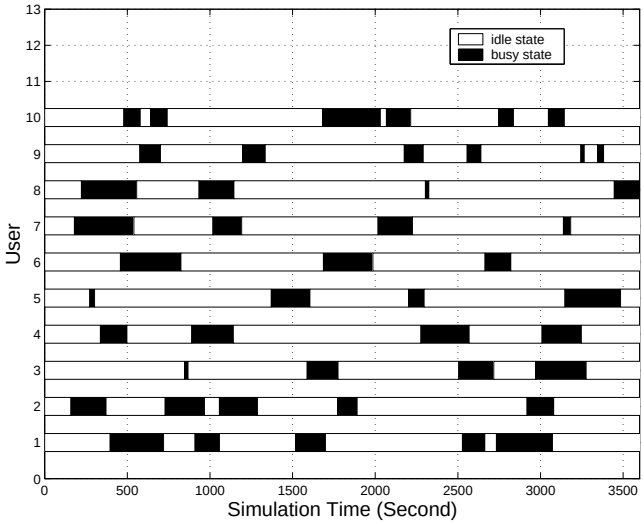


Fig. 14. The Distribution of The sessions in Scenario 4

occurs when two voice calls are active simultaneously. Since signaling messages are treated as best effort, call setup will incur longer delays in a congested condition. Fig. 15(b) shows the delay for the two approaches.

When BTR is raised to 1000 kbps, the shared links are saturated even when there is no voice call in progress. In this situation, most of the call connection setups take much longer time than in a lightly loaded condition. Fig. 15(c) depicts such results.

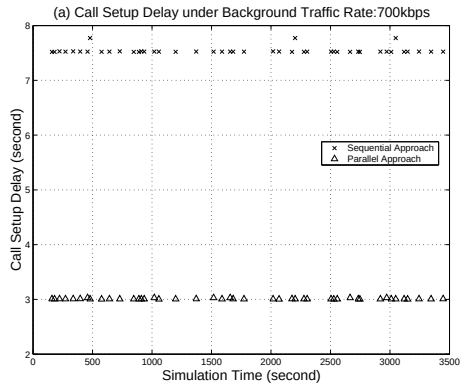
4.2.3 Overhead Reduction of Parallel Approach

Due to combination of RSVP information with SIP, there are some reductions of the packet overhead, which leads to the decrease of the link load. To verify this, we calculate the packet overheads in the two approaches. The SIP message example assumed in our study is taken from examples that were developed during the SIP interoperability test [23]. It represents an example of a minimum set of functionality for SIP to be used in IP Telephony applications. The RSVP information conforms to proposed IETF RSVP usage [24], expressed as RSVP OBJECTs.

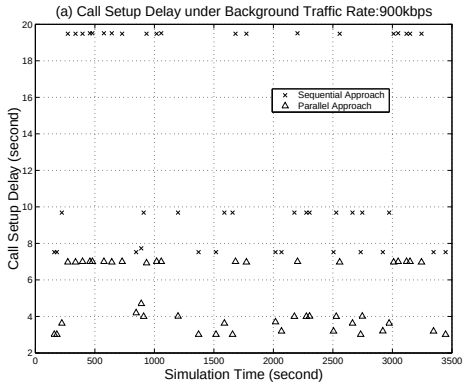
The comparison of the packet overheads in the two approaches is illustrated in Table 2 and shown in Fig. 16. The overhead reduction of the parallel approach is 21.46% in IPv4 and 21.24% in IPv6 in the typical message example.

5 Conclusion

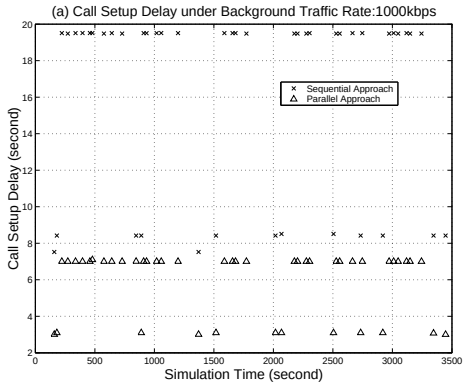
In this paper, we proposed a scheme that integrates dynamic resource negotiations with the Internet telephony call setup process to ensure that adequate resource is



(a) Call Setup Delay under BTR: 700 kbps



(b) Call Setup Delay under BTR: 900 kbps



(c) Call Setup Delay under BTR: 1000 kbps

Fig. 15. Simulation Results under Scenario 2

Table 2
The Comparison of the Packet Overheads in the Two Approaches

	Sequential Approach		Parallel Approach	
Messages (byte)	IPv4	IPv6	IPv4	IPv6
SIP messages	3600	3960	2000	2200
RSVP messages	352	560	144	248
Compound messages	0	0	960	1112
Total	3952	4520	3104	3560

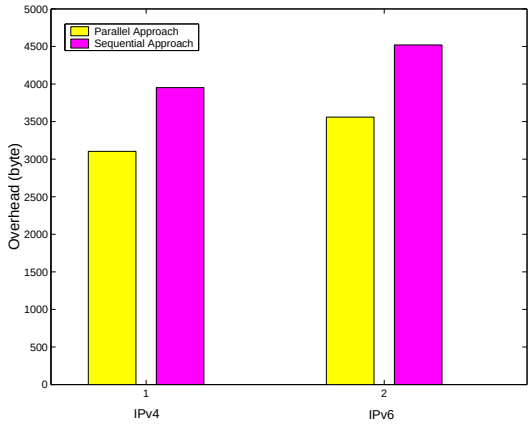


Fig. 16. The Packet Overheads of the Two Approaches

available for real-time traffic. Because the existing sequential approach would result in high call setup delay, we proposed a parallel approach to establish resource negotiation simultaneously with call signaling. Resource request information is encapsulated in call signaling messages to minimize the number of exchanged messages during call setup. Application-layer mobility support mechanisms are introduced in this approach. We also apply threshold quality control scheme to ensure that the service is above the acceptable level before session starts. Performance analysis and simulation results show that the parallel approach achieves better performance in terms of call setup delay as compared to the sequential approach.

References

[1] G. Camarillo, W. Marshall, and Jonathan Rosenberg. Integration of Resource Management and SIP, IETF RFC 3312, October 2002.

[2] Pawan Goyal et al. "Integration of call signaling and resource management for IP telephony". *IEEE Network*, 13(3):24 – 32, May/June 1999.

[3] Henning Schulzrinne, Jonathan Rosenberg, and Jonathan Lennox. "Interaction of Call Setup and Resource Reservation Protocols in Internet Telephony". Technical report, Columbia University and Bell Laboratories. http://www.cs.columbia.edu/hgs/sip/drafts/resource.pdf, June 1999.

[4] Henning Schulzrinne and Jonathan Rosenberg. "Internet Telephony: Architecture and Protocols - an IETF Perspective". *IEEE Computer Networks and ISDN system*, 31(3):237–255, February 1999.

[5] ITU-T. Packet-Based Multimedia Communications Systems, ITU-T Recommendation H.323 (07/03), July 2003.

[6] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, P. Sparks, et al. SIP: Session Initiation Protocol, IETF RFC 3261, June 2002.

[7] R. Braden et al. Resource ReSerVation Protocol(RSVP)-Version 1 Functional Specification, IETF RFC 2205, September 1997.

[8] Huai-An P. Lin et al. VoIP Signaling Performance Requirements and Expectations, Internet Draft, Internet Engineering Task Force, Work in progress, March 1999.

[9] Tony Eyers and Henning Schulzrinne. "Predicting Internet Telephony Call Setup Delay". In *Proceedings of First IP Telephony Workshop*, pages 107 – 126, Berlin, Germany, April 2000.

[10] D. Awduche, L. Berger, D. Gan, T. Li, et al. RSVP-TE: Extensions to RSVP for LSP Tunnels, IETF RFC 3209, December 2001.

[11] Walter J. Goralski and Matthew C. Kolon. *IP telephony*. McGraw–Hill, 2000.

[12] D. Durham Ed., J. Boyle, R. Cohen, S. Herzog, R. Rajan, and A. Sastry. The COPS (Common Open Policy Service) Protocol, IETF RFC 2748, January 2000.

[13] A. Ponnappan, Lingjia Yang, R. Pillai, and P. Braun. "Policy Based QoS Management System for the IntServ/DiffServ Based Internet". In *Proceedings of the Third International Workshop on Policies for Distributed Systems and Networks, 2002*, June 2002.

[14] P. Flegkas, P. Trimintzios, and G. Pavlou. "A policy-based quality of service management system for IP DiffServ networks". *IEEE Network Magazine, special issue on Policy Based Networking*, 16(2):50–56, March/April 2002.

[15] Anindya Neogi et al. "Performance Analysis of an RSVP-Capable Router". *IEEE Network*, pages 56–63, September/October 1999.

[16] J. Rosenberg and H. Schulzrinne. Reliability of Provisional Responses in the Session Initiation Protocol (SIP), IETF RFC 3262, June 2002.

[17] J. Rosenberg. The Session Initiation Protocol (SIP) UPDATE Method, IETF RFC 3311, September 2002.

[18] Charles E. Perkins. *Mobile IP: Design Principles and Practices*. Prentice Hall, October 1997.

[19] Yu GE, Winston SEAH, Seng-Kee TAN, Wang-Cho CHENG, Qi YAO, and Chee-Wei ANG. "Extending Domain Name Service to Support Internet Telephony Personal Mobility". In *Proceedings of the 59th IEEE Vehicular Technology Conference* , May 2004.

[20] S. V. Andersen et al. Internet Low Bit Rate Codec, Internet Draft, Internet Engineering Task Force, Work in progress, September 2002.

[21] M. Handley and V. Jacobson. SDP: Session Description Protocol, IETF RFC 2327, April 1998.

[22] The Network Simulator - ns-2 Webpage, http://www.isi.edu/nsnam/ns/.

[23] Alan Johnston, Steve Donovan, et al. SIP Call Flow Examples, Internet Draft, Internet Engineering Task Force, Work in progress, February 2002.

[24] J. Wroclawski. The use of RSVP with IETF Integrated Service, IETF RFC 2210, September 1997.

APPENDIX

A-1 Proof of CSD in parallel approach with packet loss

Proof. The mean of CSD in parallel approach can be written as

$$\tilde{T_p} = \sum_i p_i T_{pi} \tag{15}$$

where T_{pi} is the CSD when i packets are lost. In Fig. 4, the INVITE2 and ACK between Proxy and Server Group will be transmitted within the same domain, so we assume that they will not get lost. Therefore, the CSD only considers two packet losses. In the parallel approach, the compensation delay T_{com} includes the refresh timer value and retransmission delay of a packet, so its terms are the same as (1) except the timer value. T_{com} can be written as:

$$T_{com} = \sum_{i=1}^{M}(T_{pthi} + T_{trsi} + T_{quei}) + T_{svc} + T_{cc} + T_{ret1} \tag{16}$$

When no packet is lost, all messages are transmitted successfully and no retransmission happens. This is the best case where no compensation delay occurs. The CSD is expressed as

$$T_{p0} = T_p \tag{17}$$

where T_p is the CSD when no packet is lost. The probability of the best case is

$$p_0 = \binom{2}{2}(1-p_l)^2 = 1 - 2p_l + {p_l}^2 \tag{18}$$

When one packet is lost, there is one retransmission. In this case, either the packet from the call initiator or that from the callee gets lost; that is, no response is received from the other party within the time limit (refresh timer value). Therefore, it should retransmit that packet and the CSD (T_{p1}) has one unit of compensation delay base on the best case CSD (T_p),

$$T_{p1} = T_p + T_{com} \tag{19}$$

The probability of this case p_1 is written as

$$p_1 = \binom{2}{1} p_l(1-p_l) = 2p_l - 2{p_l}^2 \tag{20}$$

When two packets are lost, there are two retransmissions. In this case, the call initiator sends out the first packet and does not receive any response from the supposed callee when the refresh timer expires; it should retransmit that packet. Also the second signaling message is lost and the callee has to retransmit it. Therefore,

there are two units of compensation delay in the CSD (T_{p2}),

$$T_{p2} = T_p + 2T_{com} \tag{21}$$

The probability of this case p_2 can be written as

$$p_2 = \binom{2}{2} {p_l}^2 = {p_l}^2 \tag{22}$$

Inserting (17), (18), (19), (20), (21), (22) and (16) into (15), we obtain

$$\begin{aligned}\tilde{T_p} &= T_p(1 - 2p_l + {p_l}^2) + (T_p + T_{com}) \\ &\quad (2p_l - 2{p_l}^2) + (T_p + 2T_{com}){p_l}^2 \\ &= T_p + 2p_l \textstyle\sum_{i=1}^{M} T_{pthi} + 2p_l \sum_{i=1}^{M} T_{trsi} \\ &\quad + 2p_l \textstyle\sum_{i=1}^{M} T_{quei} + 2p_l T_{svc} + 2p_l T_{cc} + 2p_l T_{ret1}\end{aligned} \tag{23}$$

This ends the proof. □

A-2 Proof of CSD in sequential approach with packet loss

Proof. The mean of CSD in the sequential approach ($\tilde{T_s}$) can be written as

$$\tilde{T_s} = \textstyle\sum_i p_i T_{si} \tag{24}$$

where T_{si} is the CSD when i packets are lost. According to Fig. 1, the CSD only considers four SIP packet losses and four RSVP packet losses. Our analysis divides the losses into two parts, i.e., SIP losses and RSVP losses, then integrates them later.

The analysis of the sequential approach is similar to that of the parallel approach above. Let T_{ssi} (T_{sri}) be the latency when i SIP (RSVP) packets are lost. (24) can be expressed as

$$\tilde{T_s} = \tilde{T_{ss}} + \tilde{T_{sr}} = \textstyle\sum_i p_i T_{ssi} + \sum_i p_i T_{sri} \tag{25}$$

where $\tilde{T_{ss}}$ is the mean of T_{ssi} and $\tilde{T_{sr}}$ is the mean of T_{sri}.

When SIP messages are lost, T_{coms}, which represents the compensation delay for SIP messages, can be written as

$$T_{coms} = \textstyle\sum_{i=1}^{M}(T_{quei} + T_{trsi}) + T_{cc} + T_{ret1} \tag{26}$$

When there is no packet loss, let T_{ss} be SIP message latency which contributes to the CSD in the sequential approach. In this case,

$$T_{ss0} = T_{ss} \tag{27}$$

The probability p_0 can be written as

$$p_0 = \tbinom{4}{4}(1 - p_l)^4 \tag{28}$$

When one SIP packet is lost, there is one retransmission and the SIP message latency can be expressed as

$$T_{ss1} = T_{ss} + T_{coms} \tag{29}$$

The probability p_1 can be expressed as

$$p_1 = \tbinom{4}{1} p_l (1 - p_l)^3 = 4p_l - 12{p_l}^2 + 12{p_l}^3 - 4{p_l}^4 \tag{30}$$

When two SIP packets are lost, two retransmissions occur. Therefore, the SIP message latency will comprise two units of compensation delay and can be expressed as

$$T_{ss2} = T_{ss} + 2T_{coms} \tag{31}$$

The probability in this case is

$$p_2 = \tbinom{4}{2} {p_l}^2 (1 - p_l)^2 = 6{p_l}^2 - 12{p_l}^3 + 6{p_l}^4 \tag{32}$$

When three SIP packets are lost, they have to be retransmitted. This latency is

$$T_{ss3} = T_{ss} + 3T_{coms} \tag{33}$$

The probability is presented as

$$p_3 = \tbinom{4}{3} {p_l}^3 (1 - p_l) = 4{p_l}^3 - 4{p_l}^4 \tag{34}$$

When four SIP packets are lost, similarly, the SIP message latency can be written as

$$T_{ss4} = T_{ss} + 4T_{coms} \tag{35}$$

The probability is

$$p_4 = \tbinom{4}{4} {p_l}^4 = {p_l}^4 \tag{36}$$

Inserting (27), (28), (29), (30), (31), (32), (33), (34), (35), (36) and (26) into (25), we obtain the mean of T_{ssi} ($\tilde{T_{ss}}$)

$$\begin{aligned}\tilde{T_{ss}} &= p_0 T_{ss} + p_1(T_{ss} + T_{coms}) + p_2(T_{ss} + 2T_{coms}) \\ &\quad + p_3(T_{ss} + 3T_{coms}) + p_4(T_{ss} + 4T_{coms}) \\ &= T_{ss} + 4p_l[\textstyle\sum_{i=1}^{M}(T_{quei} + T_{trsi}) + T_{cc} \\ &\quad + T_{ret1}]\end{aligned} \tag{37}$$

When RSVP messages are lost, T_{comr}, which represents the compensation delay for RSVP messages, can be written as

$$\begin{aligned}T_{comr} &= 0.5 \textstyle\sum_{i=1}^{M}(T_{pthi} + T_{rsvi} + 2T_{quei}) + T_{cc} \\ &\quad + T_{ret2}\end{aligned} \tag{38}$$

The loss probability of PATH and RESV is equal, because the two messages are transmitted along the same routes. Therefore, we take the mean value of their latency to simplify the analysis.

Let T_{sr} be RSVP message latency in the sequential approach when there is no packet loss. Because the RSVP packet loss probability and delay algorithms are the same as those of SIP, we can use (27), (29), (31), (33) and (35) after substituting T_{sr} for T_{ss}, T_{sri} for T_{ssi} and T_{comr} for T_{coms}. Inserting the substituted equations as well as (38), we obtain the mean of T_{sri} ($\tilde{T_{sr}}$)

$$\begin{aligned}\tilde{T_{sr}} &= p_0 T_{sr} + p_1(T_{sr} + T_{comr}) + p_2(T_{sr} + 2T_{comr}) \\ &\quad + p_3(T_{sr} + 3T_{comr}) + p_4(T_{sr} + 4T_{comr}) \\ &= T_{sr} + 4p_l[0.5\sum_{i=1}^{M}(T_{pthi} + T_{rsvi} + 2T_{quei}) + T_{cc} \\ &\quad + T_{ret2}] \end{aligned} \tag{39}$$

Inserting (37), (39) and (3) into (25), we obtain the mean of CSD

$$\begin{aligned}\tilde{T_s} &= \tilde{T_{ss}} + \tilde{T_{sr}} \\ &= T_p + (2 + 2p_l)\sum_{i=1}^{M} T_{rsvi} + 2T_{pr} + (5 \\ &\quad + 8p_l)\sum_{i=1}^{M} T_{quei} + (3 + 4p_l)\sum_{i=1}^{M} T_{trsi} \\ &\quad + (5 + 6p_l)T_{cc} + 4p_l T_{ret1} + 2p_l \sum_{i=1}^{M} T_{pthi} \\ &\quad + 2p_l T_{ret2} \end{aligned} \tag{40}$$

This ends the proof. □