

Publishing Trickle ICE Capability Through Presence Information

Kiran Kumar Guduru*

* Lead Engineer, Samsung R&D Institute India – Bangalore, India
kiran.guduru@samsung.com

Abstract— Interactive Connectivity Establishment is the process of establishing a transport connection between two peers behind Network Address Translators. To reduce the session setup delay with the existing ICE process, the concept of Trickle ICE is introduced. With Trickle ICE, peers can communicate individual ICE candidates immediately after gathering. The User Agents supporting this Trickle ICE feature, and initiating a communication session, has no information whether the remote peer will support Trickle ICE or not, resulting in the lack of information for backward compatibility. Presence information data provides a complete view of the communication capabilities of a User Agent. This data can describe different aspects of the user agent's communication capabilities like characteristics of device, physical location, ability to take part in a communication session, state of the user etc. This research work proposes a solution to communicate the capability of one peer, towards the support of Trickle ICE, to the remote peer through presence information and analyzes the number of signaling messages required for establishing a session between two peers. This work also analyzes the amount of bandwidth utilized using SIP as example signaling protocol through descriptive analysis method.

Keywords— ICE, Trickle ICE, Presence, User Agent Capabilities, Peer to Peer Connection

I. INTRODUCTION

Internet usage is increasing very rapidly through desktops as well as mobile devices. With the increase in the usage of internet, the demand for Internet Protocol (IP) based communication applications is also increasing. This increased demand for IP based applications on mobile devices insists for reduced session establishment time, without compromising in session establishment success rate.

A logical end user entity is termed as a User Agent. The device that initiates the session is termed as User Agent Client (UAC) and that of the devices that receives the session is termed as User Agent Server (UAS).

User Agents, either UAC or UAS, may be behind the (Network Address Translator) NAT / Firewall. In this scenario, it is not possible to establish a transport connection between the two User Agents directly with their self / local IP addresses and port numbers. To establish a transport connection between two User Agents, that might be behind NAT, Internet Engineering Task Force (IETF) has standardized a method called Interactive Connectivity Establishment (ICE) [1]. A logical entity that comprises of the IP address, port number, and other required details of connection entity is termed as an ICE candidate or

simply candidate. For establishing a transport connection, ICE follows the process with the following steps

- Gathering ICE candidates
- Sorting of ICE candidates
- Exchanging of ICE candidates with the remote peer
- Pinging of ICE candidates in the order of their priorities
- Establishing the transport connection between the best pair of ICE candidates, identified as per their priorities

During this process User Agents needs to wait for a considerable amount of time, to complete the Gathering, exchanging and sorting of ICE candidates, at both the ends before starting the connectivity checks. This delay of connection establishment varies with factors like network latency, processing speed of the User Agent, Session Traversal Utilities for NAT (STUN) [2] and Traversal Using Relays around NAT (TURN) [3] server response times.

Connection establishment time delay with the existing system demands the need for reducing the connection establishment time. To reduce the connection establishment time, a process called Trickle ICE is proposed and being standardized by IETF. Trickle ICE proposes incremental provisioning of ICE candidates [4] to the remote peer immediately after gathering of each ICE candidate, instead of waiting for gathering all the ICE candidates.

Since Trickle ICE is an upcoming standard, all the User Agents might not support this technology. So provisioning of few ICE candidates instead of complete set of ICE candidates is specified, resulting in Half Trickle ICE [4]. This demands the need for the UAC to know whether the UAS supports Trickle ICE or not prior to initiating the session. Till now there is no means for communicating the capability for supporting the Trickle ICE feature to the remote peer before initiating a communication session. This initiates the need for identifying a solution for conveying the capability for supporting Trickle ICE feature to the remote User Agent. This research work proposes a solution to broadcast the support for Trickle ICE capability by a User Agent to all the User Agents in the buddy list through presence information.

The following section explains the existing methods being followed and related work to convey the support for Trickle ICE to the remote User Agents.

II. RELATED WORK

Trickle ICE process, intended for reducing the session establishment time is being standardized by IETF under Multiparty Multimedia Session Control (MMUSIC) working group. The drafted standard [4] is being adopted by IETF Web Real Time Communications working group (IETF RTCWEB). IETF RTCWEB working group along with World Wide Web Consortium (W3C) WebRTC working group defines standards for browser, making provision for Java Script application to interact with browsers to establish multimedia sessions between browsers or between browser and any other kind of User Agent other than browser through gateways [5, 6]. A new Multi-purpose Internet Mail Extension (MIME) type [7], “application/ice-candidates” has been proposed to add only ICE candidates to SIP message body in order to support Trickle ICE capability through SIP signaling “in press” [8].

IETF RTCWEB working group proposed a method for identifying the capability of the remote User Agent, to support incremental provisioning of ICE candidates. In that method initial SDP without ICE candidates will be negotiated along with the indication to convey about the capability of supporting Trickle ICE to the remote User Agent. Depending on this negotiation, subsequent offer / answers with ICE candidates will be negotiated.

This approach will give rise to following scenarios. In the first scenario both the User Agent applications will support Trickle ICE. In this scenario, the User Agent can negotiate with initial offer / answer before gathering of ICE candidates to know about the capability of remote User Agent to support incremental provisioning of ICE candidates, and then negotiate the individual ICE candidates immediately after they are gathered. In the second scenario UAS do not have capability to support Trickle ICE. In this case, since UAC do not have any knowledge about the capability of UAS before initiating the negotiation, it will send the initial offer / answer without ICE candidates and get to know that remote User Agent do not support Trickle ICE. Now UAC waits for complete gathering of ICE candidates and then sends the subsequent offer.

From the above two scenarios, since the probability to support or not to support the incremental provisioning of ICE candidates by the remote User Agent is equal. UAC has equal probability, 0.5, either to have a successful reduced session establishment time or failure by sending multiple offer / answer negotiation messages, in cases of UAS supporting or not supporting Trickle ICE feature respectively.

In order to overcome this problem WebRTC working group proposed one more intermediate solution. The UAC will wait until the first ICE candidate, generally host candidate to gather and then starts the negotiation. Later it adds the other candidates gathered in subsequent offer /answers similar to that as explained above. With this approach, ICE candidate pings for checking the connection will start immediately after the first offer / answer negotiation. This will result in chances for establishing a transport connection immediately after the initial offer / answer negotiation. However a successful connection establishment in this case will be possible if both the User Agents are with public IP address. In reality the chances of having public IP addresses for both the User Agents is very

less when compared to that of having either one or both the User Agents to be behind the NAT / Firewall. In such scenarios, User Agents has to wait for a time required for gathering the initial ICE candidates.

But in all these approaches, UAC and UAS have to undergo a minimum of two offers / answers negotiation cycles to complete the negotiation process. This will result in increased signaling messages that are more costly, in the scenario where the UAS not having the capability to support incremental provisioning of ICE candidates. This problem demands the need for identifying the capability of UAS to support Trickle ICE before initiating the session negotiation.

One of the existing methods that give the information about the capabilities of a User Agent to other User Agents is Presence Information. Presence Information is an xml structure and is termed as Presence Information Data Format (PIDF) structure.

Presence Information is primarily designed for Instant Messaging (IM) which was later enhanced for other kinds of sessions like voice and video calling etc. Presence Information primarily provides the information about the availability of the user. A typical presence service model is shown in Figure 1 [9]. A WATCHER can get the presence information either through notifications from the presence service or by requesting the presence service at regular intervals of time called FETCHER which internally works based on a POLLER. The WATCHER, shown in Figure 1, is a subscriber based application. The WATCHER applications send the Subscribe request to the Presence Service to register the notifications of the PRESENTITY status changes from peers. PRESENTITY is the service that actually publishes the presence status of the User Agent. PRESENTITY registers with the Presence Service and sends the PRESENTITY status updates of the User Agent. Based on the status changes of the PRESENTITY, Presence Service sends the information to WATCHER in the form of notifications.

Presence Information is distributed in the xml format called PIDF. A new media type, “application/pidf+xml”, is also defined to represent the XML MIME (Multipurpose Internet Mail Extension) entity for PIDF [10]. One of the characteristics of PIDF was that the document always needs to carry all the presence information available for presentity. This specification was extended to carry partial presence information, excluding the unchanged information in the new PIDF message [11].

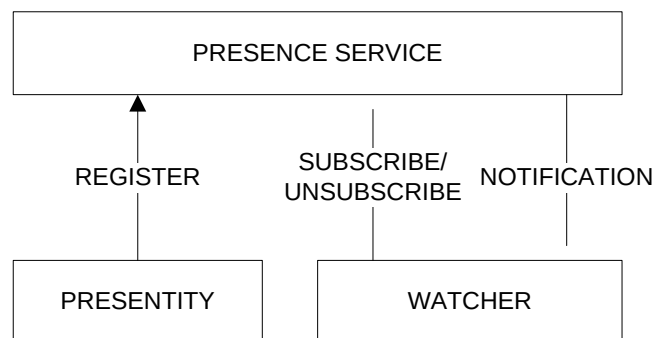


Figure 1. Presence Service

An example simple presence information message without Trickle ICE information in its status tag is as shown in Figure 2. The xml version and namespace tags specify the default xml version and namespace of the xml format. The <status> tag provides the information about the availability of the User Agent. <location> tag conveys the location of the User Agent. And optional <contact> tag gives the contact information of the User Agent. Apart from those shown in Figure 2 Presence information can also provide the priorities for contact address, time stamp of the change of this tuple information and human readable text comments etc.

Session Initiation Protocol (SIP) [12] is widely used IP based protocol for establishing, maintaining and terminating audio / video / IM sessions. The presence event packaging has been extended to support SIP based applications through subscriptions and notifications of presence [13]. The data model used by SIP for IM and Presence Leveraging Extensions (SIMPLE) [14] provides guidance on how to map various communication systems into presence documents.

MiYoung Hub, SunOk Park, Wook Hyun, JaeChon Han, IlJin Lee and ShinGak Kang explained the basic call flows for interoperability test in the presence services based on SIP [15]. They described the internal logical system of the presence services system.

Ron Shacham, Henning Schulzrinne, Wolfgang Kellerer and Srisakul Thakolsri described the composition for enhanced presence [16]. They proposed the use of presence composition to remote User Agent about the unusable data and create more usable new data. They also explained the policy for composition that a user can specify.

Dongmei Jiang, Tet Hin Yeap, Luigi Logrippo and Ramiro Liscano described three layered architecture for communicating personalized information for SIP multimedia communications [17]. It enhances presence information by extending the Call Processing Language (CPL) [18] to communicate the personalization information like location, role, availability, line status etc., apart from the simple online / offline indicators.

The following section explains the proposed system that publishes the User Agent's support for Trickle Ice capability through presence information.

```
<?xml version="1.0" encoding="UTF-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
  xmlns:local="urn:example-com:pidf-status-type"
  entity="pres:someone@example.com">
  <tuple id="ub93s3">
    <status>
      <basic>open</basic>
      <local:location>home</local:location>
    </status>
    <contact>im:someone@example.com</contact>
  </tuple>
</presence>
```

Figure 2. Presentity Information without Trickle ICE

III. PUBLISHING TRICKLE – ICE CAPABILITY THROUGH PRESENCE INFORMATION

User Agents, involved in peer to peer communications, may have different capabilities apart from just online / offline presentity status. The capabilities that can be added to the presence information has been extended to publish the status like busy / away, location information, active account, information about the multiple logins, line status etc., along with the online / offline presentity state [11].

Trickle ICE is a common feature / capability that a User Agent can support for establishing a session with any peer in its Buddy List. User Agent's Support for Trickle ICE feature is specific to that User Agent and does not dependent on any other entity. Trickle ICE is a specific and common capability just like that of any other capabilities of a User Agent like location, role, availability etc. that persists commonly for all the Buddies. So it is easy for User Agents to publish the support for Trickle ICE capability through presence information.

User Agents capability for supporting / not supporting Trickle ICE feature will not change over the User Agents session, in general. The scenarios where the capability of User Agents for supporting Trickle ICE feature may change are continuing an established session from a different device with secondary login or changing the device and thereby the priority of contact information before establishing the session.

User Agents support for Trickle ICE feature generally will not change over the time period. So User Agents can convey the support for Trickle ICE capability, to other peers, in initial presence information document (PIDF). Since this information will not change over the time, there will be no need to send this information in updated partial presence information. This will reduce / will not increase the amount of information exchanged between the peers in subsequent update messages and will be similar to that of without Trickle ICE capability.

A simple example of Presence Information message that is carrying the information about the support of Trickle ICE capability, in the xml format, is as shown in Figure 3. This Presence Information message is similar to that of shown in Figure 2. Apart from that it carries the presence information in <local:trickleIce> tag of the <status> tag.

```
<?xml version="1.0" encoding="UTF-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
  xmlns:local="urn:example-com:pidf-status-type"
  entity="pres:someone@example.com">
  <tuple id="ub93s3">
    <status>
      <basic>open</basic>
      <local:location>home</local:location>
      <local:trickleIce>yes</local:trickleIce>
    </status>
    <contact>im:someone@example.com</contact>
  </tuple>
</presence>
```

Figure 3. Presentity Information with Trickle ICE

ICE specifications provide the facility for backward compatibility while using Trickle ICE feature. If a UAS does not support Trickle ICE feature, UAC can easily fall back to support ICE capability in the old format [1] by configuring the application level settings. Since the applications can identify the support for Trickle ICE capability of UAS before initial session establishment request, In case if UAS is not supporting Trickle ICE capability, the number of request / responses exchanged between the two peers can be reduced. The signaling request / response messages exchanged for establishing a session between two peers can be either by using standard Session Description Protocol (SDP) [19] or any proprietary signaling means.

Presence information that includes the other capabilities of a User Agent may change from time to time based on the user actions / application configurations. The initial PIDF format was such that the complete presentity information was notified to the peers even with a small change in the presentity information. This PIDF format was enhanced [11] to notify only the changed / updated presentity information in the new / subsequent PIDF message. Rest of the information, without changes in any capabilities, will not be shared to the peers. Figure 4 presents an example PIDF message that updates the state of the user to busy. This updated PIDF message does not contain the information about location / trickle Ice / contact details. It contains only the mandatory fields and the updated presentity status of the User Agent. In Figure 4 User Agent publishes to peers that the user is busy using the <im:im> tag. It does not contain <location>, <trickleIce> and <contact> tags of the xml PIDF format.

The following section theoretically describes the results with the proposed system, publishing Trickle Ice capability through presence information document format.

IV. RESULTS AND DISCUSSIONS

Transmitting signaling messages between peers is more costly when compared to that of media between peers. Signaling messages plays the key role in reducing the session establishment time. Trickle ICE is a feature that is proposed and being standardized by IETF to reduce the session establishment time.

```
<?xml version="1.0" encoding="UTF-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
  xmlns:local="urn:example-com:pidf-status-type"
  entity="pres:someone@example.com">
  <tuple id="ub93s3">
    <status>
      <basic>open</basic>
      <im:im>busy</im:im>
    </status>
  </tuple>
</presence>
```

Figure 4. Partial Presence Information

Using the proposed system, UAC can get the knowledge of UAS's support for Trickle ICE capability through presence information before establishing the session. By having knowledge about the support of Trickle ICE capability by UAS before initiating the session reduces the number of signaling messages exchanged between the two peers, in case if both the peers supports Trickle ICE capability.

Supporting or not supporting of Trickle ICE capability by either of the User Agents results in the following four scenarios.

- Both UAC and UAS not supporting Trickle ICE feature
- UAC not supporting Trickle ICE feature but UAS supports Trickle ICE feature
- UAC supports Trickle ICE feature but UAS does not supports Trickle ICE feature
- Both UAC and UAS supports Trickle ICE feature

In the scenario of both UAC and UAS does not support the Trickle ICE feature, there is no need for UAC to check for the remote peers capability to support Trickle ICE feature. UAC will wait until all the ICE candidates are gathered and then initiate the session with the UAS. Since UAS also does not supports the Trickle ICE feature, UAS also waits until all the ICE candidates are gathered before responding to the UAC request for establishing the session. This results in transmitting only 1 signaling message from each peer for establishing the session.

In the scenario where UAC does not supports the Trickle ICE feature and that of UAS supports it, UAC ignores the capability of UAS for supporting Trickle ICE feature. UAC waits for completing the ICE candidate gathering process before initiating the session establishment process. UAS understands through the request from UAC that UAC does not support the Trickle ICE feature and responds to the UAC's request after gathering all the ICE candidates. In this scenario also, only 1 signaling message is transmitted from each peer for establishing the session.

In the scenario where UAC supports Trickle ICE feature and UAS does not support Trickle ICE, the following cases will arise in existing system. UAC can either send the initial request without ICE candidates or it can send the initial request with a minimum set of ICE candidates, generally one candidate resulting in partial Trickle ICE support, termed as Half Trickle ICE [4]. In both the cases UAC has to send 2 requests to UAS. Based on the requests received UAS also has to send 2 corresponding responses. Where as in case of proposed system, since UAC has the knowledge that UAS does not support Trickle ICE feature, UAC sends only 1 request after completing the ICE gathering process and receives the corresponding response from the UAS. This reduces 1 signaling message exchange between the two peers.

In the scenario where both UAC and UAS supports Trickle ICE feature, results in initiating the requests either without ICE candidates or with minimum set of ICE candidates, resulting in Half Trickle ICE. Consider the case where UAC initiates the

request without ICE candidates and the worst case scenario that results in sending three types of ICE candidates in there different messages. In this case UAC sends 4 requests for establishing the session. Similarly UAS also sends the corresponding 4 responses to the requests received from UAC. Even though the number of message exchanges is more in this case, the time required for reducing the session establishment time reduces.

The number of signaling messages required to be initiated from a single peer for establishing a session are depicted in Table I. The total bandwidth utilization for completing the session setup will be considered by adding the bandwidth utilized for sending the request and response from each peer in each of the different scenarios respectively.

Let us consider, for example, SIP as the signaling protocol used for establishing the session. Considering the sizes of SIP signaling messages, analyzed from an average of 25 experimental results, to be 350 bytes, the size of SDP indicating a simple audio session is 450 bytes and the size of each ICE candidate is 110 bytes each as that considered in “MIME Types for ICE Candidates to Reduce SIP Session Setup Signaling Messages” [8]. The total bandwidth utilized for sending and receiving a single message request – response transaction from each User Agent is

$$B_{invd} = S_I + S_{180} + 2S_{sdp} + 2\sum_{i=1}^3 S_{cand} \quad (1)$$

Where S_I and S_{180} are the sizes of SIP Invite and 180 Ringing response messages, S_{sdp} is the size of SDP message and S_{cand} is the size of each ICE candidate. So the total bandwidth utilized as per the sizes assumed is 2260 bytes.

If the negotiation is initiated without ICE candidates in first message and then with all the ICE candidates in the next message, then the corresponding bandwidth utilized are as shown in equations (2) and (3) respectively.

$$B_{invb} = S_I + S_{180} + 2S_{sdp} \quad (2)$$

$$B_{upd} = S_U + S_{200} + 2S_{sdp} + 2\sum_{i=1}^3 S_{cand} \quad (3)$$

Now the total bandwidth utilized for these two request response transactions is from equations (2) and (3) are 1600 bytes and 2260 bytes respectively resulting in a total bandwidth usage of 3860 bytes. From equations (1) and (3) it is observed that there is a total increase of in bandwidth usage 41.45%.

If the same messages are transmitted using that of ICE candidate MIME type “application/icecandidate” defined in [8] “in press”, then equation (3) will be replaced by equation (4).

$$B_{upd} = S_U + S_{200} + 2\sum_{i=1}^3 S_{cand} \quad (4)$$

TABLE I. NUMBER OF SIGNALING MESSAGES REQUIRED FOR ESTABLISHING SESSION

Trickle ICE	Number of signaling messages sent from each User Agent in existing system	Number of signaling messages sent from each User Agent in proposed system
UAC not supports UAS not supports	1	1
UAC not supports UAS supports	1	1
UAC supports UAS not supports	2	1
UAC supports UAS supports	4	4

From equations (2) and (4), bandwidths utilized for secession setup are 1600 bytes and 1360 bytes respectively, resulting in a total increase of 2960 bytes. From equations (1) and (4) it is observed that there is a total increase in bandwidth usage of 23.64%.

A similar kind of reduction in bandwidth usage will be possible with any IP based signaling protocols, by identifying the support for incremental provisioning of ICE candidates before initiating the session setup procedure. The other majorly used IP based signaling protocol include Extensible Messaging and Presence Protocol (XMPP) and any other proprietary signaling protocol. XMPP is having the capability of transmitting transport information, namely one or more ICE candidates, separately without SDP.

Presence service, in general, stores the presence information of the Presentity. Since the presence services need to store the indication about the support for trickle ICE, it needs an extra space of approximately 40 bytes for each Presentity that supports trickle ICE feature and intimates it to remote peers through presence information. The total amount of increase in the storage space required to store the trickle ICE information is directly proportional to the number of users registered to that Presence service.

V. CONCLUSION AND FUTURE WORK

Trickle ICE is a feature that was proposed and being standardized by IETF to reduce the session establishment time. This work proposed the way for communicating the capability for supporting Trickle ICE feature by User Agent through Presence information. The importance of understanding the remote peers capability to support Trickle ICE in reducing the number of signaling messages required to establish a session in different scenario have been described. It is identified that this information is more useful in case of UAC supports Trickle ICE capability and UAS does not supports Trickle ICE capability. The observed total increase in bandwidth usage is 41.45% and 23.64% in case of using the existing SIP message headers and that of using MIME type for ICE candidates.

As part of future work, this work will be continued by analyzing the amount of time required for establishing a session between two peers in different network conditions and in different scenarios, like UAC and UAS that supports / not supports Trickle ICE feature, and also in conditions like having and not having the prior information about the support for Trickle ICE from the remote peer.

ACKNOWLEDGEMENT

I would like to thank Sachin Dev and Rajaram Hanumantacharya Naganur for their valuable inputs and suggestions.

REFERENCES

- [1] J. Rosenberg "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", IETF RFC 5245, April 2010.
- [2] J. Rosenberg, R. Mahy, P. Matthews, D. Wing, "Session Traversal Utilities for NAT (STUN)", IETF RFC 5389, October 2008.
- [3] R. Mahy, P. Matthews, J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", IETF RFC 5766, April 2010.
- [4] E. Iovov, E. Rescorla, J. Uberti, "Trickle ICE: Incremental Provisioning of Candidates for the Interactive Connectivity Establishment (ICE) Protocol", IETF draft-ietf-mmusic-trickle-ice-01, February, 2014.
- [5] J. Uberti, C. Jennings, "Javascript Session Establishment Protocol", IETF draft-ietf-rtcweb-jsep-06, February 2014.
- [6] Adam Bergkvist, Daniel C. Burnett, Cullen Jennings, Anant Narayanan, "WebRTC 1.0: Real-Time Communication between Browsers", W3C Editors Draft, July 2014, <http://w3c.github.io/webrtc-pc/>.
- [7] E. Burger, "Critical Content Multi-purpose Internet Mail Extensions (MIME) Parameter", IETF RFC 3459, 2003.
- [8] Kiran Kumar Guduru, Usha J, "MIME Type for ICE Candidates to Reduce the Size of SIP Session Setup Signaling Messages", IEEE International Conference On Communication System and Network Technologies, 2015.
- [9] M. Day, J. Rosenberg, "A Model for Presence and Instant Messaging", IETF RFC 2778, February 2000.
- [10] H. Sugano, S. Fujimoto, G. Klyne, A. Bateman, W. Carr, J. Peterson, "Presence Information Data format", IETF RFC 3863, August 2004.
- [11] M. Lonnfors, E. Leppanen, H. Khartabil, J. Urpalainen, "Presence Information Data Format (PIDF) Extension for Partial Presence", IETF RFC 5262, September 2008.
- [12] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnson, J. Peterson, R. Sparks, M. Handley, E. Schooler, "SIP: Session Initiation Protocol" IETF RFC 3261, 2002.
- [13] J. Rosenberg, "A Presence Event Package for the Session Initiation Protocol (SIP)", IETF RFC 3856, August 2004.
- [14] J. Rosenberg, "A Data Model for Presence", IETF RFC 4479, July 2006.
- [15] Mi Young Huh, SunOk Park, Wook Hyun, JaeChon Han, IlJin Lee, ShinGak Kang, "Basic Call Flow for Interoperability Test in the Presence Services based on SIP", The 8th International Conference on Advanced Communication Technology, 2006, PP 836 – 839, ISBN 99-5519-129-4.
- [16] Ron Sacham, Henning Schulzrinne, Wolfgang Kellerer, Srisakul Thakolsri, "Composition for Enganced SIP Presence", 12th IEEE Symposium on Computers and Communications, 2007, PP 203 – 210, ISBN 1-4244-1521-7.
- [17] Dongmei Jiang, Tet Hin Yeap, Luigi Logrippo, Ramiro Liscano, "Personalization for SIP Multimedia Communications with Presence", 2005 International Conference on Services Systems and Services Management (ICSSSM), 2005, PP 1365 – 1368, ISBN 0-7803-8971-9.
- [18] J. Lennox, H. Schulzrinne, "Call Processing Language (CPL): A Language for User Control of Internet Telephony Services", IETF RFC 3880, October 2004.
- [19] M. Handley, V. Jacobson, C. Perkins "Session Description Protocol", IETF RFC 4566, July 2006.



Kiran Kumar Guduru received M.C.A degree from Sri Venkateswara University in 2009 and currently pursuing his Ph.D under Visvesvaraya Technological University, India. He is working as a Lead Engineer in Samsung R & D Institute India, Bangalore PVT Ltd. He is a member of W3C WebRTC and IETF RTCWEB working groups since 2012. His research interests include reducing network congestion in telecom signaling networks.