Efficient IPv4-IPv6 translation mechanism for IMS using SIP proxy

Ali A. Khudher*

Department of Computer Science,
College of Education for Pure Science,
University of Mosul,
Mosul, Iraq
Email: aliabd@uomosul.edu.iq
*Corresponding author

Alhamza Munther

University of Technology and Applied Sciences,
Muscat, Ibra, Oman
Email: alhamza.sur@cas.edu.om

Supriyanto Praptodiyono

Electrical Engineering Department,
Universitas Sultan Ageng Tirtayasa,
Banten, Serang and Cilegon, Indonesia
Email: supriyanto@untirta.ac.id

Abstract: Next-generation IP Multimedia Subsystem (IMS) has a promise future in internet technologies. Technically, IMS utilises Session Initiation Protocol (SIP) for call communication. Potentially, SIP clients might reside in coexistence network family like IPv6 and/or IPv4. The coexistence of SIP clients considered a serious issue in SIP communication and IPv6 migration in general. In this paper a SIP-Proxy IP Translator (SPIPT) is proposed to translate SIP call between from IPv6 to IPv4 and vice versa. The proposed solution comes to insure successive connectivity in term of IP version compatibility. The experiment shows acceptable CPU usage, throughput, and Call Response Time parameters. SIP proxy hits negligible amount (1.93%) of CPU usage when 100 calls load. In addition, throughput also stays with the safety level with 1.9 Kbps in the same 100 calls. In sum, the solution aims to reduce call setup complexity and leverages user experience which in turn accelerates IPv6 transition.

Keywords: SIP; IPv6; SIP over IPv6; IPv4-to-IPv6; SIP service provider.

Reference to this paper should be made as follows: Khudher, A.A., Munther, A. and Praptodiyono, S. (2022) 'Efficient IPv4-IPv6 translation mechanism for IMS using SIP proxy', *Int. J. Internet Protocol Technology*, Vol. 15, No. 1, pp.41–52.

Biographical notes: Ali A. Khudher received his BSc degree in Computer Science from Mosul University, Iraq in 2003, MSc degree in Computer Science from Universiti Teknologi Malaysia, Malaysia, in 2008 and PhD degree from the National Advanced IPv6 Centre, University Science Malaysia (USM), Malaysia in 2014. Currently, he is a Senior Lecturer at Computer Science Department, University of Mosul, Iraq. His current research interests include multimedia communication, VoIP, SIP technologies and IPv6 connectivity.

Alhamza Munther is an Assistant Professor in College of Applied Sciences, Sultanate of Oman. He received his BSc degree in Computer and Software Engineering from University of Technology, Baghdad in 2003, Master's degree in Advanced Computer Networks from Universiti Sains Malaysia (USM) in 2012 and PhD degree in Computer Engineering from Universiti Malaysia Perlis, Malaysia, in 2017. His research interests include on overlay networks, multimedia distribution and traffic engineering.

Supriyanto Praptodiyono received his BEng degree in Electrical Engineering from Brawijaya University Malang, Indonesia in 1999, MSc and PhD degrees in Advanced Computer Networks from the National Advanced IPv6 Centre, Universiti Sains Malaysia (USM) Malaysia 2010 and 2015, respectively. His research interest includes computer networks, IPv6 security and wireless communication.

1 Introduction

Next generation IP Multimedia Subsystem (IMS) found by Camarillo and García-Martín (2008) has been introduced by 3rd Generation Partnership Project (3GPP) standards to provide real-time communication services like Voice over IP (VoIP) (Bertrand, 2007). IMS utilises Session Initiation Protocol (SIP) published its first RFC by Rosenberg et al. (2002) as core protocol for handling call sessions by initiating, modifying and tearing down multimedia sessions, (Johnston, 2015). SIP entities potentially reside in different network islands (IPv4 or IPv6). One SIP element can be configured with IPv4, IPv6, or IPv4/IPv6 (dual-stack). Recently, SIP over IPv6 continues to be employed and deployed around the world, (Hoeher et al., 2007).

Just like other technologies, SIP has to cope with IPv6 migration and meet its requirements by providing a seamless integration and coexistence strategies stated by Cui et al. (2012). In fact, SIP coexistence between IPv4 and IPv6 does not come without challenges, (Khudher, 2019). One significant issue arises from breaking the call during routing from one network type to another (e.g. IPv6 to IPv4). For instance, when IMS User Equipment (UE) with IPv4/IPv6 dual-stack sends a call with IPv6 INVITE message toward end-user SIP device which located in IPv4-only network, the call will fail. Technically, IPv4-only devices will reject the whole SIP transactions that contain IPv6 INVITE message that is due to the incompatibility between IPv4 and IPv6 headers, it even does not understand its format. This issue resulted to unaccomplished calls and degrades user's satisfaction. In turn, those issues on other hand will affect the acceleration of IPv6 transition, (Jabir, 2020).

To overcome this problem, several solutions have been proposed in literature. Earlier solution was the SIP-ALG proposed by Chen et al. (2004, 2005) and Han et al. (2006) which shows that the SIP server aims to establish a SIP end-to-end connection (both SIP and RTP packets) between the IPv6 end-device and IPv4 end-device. SIP-ALG initially translates SIP messages in one hand, and RTP proxy translates RTP packets in the other hand utilising unallocated port number. With Chen et al. (2010) solution, CSCF-translation only understand that the remote party resides at another network family after the SDP *c* field has examined in '200 OK' message. Once confirm, SIP server in such case starts to invoke CSCF function to translate IPv6 header and forwarded it toward remote-device in IPv4 network. Another solution Johnston et al. (2003) proposed Redirect Solution which aims to translate between IPv4 and IPv6 by checking the registration database. Immediately, SIP server get informed that the remote-end device is configured with IPv4. In such case, a SIP-server orders UAC to re-initiate (reINVITE) the call but in this time using IPv4. A recent solution by Chen and Li (2013) proposed a translation mechanism in the client side rather than the servers. All the mentioned solutions are still suffering from drawbacks such as message-flow redundant, database penalty and call complexity.

The aim of this paper is to propose SIP-Proxy IP Translator (SPIPT) which translates between IPv6 to IPv4 version in the real-time. The IPv6 translation comes to insure successful and smooth SIP call routing. In Addition, SPIPT proposed to reduce call-setup complexity and database penalty that been noticed in related works from literature.

We shall begin by presenting the background of SIP in Section 2. Related work details are provided in Section 3. Section 4 describes the proposed solution. Then, Section 5 shows the implementation part followed by evaluation in Section 6. Last, conclusion is presented in Section 7.

2 Session initiation protocol

SIP is a signalling protocol defined by SIP Working Group, under the Internet Engineering Task Force (IETF), (Shacham et al., 2009). The protocol was first published as IETF (RFC 2543) in a track to be proposed standard, (Vemuri and Peterson, 2002). SIP in general, by most cases, used to control multimedia (voice, video) by establishing, modifying and terminating sessions. SIP works jointly with Session Description Protocol (SDP), (Handley et al., 2006) which is used to negotiate for multimedia parameters such as Codec/decodec, IP address and port numbers for Real-Time Transport Protocol (RTP), (Schulzrinne et al., 2003) streams. SIP elements include UAs and SIP server (proxy, redirect, location and registrar). During call setup, SIP proxy manipulates IP addresses on the fly in order to route request messages to desired end point device. However, these IP addresses are dynamically set in certain messages' headers such as 'Contact', 'Request URI' and 'Via' headers.

2.1 SIP over IPv6

SIP was implemented at a time since IPv6 already a few years there. This means SIP technologies have built in IPv6 at the earlier stage. The 3GPP was the real group beyond implementing SIP over IPv6, (Gurbani et al., 2008). Each SIP entity registered with unique address, which is IP address. That IP address can be v4 or v6, (Sisalem et al., 2003). In both cases the IP presented in SIP message for registration and routing purpose. Thus, focusing on that part of the message will narrow the difference between IPv4 and IPv6 routing, (Davies, 2012). Within SIP message, IPv6 presented clearly in URI, From, To, Contact, and Via headers as shown in Figure 1 (Minoli, 2011; Khudher et al., 2013). The task of those headers is to determine proxy path where a request navigates through SIP network and guides corresponding responses back into the same path, (Kim, 2004). Accordingly, IPv6 also appeared in SDP, exactly in *c* and *o* parameters in order to route media packet through RTP, (Zourzouvillys and Rescorla, 2010).

Figure 1 SIP message with IPv6

```
INVITE sip:9999@[2001:db8:3c4d:1:202::]:5060 SIP/2.0.
Via: SIP/2.0/UDP [2001:db8:3c4d:1:5::]:5060;branch=z9hG4bK-2396-1-0.
From: ali0 <sip:ali0@[2001:db8:3c4d:1:5::]:5060>;tag=1.
To: sut <sip:9999@[2001:db8:3c4d:1:202::]:5060>.
Call-ID: 1-2396@2001:db8:3c4d:1:5::.
CSeq: 1 INVITE.
Contact: sip:ali0@[2001:db8:3c4d:1:5::]:5060.
Max-Forwards: 70.
Subject: Performance Test.
Content-Type: application/sdp.
Content-Length:  151.
.
v=0.
o=user1 53655765 2353687637 IN IP6 [2001:db8:3c4d:1:5::].
s=-.
c=IN IP6 2001:db8:3c4d:1:5::.
t=0 0.
m=audio 6000 RTP/AVP 0.
a=rtpmap:0 PCMU/8000.
```

3 Related work

During the transition of IPv6, SIP connection has faced several challenged. One significant issue was the interoperation between IPv4 and IPv6 networks which lead to loss connection and degrade user satisfaction as stated by (Khudher, 2019). Meanwhile, several researches have been conducted to overcome this problem. The following subsections present several solutions in around this area.

3.1 SIP ALG solution

In the ALG, the server aims to translate SIP INVITE messages headers while RTP-proxy handle the media packets translation (audio) in RTP packets. The translation mechanism in Figure 2 illustrated with high-level description as follow:

Step 1: UAC creates SIP request by sending INVITE message addressed to most-end client UAS. The (INVITE) message carries UAC‘s IPv6 address [2001:db8:3c4d:1:5::] in the ‘Contact’ header and also the same IP inside ‘Via’ header field.

Step 2–3: Once the server receives the expected INVITE message, immediately the server checks the IP family of UAC from the Contact header and UAS IP from the server database. The SIP server notified that both IPs are from different families which mean IPv4 and IPv6. Based on this case, the server performs the translation using SIP-ALG solution and that can be done by replacing the Via and Contact headers from IPv6 2001:db8:3c4d:1:202:: to IPv4 address 192.168.1.3 then forward the message to the destination UAS.

Step 4: Upon receipt of INVITE message, UAS sends ‘180 Ringing’ message to UAC with IPv4. The message travel back based on the IP stated in the Via header.

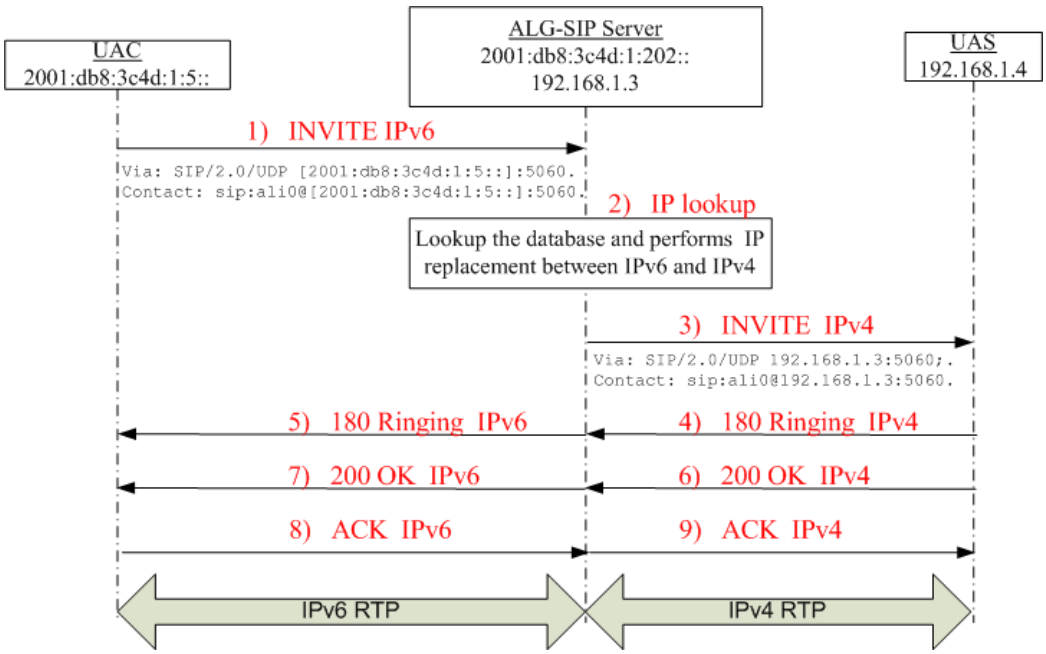
Step 5: On receipt of ‘180 Ringing’ message, the server manipulates the corresponding SIP headers from IPv4 (192.168.1.3) such as ‘Via’ header with IPv6 [2001:db8:3c4d:1:5::]. Also, the UAS’s IPv4 address 192.168.1.4 in the ‘Contact’ header is manipulated with the IPv6 address [2001:db8:3c4d:1:202::] of SIP server’s . Then, the translated ‘180 Ringing’ headers is forwarded to UAC.

Step 6: once the end-remote device answers that call, it responses the server with IPv4 200OK by filling its IPv4 address 192.168.1.4. In addition, it has to manipulate the port numbers in SDP *m* and *c* fields. Then, the ‘Via’ header will be extracted from the ongoing INVITE message to be re-addressed again toward SIP server.

Step 7: In this step, SIP server manipulates IP address in both ‘Via’ and ‘Contact’ headers. In parallel, RTP-proxy generates a new IP:port mapping to forward its RTP packets between the remote-end device and SIP server. The IPv6 address [2001:db8:3c4d:1:202::] and the new generated port number will be manipulated by SIP-ALG function.

Step 8–9: An acknowledge message (ACK) will confirm the connection with a new translated IPv4. Lastly, the both end parties will exchange their messages/media with new IP version sets.

Figure 2 Message passing in SIP-ALG solution



3.2 Redirect solution

The redirect solution is accomplished by querying the registration database. The database confirmed that the UAS is registered with an IPv4 address. Based on that confirmation, SIP server will notify the UAC to use IPv4 and should send the new INVITE message using same IPv4 family. The translation mechanism is illustrating in Figure 3 and in the following steps:

Steps 1–3: UAC generates an IPv6 INVITE message and send it toward the SIP server. The server in turn compares the IP family for both UAC and UAS. If they are in different network the SIP-server will directly inform the remote-end party with its IPv6 through '302 Move Temporarily' message. This message carries the UAS's IPv4 address 192.168.1.4 in its 'Contact' header indicating that IPv4 is the desired version to be used for this call.

Step 4: In this step, UAC device will acknowledge to server with its IPv6 'ACK' to the SIP server as a completion of the ongoing transaction.

Step 5: UAC regenerates a new INVITE message but this time with IPv4 and send it toward its destination UAS through its outbound SIP proxy.

Step 6: In this stage a remote-end device sends '180Ringing' IPv4 message through SIP-server back to UAC. Upon the success of this step, a same '180Ringing' message will be send to end device.

Steps 7–8: If a call answered by called party, an IPv4 '200 OK' message will be transmitted from end to end devices through SIP server. The address of the '200OK' will carry the IPv4 address 192.168.1.4 and port number belong to UAS's RTP (*c* and *m* fields). Later, when a '200 OK' message is received, UAC replies with IPv4 'ACK' message.

Step 9: Both parties will exchange the media packets through RTP transmission by using IPv4 only.

3.3 CSCF solution

Call Sessions Control Function (CSCF) is a translation mechanism that performs at SIP server side as proposed by Whai-En et al. (2005). When the INVITE message received by the SIP-server from the UAC it immediately applies the function of CSCF-translation in order to translate/translate-back IPv6 SIP message. Next, the server will forward the translated subsequence messages with IPv4 to remote-end device. The IPv4 '200 OK' received from the UAS indicating that the UAS only use IPv4 and it is ready to transmit using IPv4 RTP packet. The next steps illustrate CSCF translation and illustrated in Figure 4.

Step 1: Initially, UAC generate INVITE message (with IPv6) toward the UAS through SIP server.

Steps 2–3: Upon receipt of the INVITE message, SIP server informed that the end party UAS registered with IPv4 only. Immediately, the server applies the functions of CSCF-translation and starts to manipulate all IPv6 addresses in INVITE message to its IPv4. That can be done by utilising the SIP server's IPv4 address 192.168.1.3 to change the UAC's IPv6 address [2001:db8:3c4d:1:5::] in the 'Contact' headers field. Next, CSCF-translation access the database (SIP registrar server) to retrieves the corresponding IPv4-address 192.168.1.5 of UAC and changed with the [2001:db8:3c4d:1:5::] IPv6 address.

Step 4: At this step, UAS replies with '180Ringing' IPv4 message to UAC back to SIP-server. In this case, SIP-server will proxy an '180Ringing' message to UAC.

Step 5: Once the called party answer the call, the UAS replies with an IPv4 '200 OK' message, to include the UAS's IPv4 address 192.168.1.4 in the SDP *c* field.

Step 6: The UAC in this time utilises IPv4 (192.168.1.5) to send subsequence messages such as 'ACK' message to remote-end device. The predefined port number will be used as well.

Figure 3 Message passing in redirect solution

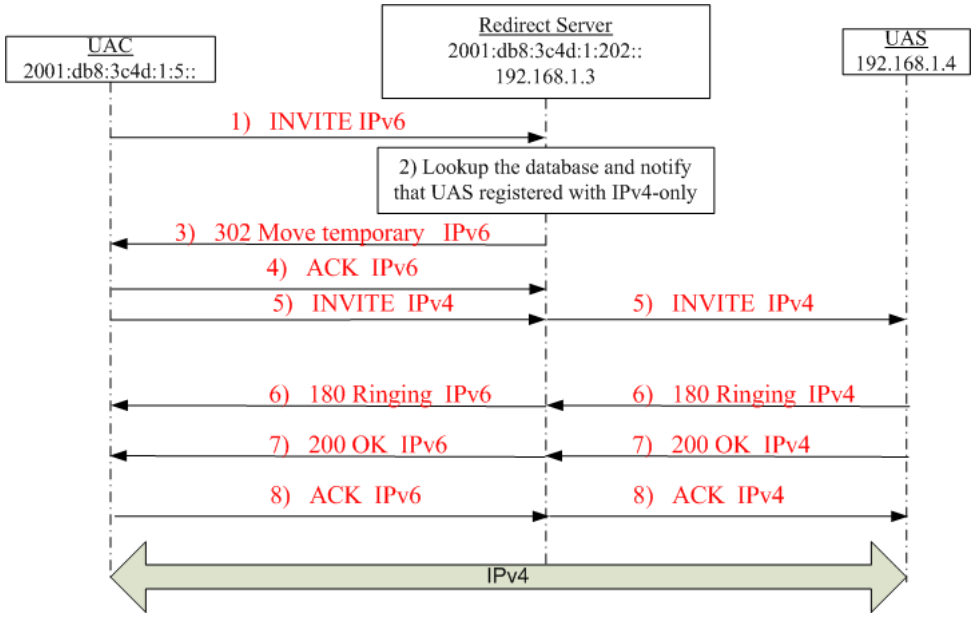
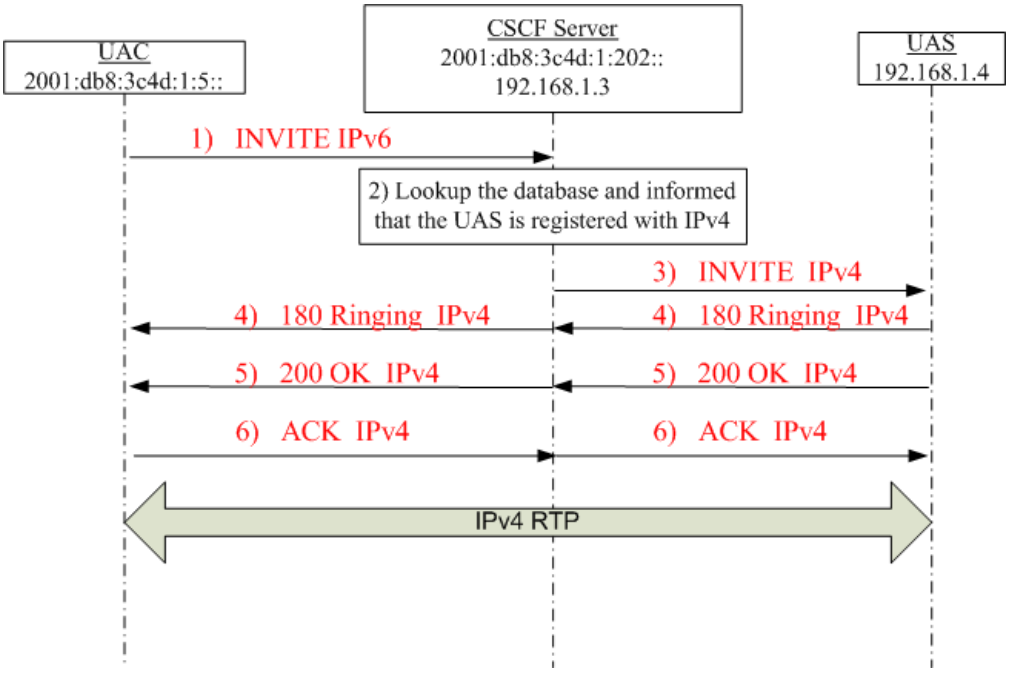


Figure 4 CSCF solution message passing



3.4 Client-based solution

In this kind of translation, the UAC device will offer pre-agreed IP for network capability. Using this, it made both end devices chose their preferable IP version for their subsequence transmission. Initially, UAC offers both IPv4 and IPv6 addresses to be used during audio transmission through RTP protocol. In turn, the UAS will pike up a suitable (or high priority) of IP version. After successive INVITE transaction, the UAC will gain the IPv4 address 192.168.1.4 as well as the port number of remote-end device. Finally, the RTP IPv4 packets will travel directly from end to end devices by passing SIP server. All the solutions mentioned in the literature above have some limitations such as call setup latency, call complexity, QoS and other degrading factors. For IMS-ALG solution utilised TrGW to translate RTP packets causes higher RTP transmission delay, packet loss and jitter. This occurs due to the extra connection to RTP server. Regarding Redirect-Solution, the extra STUN messages exchange increases call-setup latency. While in CSCF-translation solution, the database lookup penalty decreases session time during message signalling, and required extra database queries in order to check the called party IP address(s). However, Client-based solution shows another kind of drawbacks while communication with the last-end devices. The called

party (last-end) device might not be supported with IPv6 or the client does not configured from the user with IPv6 network. All the disadvantage mentioned above are correlated to each proposed solutions. However, there is a shared-point obstruction occurred in all solutions mentioned above which are IP comparison and message field extraction. In all cases, during the call setup, SIP server check the field URI, Contact, Via and *c* headers to determine IP family that the message use. Thus, this kind of operation considered to be time consumption and in turn will effect negatively to SIP signalling setup. In the next section, the proposed solution comes to override the limitation beyond the above-mentioned solutions.

4 SIP-Proxy IP translator

In this section the proposed solution is designed and described, to be implemented and evaluated in the next sections. Initially, SIP-Proxy IP Translator (SPIPT) is designed under the SIP proxy that manage the call between two parties, source (UAC) and destination (UAS). The called party (UAC) might be located in an IPv4-only network so it only understands IPv4 format and can only communicate using IPv4 in both signalling and media. In other case, the called party might be in IPv6-only network and unable to communicate with other IP families. Also, dual-stack strategy is found to make the end party configured with both networks, which means it listen to both IPv4 and IPv6 at the same machine, (Schinazi and Pauly, 2017). For the three cases mentioned above, we need a solution that deal with destination as it is with no further action to be taken from the end party. Actually, guessing the end UAS network family is the key-factor for any translations mechanism in SIP connectivity. When UAC initiates a call, other party might reside in IPv4, IPv6, or IPv4/IPv6 network. The call with different IP family may not reach to destination or resulted with incomplete transaction. The proposed SPIPT solution performs the translation by forwarding the call as it is to the end destination. If an end party located at the same IP family then the call accomplished without any extra operation needed. Otherwise, the INVITE will fail with negative '480 Temporary Unavailable' response. Based on that response, the server learned that UAS is located in different network resulting the server to relay a new INVITE with other IP family. See system flow-chart in Figure 5.

Figure 5 IP version translation logic

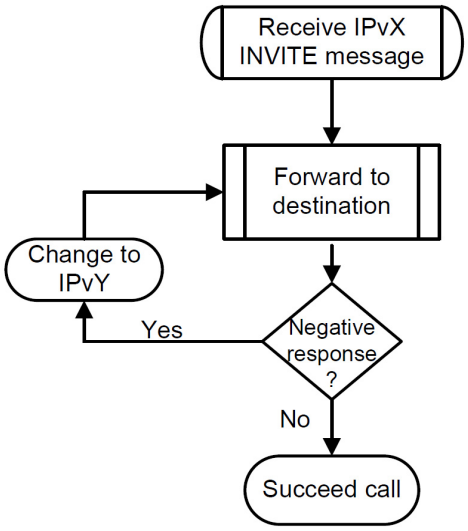
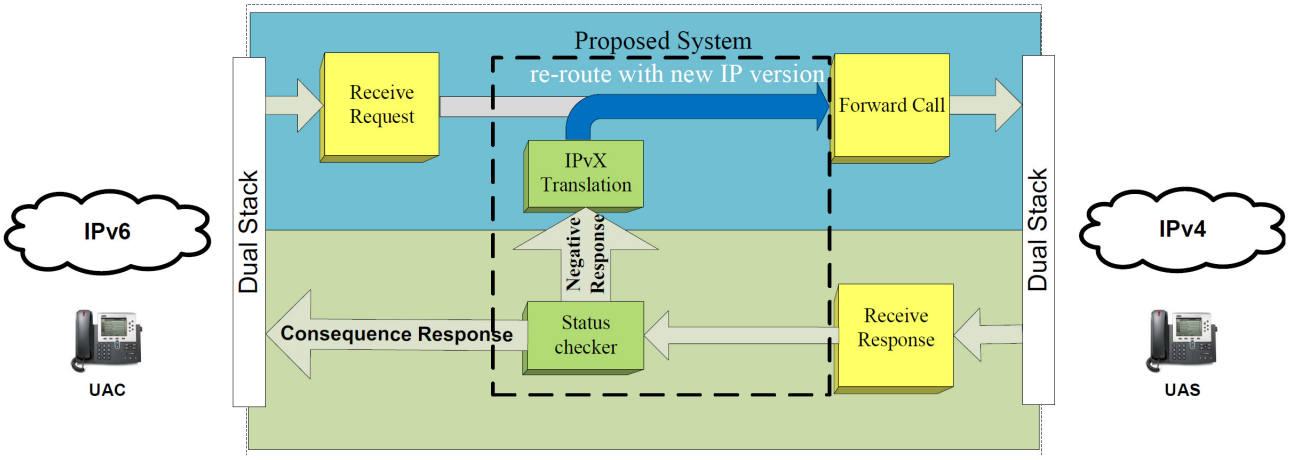


Figure 6 The proposed SIP translation components



For instance, if the call initiated with IPv4 and the destination is IPv6, the server will alter the IP family and forward using IPv6 and vice-versa. The proposed solution designed to cope with 4 cases which are as follow:

1 *IPv4 to IPv4 network*: It is a simplest and a normal case when an IPv4 UAC sends an INVITE message to an IPv4 UAS. Since both clients located within IPv4 network the call will established smoothly with no extra action required.

2 *IPv4 to IPv6*: In this case the UAC is an IPv4 address and the remote party is located in an IPv6 network (IPv6-only or IPv6/IPv4 dual-stack strategy) the server initially sends the INVITE request with an IPv4 address toward IPv6 end destination. The INVITE request will fail. Based on that the SIP server will receive '480 Temporary Unavailable' response message which indicates that the other party does not accept a call. The server immediately creates a now INVITE message this time with IPv6 and starts relay it toward the destination (Note that all SIP servers releases are listen to both IPv4 and IPv6).

   For the dual-stack client, if the UAS configured to use IPv4 firstly, then no translation will occur in this case (similar to case 1). On the other hand, if client configured to use IPv6 with high priority the server aims to translate the call by re-generating the call with new INVITE request using IPv6. The subsequence messages between server and UAS continues with IPv6 connection. The other leg connection between UAC and server will remain using IPv4. It is worth mentioning that the media session will be affected by associating the new IPs that the INVITE message obtained.

3 *IPv6 to IPv6*: When both UAC and UAS are using IPv6-only then no translation required and the call go smoothly without any error. However, if both devices configured with dual-stack, the devices will use either IPv6 or IPv4. That actually depends on the priority set in the device. There are many factors that really affect weight-priority by using different mechanism such as happy eyeball service. For instance, a duel-stack device might prefer to use IPv4 address rather than IPv6. That is probably due to QoS that IPv4 provide, best route used in IPv4 network, low contribution in IPv6 for that region/country. In such case, same translator will use a translation technique to translate from IPv6 to preferable IPv4 or Preferable IPv4 to IPv6. This scenario leaves the IP-list priority unchanged. Thus, no further DNS mechanisms required, which in turn avoid DNS-lookup penalty and resource/time consumption.

4 *IPv6 to IPv4*: This is a real case that all translation mechanism found for. This considered a critical case between two legs of communications which are IPv6 and IPv4. That is because all SIP clients that support IPv6 (whether IPv6-only or dual-stack) will choose IPv6 as a default address to initiate the communication with. If IPv6 client try to connect with the end party UAS which is located in IPv4 network the call is incomplete or fail at all. In this case, the proposed SPTIP receives an IPv6 INVITE message from UAC and forwards it toward UAS. The server acts as a proxy by sending the INVITE message to destination without any modification. Immediately, the server will receive '480 Temporary Unavailable' response from UAS indicating that the INVITE message does not succeed. The proxy will reroute the call but this time with IPv4 address (Note that the server can listen to both IPv6 and IPv4). Continuously, the subsequence transactions accomplished successively based on the first translation that occurred in the INVITE message. Figure 7 illustrates the message passing and the following explains the call steps details with high description level.

Figure 7 SPIPT solution message passing

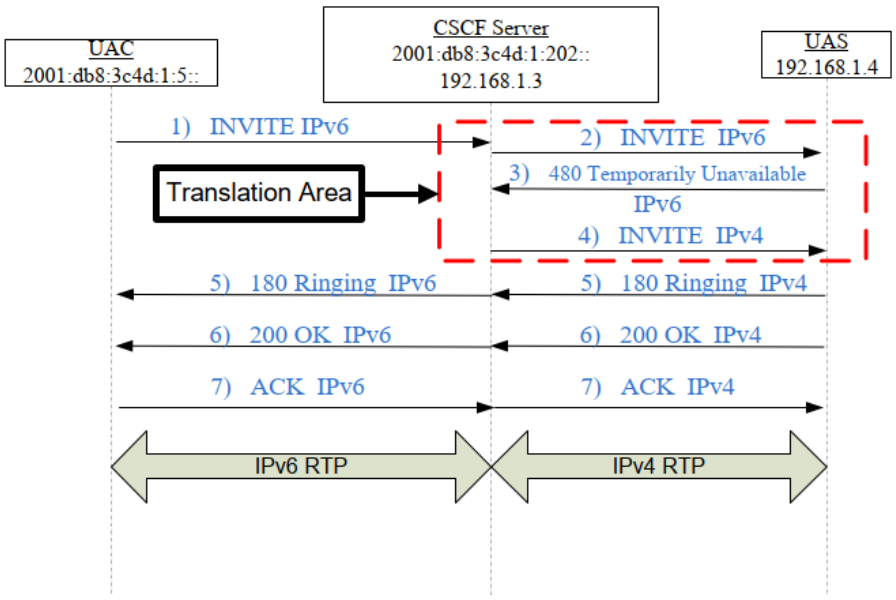


Table 1 Hardware specifications

Entity	*SIP proxy Server*	*UAC Machine*	*UAS Machine*
Domain name/IP	(IPv4:192.168.1.3) (IPv6:[2001:db8:3c4d:1:202::])	(IPv4:192.168.1.5) (IPv6:disabled)	(IPv4:disabled) (IPv6:[2001:db8:3c4d:1:4::])
Machine Model	Dell-Vostro (PC)	Dell-Vostro Laptop	Lenovo B570e Laptop
Operating System	Ubuntu 16.04.5 server	Debian Stretch version 9	Debian Stretch version 9
Kernel	4.4.0 -131-generic	4.9..0-7-686-pae	4.9..0-7-686-pae
RAM	2 Giga Byte	2 Giga Byte	4 Giga Byte
CPU	3.00GHz	900@2.20GHz	i3-2310M @ 2.10GHz
SIP software	OpenSIPs 2.4	SIPp 3.3	SIPp 3.3
Ethernet	100Mb/s	100Mb/s	100Mb/s

Step 1: Initially, UAC with IPv6 device generates an IPv6 INVITE message willing to communicate with the end party (UAS). The INVITE message will be routed toward the SIP server. By default it generates the INVITE message with IPv6 address. In this step the UAC does not know the UAS network family.

Step 2: Upon receipt of INVITE message, SIP server forwards the IPv6 INVITE message toward the end party UAS without any modification. If the UAS configured with IPv6 then the subsequence messages continues with successive call status. However, if the UAS is located at IPv4 network, then the call will fail.

Step 3: SIP server receives a 480 (temporarily unavailable) response message which indicating that the UAS is located at different network family and unable to understand the INVITE message. SIP server checks this response using *t-check-status* (exported function) in OpenSIPs server.

Step 4: in this step SIP server acts as a proxy by forwarding a new INVITE message toward UAS destination but in this time with IPv4 address. SIP proxy sends a new INVITE using *t-relay* function to keep the call with the same transaction and dialog. Keep in mind, even though the destination IP address has alter from IPv6 to IPv4 the session remains unbroken and does not loss the sequence. In other word, a new INVITE message that generated from the translation mechanism is not a reINVITE message. In this selution, reINVITE message is avoided here as the reINVITE message cause the transaction to be lost and force the sender to send a new INVITE using new connection. This will help to keep track of the call even the proxy status set to stateless.

Step 5: Upon the receipt of the new IPv4 INVITE message from the UAS, UAS will response with '180Ringing' message back to the server depend on the IPv4 (192.168.1.3) in the Via field that obtained INVITE header message. This step is indicating that the call was accepted without any problem.

Step 6: SIP proxy receives the '180 Ringing' message and sends it to UAC by replacing the Via header with UAC's IPv6 address 2001:db8:3c4d:1:5::. Note that, the Contact header is remained unchanged contains the IPv4 address of the UAS (192.168.1.4). Finally, the UAC start to play the ring-back media notifying the calling party.

Step 7 and 8: Once the called party accepts a call, a 200 OK message will be created and send to SIP server by using the IPv4 address (192.168.1.3) retrieved from the Via header. The main task of the 200 OK is to offer the main parameters that will be used for media communication. For that, the *c* header in the SDP will be filled with the UAS IPv4 address (192.168.1.4) along with the *m* header which contains the port number. SIP server refers to Via header to send the 200 OK toward the UAC. In this case, the top IP in the Via header is the UAC IPv6 address 2001:db8:3c4d:1:5::.

Step 9: after the UAC receives the 200 OK message it accepts the offer from the other party and directly confirms the call by sending an ACK message to the server using IPv6 address 2001:db8:3c4d:1:202::. The SIP server will transfer this ACK message toward the UAS using IPv4 address.

Step 10: At this point, both UAC and UAS start to communicate with each other by exchanging the RTP packet also with two legs of connection, IPv4 and IPv6, (Praptodiyono et al., 2019).

5 Implementation

The proposed solution implemented in local network, all machines are isolated on a 1000-Mbps to avoid exterior factors. SPIPT solution is implemented inside a SIP server, specifically SIP proxy. The proxy is connected with two legs connections which are IPv6 and IPv4 devices. SIP proxy is configured with dual-stack interface. While UAC and UAS devices are configured with IPv6 and IPv4, respectively. Figure 6 shows translation proxy components. OpenSIPs see the website by Bogdan (2020; Goncalves and Iancu, 2016) server is used as a platform for our system that is due to its compatibility, reliability and high connectivity using dual-stack environment. The system utilises OpenSIPs as a proxy server by dealing with a logic route and handled with routing script. For better route tracing OpenSIPs configured as a statfull proxy and use shared cash memory *shmem*, (Khudher et al., 2013). It is worth to mention here

in the technical part that neither RTP proxy nor media proxy are involved in this translation, the translation only occur during signalling session, (Munther et al., 2019).

SIPp is a *de facto* standard tool implemented by Richard (2014) utilised to generate SIP traffic. SIPp clients are used as UAC and UAS in separated devices. SIPp is used to generate SIP traffic from source UAC toward distention UAS also it has ability of message modification and high customised XML language. SIPp is chosen due to its ability of handling high traffic and easy to deal with IPv6 format. Clients are setup under Linux machines with the same network. Table 1 provide hardware specifications for all devices involved in the implementation which are OpenSIPs proxy under Linux machine (Ubuntu), UAC-SIPp machine (Centos) and UAS-SIPp machine (Debian)

6 Evaluation

To validate the proposed SPIPT solution several comparisons and discussions have been conducted in this paper. The solution is compared and validated based solutions on most significant factors. The proposed system shall be evaluated into two results sectors which are quantitative and qualitative results as discussed at the following subsections. The qualitative results focuses on the results compared to other proposed solutions whereas the quantitative results conduct SIP proxy assessment based on three parameters which are CPU usage, Throughput, Call response time.

6.1 Qualitative results

UAs modifications: For SIP-ALG solution both UAC and UAS do not involve in translation mechanism. In contrast to SIP-ALG, Redirect-solution requires a one IP retrieval from the UAS '302 Moved Temporary' in order to be assigned for reINVITE message. Similarly, CSCF-solution force UAC to utilise IPv4 after '180 Ringing' is received. However, client-based solution, from the name indicates, it required full UAC modification for both signalling and media translation. Regarding the SPIPT solution, no UAs modification required and does not perform any IP preference when DNS function is used such as happy eyeball approach.

SIP server modification: When SIP message received by SIP server, a translation might require modification. Technically, this modification depends on the translation mechanism used. For instance, SIP-ALG requires modifying all SIP messages as well as RTP packets, which labelled higher modification in Table 2. Whereas CSCF-solution involve only INVITE message for its translation operation. Redirect-solution and the SPIPT solution do not perform any kind of message modification rather just use the standard response messages which are '302 Move Temporary' and '480 Temporary Unavailable' to involve with minimum modification. Client-based solution does not involve the server in the translation.

Table 2 SPIPT solution versus other solutions

Translation method	*UAs modified*	*SIP server modified*	*Exchanging message complexity*	*RTP Transmission*
ALG	No	High	8 with 3 Transactions	Yes
CSCF	Yes	Medium	8 with 2 Transactions	No
Redirect	Yes	Low	11 with 1 transaction	No
Client	Yes	Low	8 with 1 transaction	No
SPIPT	No	Low	8 with 1 transaction	No

Exchanging message occurrence: ALG-solution performs its translation with 8 exchanging but these messages are broken into 3 transactions. In contrast, CSCF-solution requires 8 messages with one extra transaction. Break a transaction is a degrade behaviour in SIP communication which leads to miss-trace calls path and more difficulty in load balancing and rate-calculation. Redirect and client-based solution are using 11 and 8 messages, respectively. It is worth mentioning here that the call in these two solutions done with one transaction. The SPIPT solution accomplished without broken transaction occur, it only designed with extra 1 response which is '480 Temporary Unavailable' over the standard exchanging message.

RTP transmission: ALG require to open a new port number for each new RTP connection which in turn increase the media transmission latency and also breaks end-to-end connection sequence. While the other solutions including the proposed SPIPT solution do not require any RTP involvement due to the early translation occur in INVITE message. In sum, it requires no modification or translation at media level, the translation mechanisms take place at the signalling part only, (Khudher and Ramadass, 2015).

6.2 Quantitative results

The workload in this research is generated from the SIPp client toward the SIP server. Dramatically, the workload calls are increased starting from 10 calls per second (cps) up to the desired loads (100, 200, 300, 400 and 500). The evaluation is conducted based on two scenarios; Translation (IPv6 to IPv4) and No-translation (IPv6 to IPv6) proxy by evaluating the following parameters:

- CPU utilisation (using one processor and disabling other cores).
- Throughput (amount of received data per ms).
- Call response time (period of round trip SIP message).

Results are obtained from the average of three consecutive test runs. Also, a one test runs for 500 calls in prior of actual evaluation to ramp up the server machine.

Figure 8 shows the CPU usage percentage of the proxy with translation and without translation. Clearly we can notice that the there is a negligible amount of CPU usage when using SPIPT system compared to No-translation scenario to hit around 92% extra when 100 calls load is applied. Regarding 200 calls, No-translation scenario does not record any extra CPU load whereas SPIPT system cost the proxy 2.11% of CPU usage. Interestingly, the SPIPT system will cause the proxy to consume more CPU usage as the number of calls increase. For instance, when the 500 calls is applied, the CPU usage will differ in about 1.5% of No-translation. However, the extra CPU usage is still with normal case with no overload caused that is due to low operation involved in the proposed system and the powerful OpenSIPS proxy chosen as well.

Figure 9 shows the maximum throughput that sustained while successfully handling more than 99.99% of all call requests. There is a lower amount of data received by OpenSIPS proxy when SPIPT system is applied, for example in 100 calls rate the proxy receives 1.9 Kbps of data less than No-translation scenario. This is due to the extra process paid for IPv6 translation, which cause the proxy to slow the throughput rate at an end party. At the next 200 calls, the proxy starts to reduce the difference between the two scenarios to record only 0.7 Kbps throughput. Interestingly, when high load traffic applied, the SPIPT system gain a reasonable amount of data to receive compared to No-translation scenario, which is only 1.66 Kbps. It is worth to mention here that the proposed system is still with acceptable of throughput rate especially when no SIP message lost in call transactions.

Figure 8 CPU usages vs. call rate for translation and no-translation scenarios

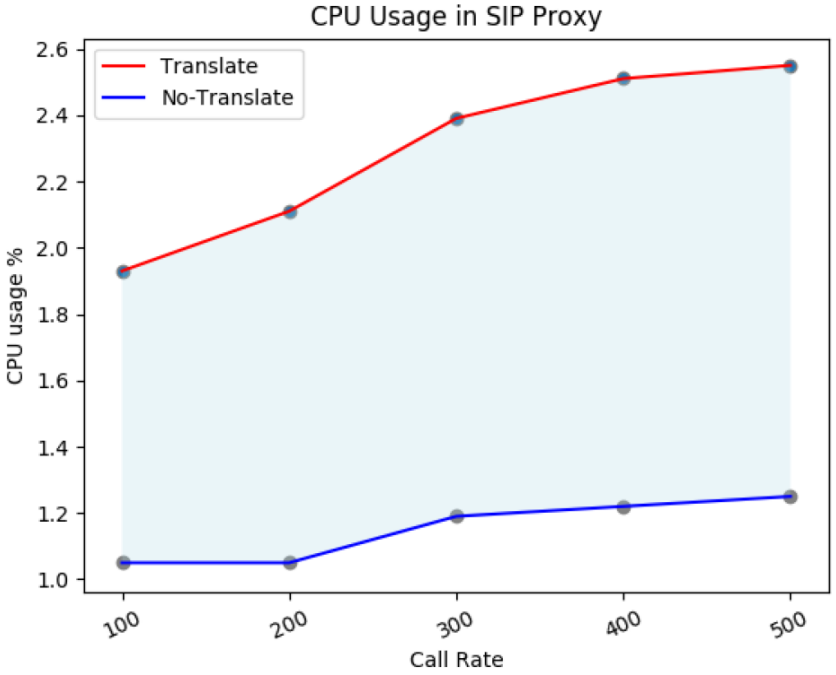


Figure 9 Peak throughputs vs. call rate for translation and no-translation scenarios

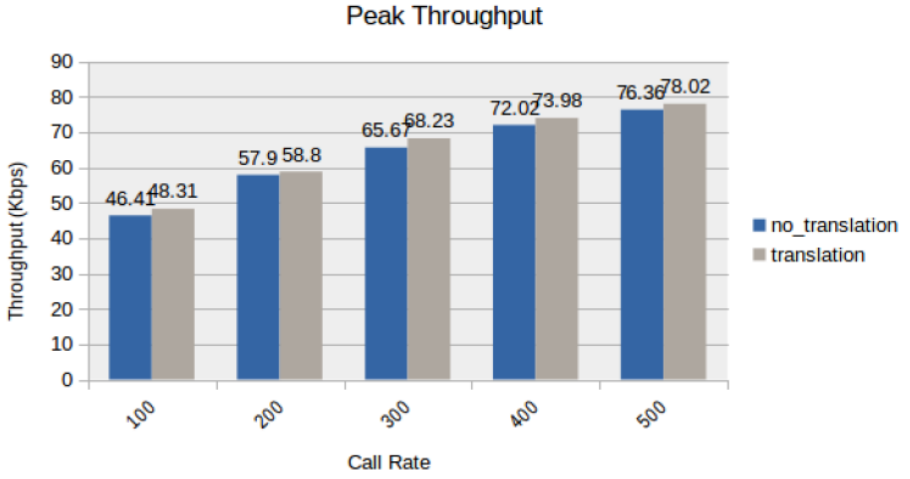
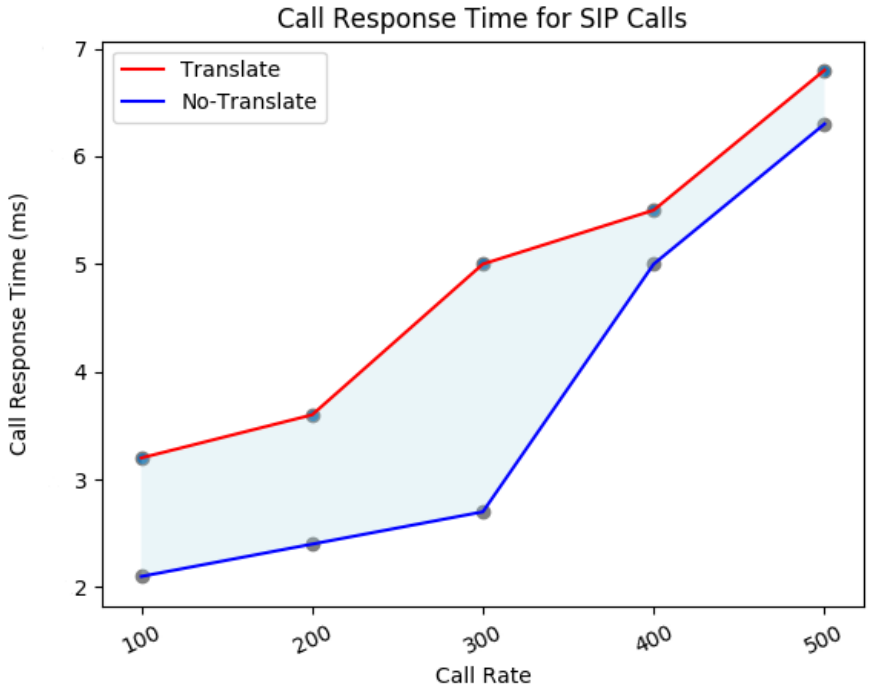


Figure 10 Call response time vs. call rate for translation and no-translation scenarios



Response time is calculated through the period between the sent INVITE and successful 200 OK in millisecond. Actually, it calculates the round trip time of the signalling session setup (first transaction). The methodology of the proposed system shows the translation process is occurred at the early stage of the call signalling session which is the INVITE message, which is made the reason of using the response time parameter. From Figure 10, we can clearly notice that for 100 call rate, SPIPT system scenario spends average of 0.032 ms to set up signalling session, then it increased dramatically as load increase, to hit 0.068 ms at 500 load which is within the acceptable response time (0.20 ms) of standared OpenSIPS proxy, (Rosenberg et al., 2002). It can be confirmed that SPIPT system impacts directly by the number of end-users. That is because the proposed system requires extra time to translate IPv6 to IPv4 address. Fortunately, the translation process in the proposed system is only occurring once for one call transaction. Therefore, it does not introduce a drawback into SPIPT system lifecycle.

7 Conclusion

Potentially, IMS clients (UEs) are located in different networks. For instance, caller resides in IPv6 network while called party in IPv4. SIP server unable to handle calls between different networks, resulting the call established incorrectly or fail at all. SPIPT solution is proposed in this paper to translate from IP version to another. The method behind this solution is to examine the end party using negative response right after the initial INVITE request. Generally, the SPIPT is a solution for four cases of networks scenarios which are IPv4-to-IPv4, IPv4-to-IPv6, IPv6-to-IPv6 and IPv6-to-IPv4. However, the latter one considered the serious problem in network coexistence especially with ISPs which are witnessing an IPv6 migration in certain country/region. It is good to mention here that the rest cases are still considered as a problem but with less impact to SIP communication due to other solutions involvements and the advantage of dual-stack. SPIPT solution implemented in a SIP proxy (OpenSIPs) configured with dual-stack IP to be able to communicate with two legs of connections. The proposed solution insures better call setup Occurrence against the four solutions in the literature. For the UA modification both ALG and the SPIPT solutions do not modify UA in both sides. Whereas SIP server modification are occur in the four solution in related works ranging from low to high. SPIPT solution does not perform any SIP server or database modification during call setup. In contrast to other solutions, SPIPT accomplish the call with one transaction and 8 exchanging messages. Obviously, SPIPT solution stays with the call track without any transaction broken. Finally, SPIPT solution increases user experience while using SIP call with other network family which in turn accelerates IPv6 migration. Also, the solution is recommended for SIP service providers whose customers are witnessing IPv6 transition in their country. Experimentally speaking, SIP server registers negligible amount of CPU usage when using SPIPT system compared to No-translation scenario to hit around 92% extra when 100 calls load with no overload caused. In addition, throughput also stay with the a valued level by recording 1.9 Kbps of data less than No-translation scenario when 100 calls load applied. The validation of the proposed system also accomplished by the amount of Call Response Time parameter to reach 0.068 ms at 500 load which is within the acceptable response time (0.20 ms).

References

Bertrand, G. (2007) *The IP Multimedia Subsystem in Next Generation Networks*, 1st ed., Rapport Technique.

Bogdan, A. (2020) *OpenSIPS the new breed of communication engine*. Available online at: https://opensips.org/ (accessed on 8 August 2020).

Camarillo, G. and García-Martín, M.A. (2008) *The 3G IP Multimedia Subsystem IMS: Merging Internet Cell*, 3rd ed., John Wiley and Sons, London.

Chen, W.E., Wu, Q., Lin, T.B. and Lo, Y.C. (2004) 'Design of SIP application level gateway for IPv6 translation', *Journal of Internet Technology*, Vol. 5, No. 2, pp.147–154.

Chen, W-E. and Li, S-H. (2013) 'Client-based internet protocol version 4-internet protocol version 6 translation mechanism for session initiation protocol multimedia services in next generation networks', *IET Networks*, Vol. 2, No. 3, pp.115–123.

Chen, W-E., Huang, Y-L. and Lin, Y-B. (2010) 'An effective IPv4-IPv6 translation mechanism for SIP applications in next generation networks', *International Journal of Communication Systems*, Vol. 23, No. 8, pp.919–928.

Chen, W-E., Lin, Y-B. and Pang, A-C. (2005) 'An IPv4-IPv6 translation mechanism for SIP overlay network in UMTS all-IP environment', *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 11, pp.2152–2160.

Cui, Y., Wu, P., Xu, M., Wu, J., Lee, Y., Durand, A. and Metz, C. (2012) '4over6: network layer virtualization for IPv4-IPv6 coexistence', *IEEE Network*, Vol. 26, No. 5, pp.44–48.

Davies, J. (2012) *Understanding IPv6*, 1st ed., Pearson Education, UK.

Goncalves, F.E. and Iancu, B.A. (2016) *Building Telephony Systems with OpenSIPS*, 2nd ed., Packt Publishing Ltd. Birmingham, UK.

Gurbani, V., Boulton, C. and Sparks, R. (2008) *Session Initiation Protocol (SIP) Torture Test Messages for Internet Protocol Version 6 (IPv6)*, No. 5118, RFC Editor. Available online at: https://tools.ietf.org/html/rfc5118 (accessed on 1 April 2020).

Han, J.C., Hyun, W., Park, S.O., Lee, I.J., Huh, M.Y. and Kang, S.G. (2006) 'An application level gateway for traversal of SIP transaction through NATs', *Proceedings of the IEEE 8th International Conference Advanced Communication Technology*, IEEE Phoenix Park, South Korea, pp.1648– 1652.

Handley, M., Jacobson, V. and Perkins, C. (2006) *SDP: Session Description Protocol*, No. 4566, RFC Editor. Available online at: https://doi.org/10.17487/rfc4566 (accessed on 2 April 2020).

Hoeher, T., Petraschek, M., Tomic, S. and Hirschbichler, M. (2007) 'Evaluating performance characteristics of SIP over IPv6', *Journal of Networks*, Vol. 2, No. 4, pp.40–50.

Jabir, A.J. (2020) 'A low cost paging scheme for clustered PMIPv6 protocol by head-MAG entity utilisation', *International Journal of Internet Protocol Technology*, Vol. 13 No.1, pp.18–24.

Johnston, A., Donovan, S., Sparks, R., Cunningham, C. and Summers, K. (2003) *Session Initiation Protocol (SIP) Basic Call Flow Examples*, No. 3665, RFC Editor. Available online at: https://tools.ietf.org/html/rfc3665 (accessed on 1 April 2020).

Johnston, A.B. (2015) *SIP: Understanding the Session Initiation Protocol*, No. 5631, RFC Editor. Available online at: https://tools.ietf.org/html/rfc5631 (accessed on 12 May 2020).

Khudher, A.A. (2019) 'SIP aspects of IPv6 transitions: current issues and future directions', *Journal of Engineering Science and Technology*, Vol. 14, No. 1, pp.448–463.

Khudher, A.A. and Ramadass, S. (2015) 'I-TNT: phone number expansion and translation system for managing interconnectivity addressing in sip peering', *Journal of Engineering Science and Technology*, Vol. 10, No. 2, pp.174–183.

Khudher, A.A., Aboalmaaly, M.F. and Naeem, A.N. (2013) 'Telephone number addressing for SIP peering within inter-domain voice communication', *Advances in Information Sciences and Service Sciences*, Vol. 5, No. 11, pp.178–186.

Khudher, A.A., Beng, L.Y. and Ramadass, S. (2013) 'A comparative study of direct and indirect static peering for inter-domain sip calls', *Proceedings of the IEEE International Conference RFID-Technologies Application*, Johor Bahru, Malaysia, pp.1–5.

Kim, P.S., Lee, M.E., Park, S. and Kim, Y.K. (2004) 'A new mechanism for SIP over mobile IPv6', *Proceedings of the International Conference Computational Science and its Applications*, Springer, Berlin, Heidelberg, pp.975–984.

Minoli, D. (2011) *Voice over IPv6: Architectures for next Generation VoIP Networks*, 1st ed., Newnes, London.

Munther, A., Mohammed, I.J., Anbar, M. and Hilal, A.M. (2019) 'Performance evaluation for four supervised classifiers in internet traffic classification', *Proceedings of the International Conference on Advances in Cyber Security*, Springer, Singapore, pp.168–181.

Praptodiyono, S., Santoso, M.I., Firmansyah, T., Abdurrazaq, A., Hasbullah, I.H. and Osman, A. (2019) 'Enhancing IPsec performance in mobile IPv6 using elliptic curve cryptography', *Proceedings of the 6th International Conference on Electrical Engineering, Computer Science and Informatics*, Bandung, Indonesia, pp.186–191.

Richard, G. (2014) *Welcome to SIPp.* Available online at: http://sipp.sourceforge.net/ (accessed on 28 April 2020).

Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R. and Handley, M. (2002) *SIP: Session Initiation Protocol*, No. 3261, RFC Editor. Available online at: https://tools.ietf.org/html/rfc3261 (accessed on 29 March 2020).

Schinazi, D. and Pauly, T. (2017) *Happy Eyeballs Version 2: Better Connectivity Using Concurrency*, No. 8305, RFC Editor. Available online at: https://tools.ietf.org/html/rfc8305 (accessed on 29 March 2020).

Schulzrinne, H., Casner, S., Frederick, R. and Jacobson, V. (2003) *RTP a Transport Protocol for Real-Time Applications*. No. 3550, RFC Editor. Available online at: https://tools.ietf.org/html/rfc3550 (accessed on 10 April 2020).

Shacham, R., Schulzrinne, H., Thakolsri, S. and Kellerer, W. (2009) *Session Initiation Protocol (SIP) Session Mobility*, No. 6531, RFC Editor. Available online at: https://tools.ietf.org/html/rfc6531 (accessed on 29 March 2020).

Sisalem, D., Fiedler, J. and Ruppelt, R. (2003) 'SIP and IPv6: why and how?', *Proceedings of the Symposium on Applications and the Internet Workshops*, IEEE Orlando, Florida, USA, pp.222–225.

Vemuri, A. and Peterson, J. (2002) *Session Initiation Protocol for Telephones (SIP-T): Context and Architectures*, No. 3372, RFC Editor. Available online at: https://doi.org/10.17487/RFC3372 (accessed on 29 March 2020).

Zourzouvillys, T. and Rescorla, E. (2010) 'An introduction to standards-based VoIP: SIP, RTP, and friends', *IEEE Internet Computing*, Vol. 14, No. 2, pp.69–73.