Reducing Multimedia Presence Information Traffic between Mobile Applications

Dongcheul Lee

Department of Electronics Computer Engineering,
Hanyang University, Seoul, Korea
jackdclee@hanyang.ac.kr

Abstract

Presence enhances multimedia services of the mobile applications by making someone's state visible to others. The main driver for Presence has been instant messaging. However, in the mobile environment, many mobile services have used Presence including voice calls, video calls, file transferring, gaming, and other multimedia services. Open Mobile Alliance proposed Presence architecture which utilizes SIP SUBSCRIBE, NOTIFY, and PUBLISH method. However, this architecture is not suitable for simple mobile applications since it is complex to implement. Most simple mobile applications use SIP OPTIONS message to query others Presence information. However, this scheme uses much signaling message when there are many targets to query. This paper presents a simple and efficient Presence model by using OPTIONS aggregation server. The server receives list-based OPTIONS message and aggregate multiple Presence information into one message. The message does not have to be transferred to the target devices since the server utilizes the information in XDMS.

Keywords: *mobile application, multimedia presence, IP multimedia subsystems*

1. Introduction

Since smart phones were introduced to the world, Presence is making communications more conveniently [1]. Presence enhances messaging and telephony services of the mobile applications by making someone's state visible to others. Presence information may include availability, communication preferences, service capabilities, social information, portrait, and location. The main driver for Presence has been instant messaging. However, in the mobile environment, many mobile applications have used Presence including voice calls, video calls, file transferring, gaming, and other multimedia services.

There are two kinds of mobile applications which need Presence. One requires near-real-time Presence information and the other requires long-polling-based Presence information. First application provides services that the status can be changed often, such as availability, location. Since the application should be notified others Presence information in near-real-time, the application should maintain an active session with a Presence server, which requires much network resources. Second application provides services that the status hardly be changed, such as communication preferences, service capabilities, social information, and portrait. Since the application polls the Presence server with long period, it can reduce the use of network resources.

In this paper, simple and efficient Presence information polling architecture is introduced. It is simple to implement since it uses SIP OPTIONS message which is simpler than SIP SUBSCRIBE method [2]. Also, the paper propose list-based OPTIONS message which reduces signaling traffic. The call flow of the proposed architecture is described and sample messages of list-based OPTIONS are shown.

2. Related Works

Presence service makes my status available to others and others' statuses available to me. There are many kinds of Presence information including availability, communication preferences, service capabilities, social information, portrait, and location. There are two approaches to enable the Presence service in IP Multimedia Subsystems (IMS) [3]. One approach is making use of SIP SUBCRIBE and NOTIFY method, which is called Open Mobile Alliance (OMA) Presence Session Initiation Protocol (SIP) for Instant Messaging and Presence Leveraging Extensions (SIMPLE) [4]. This approach is suitable for near-real-time services that the status can be changed often, such as availability, location. Another approach is using SIP OPTIONS method. This approach is suitable for non-real-time services that the status hardly be changed, such as communication preferences, service capabilities, social information, and portrait.

2.1. OMA Presence SIMPLE

SIP was extended for the Presence service by creating Presence event package. When a user subscribes to the event, the subscriber inserts the Presence token in the Event header. Figure 1 shows the architecture of OMA Presence SIMPLE. The Presence server accepts stores and distributes Presence information published by Presence sources. The Presence source provides Presence information to a Presence server on behalf of the Presentity. The Watcher requests and subscribes Presence information about Presentity from the Presence server. The Resource List Server (RLS) accepts and manages subscriptions to Presence Lists, which enables the Watcher to subscribe to multiple Presence information using a single subscription request. The Presence XML Document Management Server (XDMS) manages XML documents, such as Presence authorization rules, which are related to the use of Presence service [5]. The Resource List Server XML Document Management Server (RLS XDMS) manages XML documents, such as Presence lists, which are related to the use of the RLS. The SIP core network performs SIP signaling routing, authentication of the Presence functional entities, SIP compression services in support of Presence service. The aggregation Proxy is the contact point for the XDM client to access XML documents in XDMS. It also performs authentication of the XDM client and routes XML Configuration Access Protocol (XCAP) requests to the suitable XDMS [6]. The Shared XDMS stores XML documents which can be reused by other enablers.

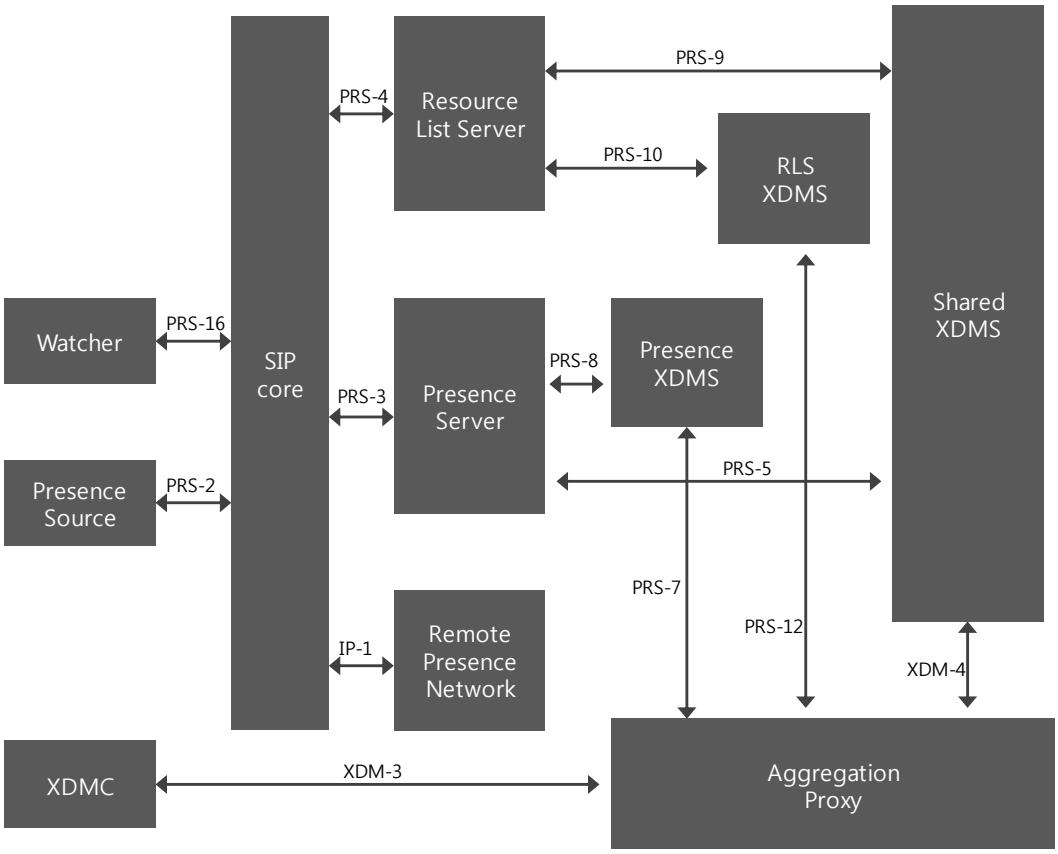

Figure 1. OMA Presence SIMPLE Architecture

Figure 2 shows the call flow of the OMA presence SIMPLE and detail description of each message is as follows.

(1) The Presence Source sends D's Presence information using SIP PUBLISH message with a Request URI D@domain.com.

(2) The Presence Server sends 200 OK message to the Presence Source.

(3) The Presence Server composes presence state for D@domain.com and applies filtering.

(4) The Watcher sends SIP SUBSCRIBE message with a Request URI D@domain.com

(5) The Presence Server sends 202 Success to the Watcher.

(6) The Presence Server sends SIP SUBSCRIBE message to XDMS for requesting policy document of D@domain.com.

(7) The XDMS sends 202 Success to the Presence Server.

(8) The XDMS sends SIP NOTIFY message to the Presence Server which contains the URL of current version of D's Presence policy document.

(9) The Presence Server sends 200 OK to the XDMS for the NOTIFY message.

(10) The Presence Server sends XCAP GET message to the URL.

(11) The XDMS sends the policy document to the Presence Server.

(12) The Presence Server applies the policy rule.

(13) The Presence Server sends SIP NOTIFY message which contains D's Presence information to the Watcher.

(14) The Watcher sends 200 OK message for the NOTIFY message.

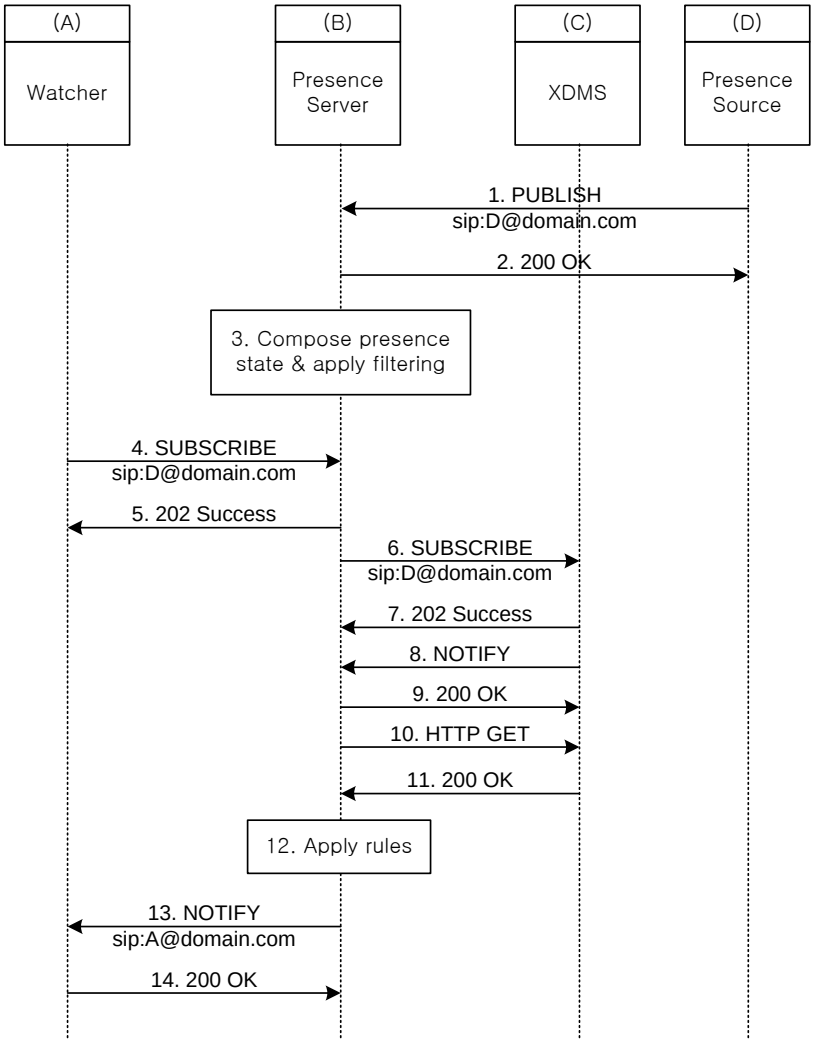

Figure 2. Call Flow of OMA presence SIMPLE

2.2. SIP OPTIONS

OMA presence SIMPLE is suitable when the Watcher want to know the status of Presence source in near-real-time. When the status is changed, the Watcher can be notified the information in near-real-time. However this architecture consumes much network resources since the Watcher and the Presence Server should maintain an active

session to send the NOTIFY message. If the Watcher is a mobile terminal, this can be serious problem since the capacity of a mobile network is usually not enough. Therefore, if the Presence information hardly changes, and if the Watcher does not have to be notified in near-real-time, the Watcher uses fetch-SUBSCRIBE which polls the Presence server and does not notified after the polls. Even though OMA presence SIMPLE provides fetch-SUBSCRIBE, there can be simple mobile applications that do not support the architecture. Those applications use only SIP OPTIONS which is very simple to implement to fetch others Presence information. SIP OPTIONS message to request the Presence information. OPTIONS message is sent to the Presence Source directly and each message is sent separately.

Figure 3 show the call flow of sending SIP OPTIONS message and detail description is as follows.

(1) User A wants to know the Presence information of User B, so A sends SIP OPTIONS message with Request URI sip:B@domain.com.

(2) B sends 200 OK message to A with a Presence information of B.

(3) User A wants to know the Presence information of User C, so A sends SIP OPTIONS message with Request URI sip:C@domain.com.

(4) C sends 200 OK message to A with a Presence information of C.

(5) User A wants to know the Presence information of User D, so A sends SIP OPTIONS message with Request URI sip:D@domain.com.

(6) D sends 200 OK message to A with a Presence information of D.

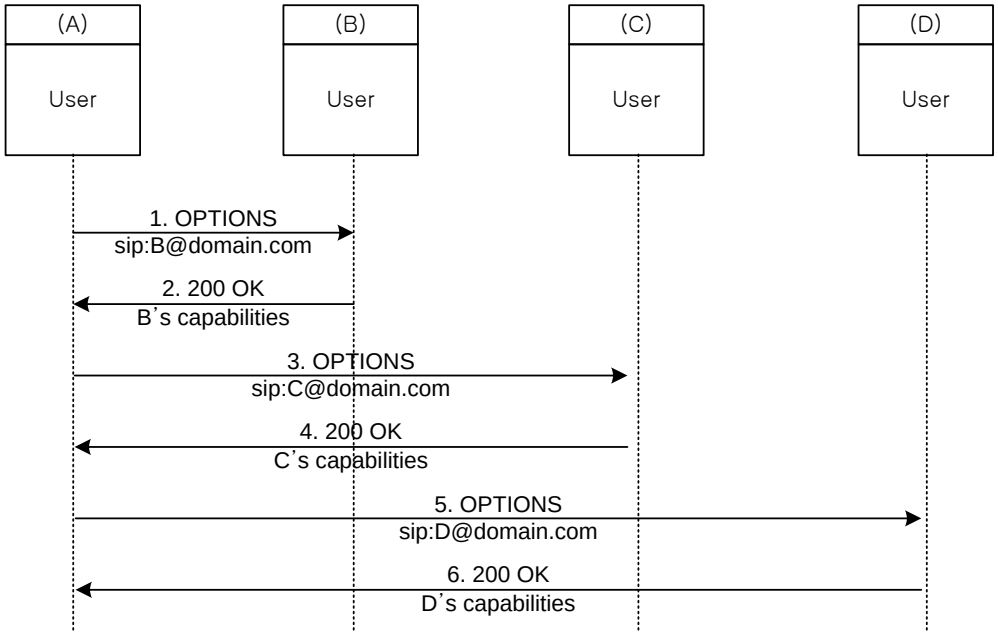

Figure 3. Call Flow of Sending OPTIONS Message

3. Proposed Architecture

Even though sending SIP OPTIONS message can be useful if the Presence information hardly changes, and if the Watcher does not have to be notified in near-real-time, it can still cause much traffic in the core network if the polling period is short and the Watcher want to know the status of many users. If the Watcher wants to request *n* status of others, the Watcher should send *n* SIP OPTIONS messages to the core network, and the core network should relay *n* SIP OPTIONS messages to the users. Also, *n* 200 OK messages are generated and forwarded to the Watcher.

Figure 4 shows the proposed architecture. It introduces OPTIONS Aggregation Server (OAS) which can receive list-based OPTIONS message from the Watcher and aggregate *n* 200 OK messages into 1 200 OK message. In this architecture, the Presentity sends XCAP PUT message to XDMS which stores the information in Presence Information Data Format (PIDF). Then OAS requests the PIDF from the XDMS for each Presentity.

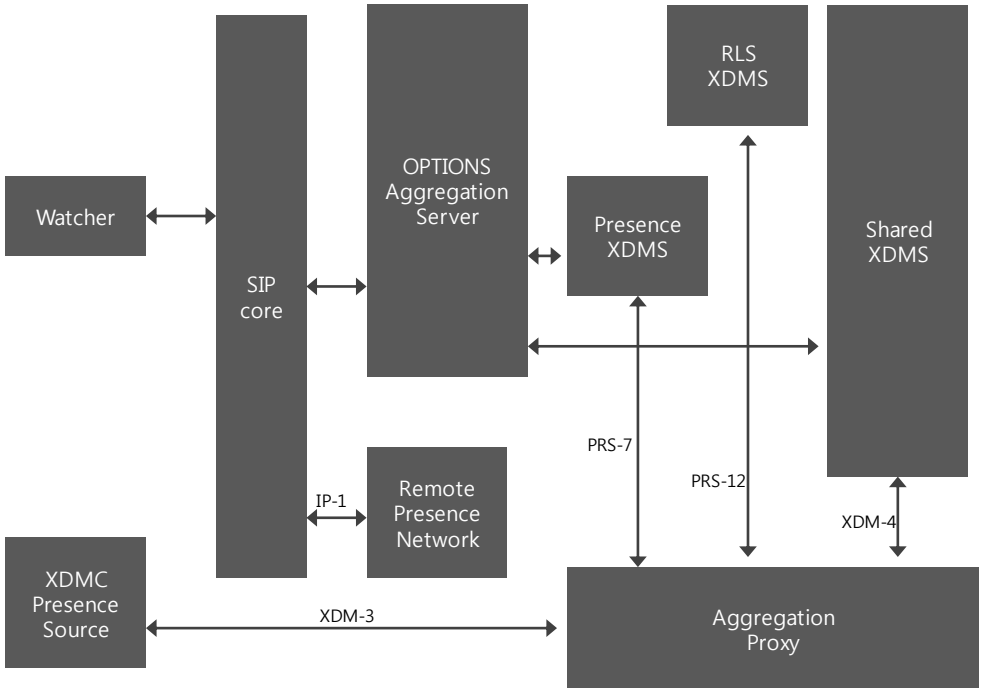

Figure 4. Architecture of the Proposed Presence Framework

Figure 8 show the call flow of the proposed architecture and detail description of the flow is as follows.

(1) The Presentity D sends a XCAP PUT message to deliver D's Presence information. Figure 5 shows the example message.

```
PUT /pidf-manipulation/users/sip:D@domain.com/index HTTP/1.1
Content-Type: application/pidf+xml
<?xml version="1.0" encoding="UTF-8"?>
```

```
<presence
xmlns="urn:ietf:params:xml:ns:pidf"
xmlns:c="urn:ietf:params:xml:ns:pidf:cipid"
xmlns:op="urn:oma:xml:prs:pidf:oma-pres"
xmlns:opd="urn:oma:xml:pde:pidf:ext"
xmlns:pdm="urn:ietf:params:xml:ns:pidf:data-model"
xmlns:rpid="urn:ietf:params:xml:ns:pidf:rpid"
xmlns:oas="urn:oas:params:xml:prs:pidf:oas"
 entity="sip:D@domain.com">
 <pdm:person id="p1">
  <oas:e-mail>mailto:username@mail-domain-name</oas:e-mail>
  <oas:birthday>01-01</oas:birthday>
  <oas:capability>
+g.3gpp.iari.ref="urn%3Aurn.7%3A3gpp.application.ims.iari.rcse.im";
+g.3gpp.iari.ref="urn%3Aurn.7%3A3gpp.application.ims.iari.rcse.ft";
+g.3gpp.iari.ref="urn%3Aurn.7%3A3gpp.application.ims.iari.rcse.sp"
</oas:capability>
 </pdm:person>
</presence>
```

Figure 5. Example of XCAP PUT Message

(2) The XDMS sends 200 OK to the Presentity in response to the PUT message.

(3) The Presentity E sends a XCAP PUT message to deliver E's Presence information.

(4) The XDMS sends 200 OK to the Presentity in response to the PUT message.

(5) The Watcher sends a list-based SIP OPTIONS message to the OAS to request the Presence information of D and E. Figure 6 shows the example of the list-based OPTIONS message.

```
OPTIONS sip:conf-fact@domain.com SIP/2.0
Via: SIP/2.0/UDP core-ip-addr:port;branch=z9hG4bK-1
To: sip:D@domain.com
From: sip:A@domain.com;tag=from1
Accept-Contact:
*;+g.3gpp.iari.ref="urn%3Aurn.7%3A3gpp.application.ims.iari.rcse.im";
+g.3gpp.iari.ref="urn%3Aurn.7%3A3gpp.application.ims.iari.rcse.ft";
```

```
+g.3gpp.iari.ref="urn%3Aurn.7%3A3gpp.application.ims.iari.rcse.sp
Content-type: application/resource-lists+xml
Cotent-Disposition: recipient-list
<?xml version=”1.0” encoding=”UTF-8”?>
<resource-lists xmlns=”urn:ietf:params:xml:ns:resource-lists”
xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance”>
<list>
<entry uri="sip:D@domain.com"/>
<entry uri="sip:E@domain.com"/>
</list>
</resource-lists>
```

Figure 6. Example of list-based OPTIONS Message

(6) The OAS sends XCAP GET message to the XDMS to retrieve D’s Presence information

(7) The XDMS sends 200 OK message, which contains D’s Presence information, to the OAS

(8) The OAS sends XCAP GET message to the XDMS to retrieve E’s Presence information

(9) The XDMS sends 200 OK message, which contains E’s Presence information, to the OAS

(10) The OAS aggregate the two 200 OK messages into one 200 OK message, which contains both Presence information of D and E, and send it to the Watcher. Figure 7 show the example of aggregated message.

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP oas-ip-addr:port;branch=z9hG4bK-1
To: sip:D@domain.com
From: sip:A@domain.com;tag=from1
Content-type: application/resource-lists+xml
<?xml version=”1.0” encoding=”UTF-8”?>
<resource-lists xmlns=”urn:ietf:params:xml:ns:resource-lists”
xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance”>
<list>
<entry uri="sip:D@domain.com">
```

```
<capa>+g.3gpp.iari-ref=”urn%3Aurn-7%3A3gpp-application.ims.iari.rcse.im”;
+g.3gpp.iari-ref="urn%3Aurn-7%3A3gpp-application.ims.iari.rcse.ft;
+g.3gpp.iari-ref="urn%3Aurn-7%3A3gpp-application.ims.iari.rcse.dp";
+g.3gpp.iari-ref="urn%3Aurn-7%3A3gpp-application.ims.iari.rcse.sp"</capa></entry>
<entry uri="sip:E@domain.com">
<capa>>+g.3gpp.iari-ref=”urn%3Aurn-7%3A3gpp-application.ims.iari.rcse.im”;
+g.3gpp.iari-ref="urn%3Aurn-7%3A3gpp-application.ims.iari.rcse.ft;
+g.3gpp.iari-ref="urn%3Aurn-7%3A3gpp-application.ims.iari.rcse.dp";
+g.3gpp.iari-ref="urn%3Aurn-7%3A3gpp-application.ims.iari.rcse.sp"</capa></entry>
</list>
</resource-lists>
```

Figure 7. Example of Aggregated 200 OK Message

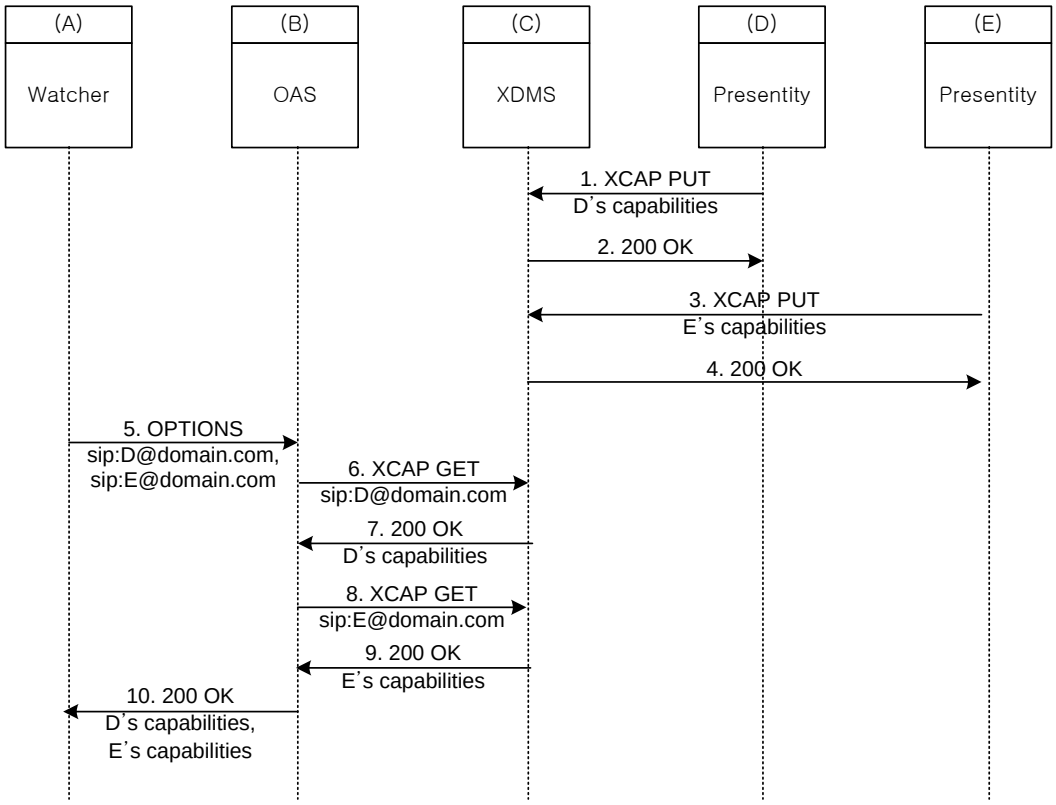

Figure 8. Call Flow of the Proposed Architecture

4. Conclusions

This paper introduced a simple and efficient Presence information polling architecture. It is simple to implement since it uses SIP OPTIONS message which is simpler than OMA presence SIMPLE architecture. Also, the paper propose list-based OPTIONS message which reduces signaling traffic. The architecture and call flow of the proposed scheme is described and example messages of list-based OPTIONS are shown.

References

[1] M. Y. Huh, W. Hyun, J. C. Han, I. J. Lee and S. G. Kang, "Design considerations for user authorization in the presence services based on SIP", Proceedings of the 7th International Conference on Advanced Communication Technology, **(2005).**

[2] A. Rios, J. Alcober, A. Oller and V. Sempere, "Design and implementation of a measurement and alert system of QoS parameters in SIP based Internet telephony", Proceedings of the 2nd Conference on Next Generation Internet Design and Engineering, **(2006).**

[3] B. Panwar and K. Singh, "IMS SIP core server test bed", Proceedings of the International Conference on IP Multimedia Subsystem Architecture and Applications, **(2007).**

[4] R. Karunamurthy, R. H. Glitho and F. Khendek, "A novel Web service for presence and its implementation in an IETF SIMPLE protocol environment", Proceedings of the IEEE International Conference on Web Services, **(2005).**

[5] L. Prati, S. Lim and N. Aschoff, "XDMS-Network Address Book enabler", Proceedings of the international Conference on IP Multimedia Subsystem Architecture and Applications, **(2007).**

[6] I. J. Lee, S. O. Park, W. H. Shin and G. Kang, "A study on buddy list control in XCAP server system for SIP-based IMPP services", Proceedings of the 7th International Conference on Advanced Communication Technology, **(2005).**