Implementation of Ad-Hoc VoIP Using UPnP

Lin-Huang Chang[1], P.D. Chuang[2]

Department of Computer and Information Science,
National Taichung University, Taichung, Taiwan
lchang@mail.ntcu.edu.tw

Abstract. The VoIP is a popular Internet application nowadays which is expected to be carried on at anytime and anywhere. However, the re-configuration problem as well as signaling and transmission delay will be major concerns for a mobile node moving into a foreign network. In this paper, we design and implement an ad-hoc VoIP architecture using UPnP protocol to provide a configuration free and free access of VoIP communication under ad-hoc network. The innovation of proposed pseudo SIP server with UPnP solves the problem with no standalone SIP server on Ad-Hoc network. The performance evaluation of the proposed mechanism demonstrates the realization of ad-hoc VoIP with free configuration and access.

Keywords: SIP, VoIP, Ad-Hoc, UPnP, pseudo SIP server

1 Introduction

The applications of voice over Internet protocol (VoIP) mostly rely on the client/server architecture. Currently, the implementation of VoIP system mostly employs Session Initiation Protocol (SIP) [1] for its signaling which is also based on the client/server architecture. On the other hand, the free installation and without fixed infrastructure of the Ad-hoc network does provide some advantage in the networking realization. The client/server architecture however is complicated to deploy on Ad-hoc network environment.

In this paper, we design an Ad-hoc VoIP architecture implementing with UPnP protocol which provides a configuration free and free access of VoIP communication under Ad-Hoc network. When a user enters such network, it will obtain all devices information of this network via UPnP, such as IP address and user corresponding name. Relying on the Plug and Play concept, every device's information will be transmitted to those with UPnP support by multicasting and vice versa. The proposed scheme allows the UA to acquire the remote user information automatically and store it into the proposed pseudo SIP server. The innovation of the proposed pseudo SIP server solves the problem with no standalone SIP server on Ad-Hoc network. Also, the IPv6 addressing with auto-configuration makes this system easier on network connectivity.

The rest of this paper is organized as follows. We review related works in section II. In section III we describe the system architecture and design of Ad-hoc VoIP using

SIP and UPnP followed by the implementation and performance analysis presented in section IV. Finally we address the conclusion in section V.

2 Related Works

In the former studies of SIP application on the Ad-Hoc Network [2-4], the authors proposed a modification of SIP protocol to support SIP signaling as well as multi-party voice conference ability under Ad-Hoc Network environment in which standalone SIP server was absent. This framework presented some mechanism with modified SIP protocol as well as SIP extension [5]. It proposed a new field in SIP message, called “Conf-Id”, to identify participants in the conference calls. The “Conf-Id” field contains unique id that comes with the SIP User Agent (UA) by computing the UA hostname as well as the Internet Protocol (IP) address. The host of each conference, as known as leader, uses the “Conf-Id” to allow participants to identify the specific conference and its corresponding host among multiple conferences. On the other hand, a new user sends the “INVITE” message with the corresponding “Conf-Id” value when it requests one host to join its conference. Then, the host will send the “CONF” message, which is another SIP modification in extension, to announce this new member to all conference participants.

This framework however has some limitations since it relies on broadcasting for message exchange between nodes. This results in inefficient resource utilization and might not be scalable under a crowded network environment. Also, the proposed SIP modifications may not be suitable for the existing SIP UA software. Therefore, extra effort needs to be done to support this extension which will be a problem for deployment.

The Universal Plug and Play architecture (UPnP) [6] was constituted by UPnP forum, which lead by Microsoft. UPnP, however, was designed to Internet appliance (IA) and automata controller solution on networking devices which provided peer to peer as well as Ad-Hoc connectivity between devices pervasively with simple open standard. The UPnP defines six phases: Addressing, Discovery, Description, Control, Event and Presentation. The UPnP describes a concept which doesn’t constrain to particular API or operation system. Every device is communicating each other with certain common standard without specific programming language or operation system.

UPnP technology provides a distributed and open network structure over TCP/IP and HTTP architecture. Through the support of the communication among control points, devices and services defined and expressed by Extensible Makeup Language (XML), UPnP technology realizes the intelligent connection to control and transfer data for point-to-point networks. While using UPnP, one can join a network dynamically, obtain an IP address automatically, announce ones facility ability or search the existence devices and services. Completed by the automatic processes, all devices can communicate directly. Since UPnP does not need device drivers, the established network by UPnP is interfacing independent. Also, UPnP uses standard TCP/IP to enable seamless proximity networking with the existing network [7]. Furthermore, UPnP application program can be implemented by any programming language and executed on any operating system. UPnP uses the description of HTML form to describe the controlling interface of the device. It not only allows the device suppliers to offer user's interface that based on browser, but also allows developers to make their own device interfaces.

UPnP architecture is composed of the control points and devices. All devices are connected with distributed and peer-to-peer network structure. It means the communication between control point and device does not go through the third party. UPnP device uses Simple Service Discovery Protocol (SSDP) to discover the target and then uses Simple Object Access Protocol (SOAP) to achieve the remote controlling. The data transmission is completed by using General Event Notification Architecture (GENA). This architecture provides a configuration free network environment for UPnP devices to interact with each other and further connect to the Internet to use the network resources. This will be location, devices and even network topology irrelative. Therefore, in this paper we design and implement an Ad-hoc VoIP architecture using UPnP protocol which provides a configuration free and access free of VoIP communication under Ad-Hoc network.

3 System Design and Implementation

The overview architecture of our proposed mechanism is shown in Fig. 1, where the whole system is based on IPv6 and the application layer consists of UPnP module, pseudo SIP server and SIP user agent (UA). There're two types of flow within this proposed architecture, including SIP message flow and UPnP message flow. The UPnP message flow conducts advertising and discovering remote users' presence among different UPnP modules sitting in each device. The SIP message flow keeps the current SIP UA compatible with our proposed system.

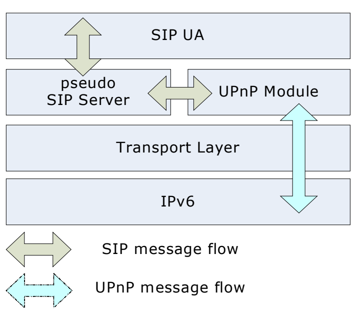

Fig. 1 The overview architecture

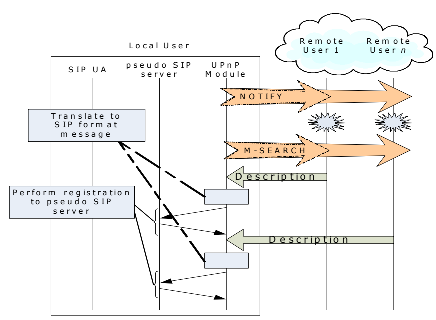

Fig. 2 Signaling flow of the proposed scheme

The proposed innovative pseudo SIP server was designed as tiny as possible. It provided limited functionality then ordinary SIP server, which intend to smaller to embed into user devices. So far as the prototyping of pseudo SIP server, it was design to accept and process limited commands, such as: REGISTER, INVITE, OPTION and some provisional message handling, like ACK, OK. Fig. 2 shows the Signaling flow of the proposed scheme with UPnP multicasted NOTIFY and M-SEARCH commands to remote user. UPnP module will acquired remote user information then translates to SIP REGISTER format then register to pseudo SIP server for further access.

In our proposed scheme, UPnP module is responsible for discovery existing remote user within current network, but user information, however, need to be transferred as useful information to upper SIP user agent. As the interface between upper SIP user agent and UPnP module, pseudo SIP server accepts remote user store by UPnP. Since UPnP discovered and acquire remote user information, it will translate to SIP REGISTER message format with certain data then send to pseudo SIP server.

Fig. 3 shows translated REGISTER message by UPnP discovery module. After the registration to pseudo SIP server, the remote user with network address, network port, contact name and additional information will be registered into pseudo SIP server. The additional information will store in user defined tag “ext” in “From” field of REGISTER message. The pseudo SIP server will parse and process this user define tag for additional information such as user status.

This gathered information by UPnP will be translated into SIP format and registered in our proposed pseudo SIP server as shown in Fig. 4. Then, the SIP UA acquires the remote user lists from the SIP assistant databases that stored on each user’s local storage devices rather than request from particular SIP server.

As the extra application execution on user mobile handheld devices, the proposed pseudo SIP server must keeps limitation on computing and storage ability, the pseudo SIP server must design as tiny as possible, which will not take too much overhead on CPU time usage and memory consumption.

```
REGISTER sip:[::1]:5060 SIP/2.0
Via SIP/2.0/UDP [fe80::202:6fff:fe09:b1da%wi0]:5060
From: “lchang” sip:lchang@[::1]:5060;tag=3ddfe2;ext=000001
To: “lchang” sip: lchang @[::1]:5060
Contact: “lchang” sip: lchang @[fe80::202:6fff:fe09:b1da%wi0]:5060
CSeq: 1 REGISTER
Expires: 3600
User-Agent: UPnP $Revision 1.12$
Content-Length: 0
```

Fig. 3 REGISTER message example

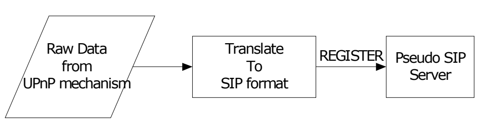

Fig. 4 Data flow of pseudo SIP server

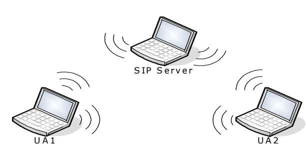

Fig. 5 Experimental topology

4 Performance Analysis

In order to test and evaluate our developed system functionality, we design and modify the testing tool set for pseudo SIP server. The testing tool set contains sipsak and siptest. Sipsak is a well-known open source SIP testing tool, which provided various function on SIP server test such as user data (userloc) manipulation as well as stress test on SIP server. We do some modification on sipsak for allowing access link-local IPv6 remote host to fulfill our requirement and testbed. We also deploy some modification on sipsak for measure the round trip or delay time of each segment within every single call initiation flow, which will be describe in later section.

We had also developed another test tool which designated to response the request from sipsak while doing experiment. The siptest acts like a simplified user agent that just response positive response without necessary database lookup than ordinary user agent does. The dummy echo user agent will minimize undetectable factors due to overheads from database lookup.

In order to experimenting and measure the performance difference between two schemes, we deploy the experimental topology shown in Fig. 5. The experimental topology was three mobile devices which connected via IEEE 802.11b wireless LAN with IBSS mode (ad-hoc mode) same wireless LAN channel and transmission rate is limited to DS/1Mbps.

We divide the delay time of SIP signaling into seven time segments from s1 to s7, shown in Fig. 6. Each time segment represents the time difference of transmission between two hosts. According to different scheme, the time segments will includes several process time segments, the interval of process time segments will be smaller than transmission time segment.

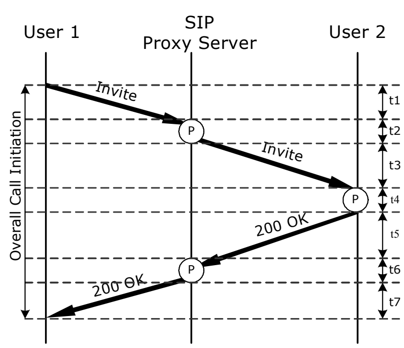

Fig. 6 Segments of SIP Signaling

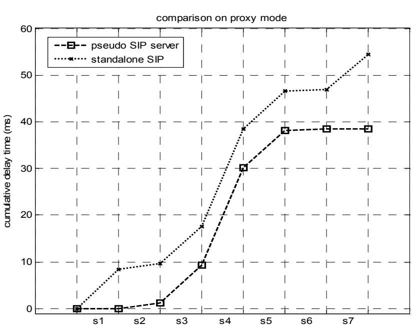

Fig. 7 Signaling delay

Since our experimental topology was three independent mobile host, it might be comes with system clock synchronization issue when measure the delay time from each timestamp of transmitted data. We deploy the time synchronizing mechanism by network time protocol (NTP). Each host will synchronize the system clock periodically to NTP server, the time synchronize procedure is carefully adjusted to avoid possible overhead to network or system resources. Due to the limit of the paper length, we just discuss the result of the processing delay, shown in Fig. 7, and our proposed scheme shows better performance than standalone SIP architecture.

5 Conclusions

In this paper, we designed and implemented an ad-hoc VoIP architecture using UPnP protocol to provide a configuration free and free access of VoIP communication under ad-hoc network. Our proposed remote user discovery module by UPnP and innovative pseudo SIP server provide the realization of ad-hoc VoIP services. Also, our system was implemented as tiny as possible which makes it much easier to be embedded into mobile devices with limited computation ability as well as storage capacity. In the future, we will further investigate the overhead elimination of remote user discovery module and conduct more analysis on the performance.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of the National Science Council of Taiwan, under Contract No. NSC-100-2221-E-142-008 and National Taichung University regarding the MoE project (No.1000121946A, and 1000157767A).

References

1. J. Rosenberg, et al., "SIP: Session Initiation Protocol," IETF RFC 3261, June 2002.
2. H. Khlifi, A. Agarwal, J-C. Grégoire, "A Framework To Use SIP in Ad-Hoc Networks," Proceedings of IEEE 2003 Canadian Conference on Electrical and Computer Engineering, pp. 985-988, May 4-7, 2003.
3. L.H. Chang, C.H Sung, S.Y Chiu, Y.W. Lin, "Design and Realization of Ad-hoc VoIP with embedded p-SIP Server", The Journal of Systems and Software, vol. 83, issue 12, pp. 2536-2555, Dec. 2010
4. L.H. Chang, C.H. Sung, H.C. Chu, J.J. Liaw, "Design and Implementation of the Push-to-Talk Service in Ad-hoc VoIP Network", IET Communications, vol. 3, Iss. 5, pp. 740–751, May 2009
5. I. Miladinovic and J. Stadler, "SIP Extension for Multiparty Conferencing," draft-miladinovic-sip-multiparty-ext-01.txt, September 2002.
6. UPnP Forum, http://www.upnp.org
7. S. Chemishkian, J. Lund, "Experimental Bridge LONWORKS / UPnP™ 1.0," IEEE on Consumer Communications and Networking Conference, pp. 400-405, Jan. 5-8 2004.