PRESENCE CLOUD ORGANIZES PRESENCE SERVERS INTO A SERVER-TO-SERVER ARCHITECTURE

D. Dedeepya Sailakshmi[1], Poreddy Dayakar[2]
[1]Mtech, CSE Dept, MLRIT, Hyderabad
[2]Mtech, Asst. Professor, MLRIT,Dundigal,Hyderabad

***Abstract*— Social network applications are becoming increasingly popular on mobile devices. A mobile presence service is an essential component of a social network application because it maintains each mobile user's presence information, such as the current status (online/offline), GPS location and network address, and also updates the user's online friends with the information continually. If presence updates occur frequently, the enormous number of messages distributed by presence servers may lead to a scalability problem in a large-scale mobile presence service. To address the problem, we propose efficient and scalable server architecture, called Presence Cloud, which enables mobile presence services to support large-scale social network applications. When a mobile user joins a network, Presence Cloud searches for the presence of his/her friends and notifies them of his/her arrival. Presence Cloud organizes presence servers into a quorum-based server-to-server architecture for efficient presence searching. It also leverages a directed search algorithm and a one-hop caching strategy to achieve small constant search latency. We analyze the performance of Presence Cloud in terms of the search cost and search satisfaction level.**

The search cost is defined as the total number of messages generated by the presence server when a user arrives; and search satisfaction level is defined as the time it takes to search for the arriving user's friend list. The results of simulations demonstrate that Presence Cloud achieves performance gains in the search cost without compromising search satisfaction.

***Index Terms*— Social networks, mobile presence services, distributed presence servers, cloud computing.**

I. INTRODUCTION

BECAUSE of the ubiquity of the Internet, mobile devices and cloud computing environments can provide presence-enabled applications, *i.e.*, social network applications/services, worldwide. Facebook [1], Twitter [2], Foursquare [3], Google Latitude [4], buddycloud [5] and Mobile Instant Messaging (MIM) [6], are examples of presence-enabled applications that have grown rapidly in the last decade. Social network services are changing the ways in which participants engage with their friends on the Internet. They exploit the information about the status of participants including their appearances and activities to interact with their friends. Moreover, because of the wide availability of mobile devices (e.g., Smartphones) that utilize wireless mobile network technologies, social network services enable participants to share live experiences instantly across great distances. For example, Facebook receives more than 25 billion shared items every month and Twitter receives more than 55 million tweets each day.

In the future, mobile devices will become more powerful, sensing and media capture devices. Hence, we believe it is inevitable that social network services will be the next generation of mobile Internet applications. A mobile presence service is an essential component of social network services in cloud computing environments.

The key function of a mobile presence service is to maintain an up-to-date list of presence information of all mobile users. The presence information includes details about a mobile user's location, availability, activity, device capability, and preferences. The service must also bind the user's ID to his/her current presence information, as well as retrieve and subscribe to changes in the presence information of the user's friends. In social network services, each mobile user has a friend list, typically called a buddy list, which contains the contact information of other users that he/she wants to communicate with. The mobile user's status is broadcast automatically to each person on the buddy list whenever he/she transits from one status to the other. For example, when a mobile user logs into a social network application, such as an IM system, through his/her mobile device, the mobile presence service searches for and notifies everyone on the user's buddy list. To maximize a mobile presence service's search speed and minimize the notification time, most presence services use server cluster technology [7]. Currently, more than 500 million people use social network services on the Internet [1].

Given the growth of social network applications and mobile network capacity, it is expected that the number of mobile presence service users will increase substantially in the near future. Thus, a scalable mobile presence service is deemed essential for future Internet applications.

In the last decade, many Internet services have been deployed in distributed paradigms as well as cloud computing applications. For example, the services developed by Google and Facebook are spread among as many distributed servers as possible to support the huge number of users worldwide. Thus, we explore the relationship between distributed presence servers and server network topologies on the Internet, and propose an efficient and scalable server-to server overlay architecture called Presence Cloud to improve the efficiency of mobile presence services for large-scale social network services.

First, we examine the server architectures of existing presence services, and introduce the buddy-list search problem in distributed presence architectures in large-scale geographically data centers. The *buddy-list search problem* is a scalability problem that occurs when a distributed presence service is overloaded with buddy search messages.

Then, we discuss the design of Presence Cloud, a scalable server-to-server architecture that can be used as a building block for mobile presence services. The rationale behind the design of Presence Cloud is to distribute the information of millions of users among thousands of presence servers on the Internet. To avoid single point of failure, no single presence server is supposed to maintain service-wide global information about all users. Presence Cloud organizes presence servers into a quorum-based server-to-server architecture to facilitate efficient buddy list searching. It also leverages the server overlay and a directed buddy search algorithm to achieve small constant search latency; and employs an active caching strategy that substantially reduces the number of messages generated by each search for a list of buddies. We analyze the performance complexity of Presence Cloud and two other architectures, a Mesh-based scheme and a Distributed Hash Table (DHT)-based scheme.

Through simulations, we also compare the performance of the three approaches in terms of the number of messages generated and the search satisfaction which we use to denote the search response time and the buddy notification time. The results demonstrate that PresenceCloud achieves major performance gains in terms of reducing the number of messages without sacrificing search satisfaction. Thus, Presence Cloud can support a large-scale social network service distributed among thousands of servers on the Internet.

Presence Cloud can also be utilized by Internet social network applications and services that need to replicate or search for mutable and dynamic data among distributed presence servers. The second contribution is that we analyze the scalability problems of distributed presence server architectures, and define a new problem called the buddy-list search problem. Through our mathematical formulation, the scalability problem in the distributed server architectures of mobile presence services is analyzed. Finally, we analyze the performance complexity of Presence-Cloud and different designs of distributed architectures, and evaluate them empirically to demonstrate the advantages of Presence Cloud.

II. PROBLEM STATEMENT

Well known commercial IM systems leverage some form of centralized clusters to provide presence services. Jennings III *et al.* presented taxonomy of different features and functions supported by the three most popular IM systems, AIM, Microsoft MSN and Yahoo! Messenger. The authors also provided an overview of the system architectures and observed that the systems use client-server-based architectures. Skype, a popular voice over IP application, utilizes the Global Index (GI) technology to provide a presence service for users. GI is a multi-tiered network architecture where each node maintains full knowledge of all available users. Since Skype is not an open protocol, it is difficult to determine how GI technology is used exactly. Moreover, Xiao *et al.* analyzed the traffic of MSN and AIM system. They found that the presence information is one of most messaging traffic in instant messaging systems. In, authors shown that the largest message traffic in existing presence services is buddy NOTIFY messages.

Recently, there is an increase amount of interest in how to design a peer-to-peer SIP. P2PSIP has been proposed to remove the centralized server, reduce maintenance costs, and prevent failures in server-based SIP deployment. To maintain presence information, P2PSIP clients are organized in a DHT system, rather than in a centralized server. However, the presence service architectures of Jabber and P2PSIP are distributed, the *buddy-list search problem* we defined later also could affect such distributed systems. It is noted that few articles in discuss the scalability issues of the distributed presence server architecture. Saint Andre analyzes the traffic generated as a result of presence information between users of inter-domains that support the XMPP. Houri *et al.* Show that the amount of presence traffic in SIMPLE can be extremely heavy, and they analyze the effect of a large presence system on the memory and CPU loading. Those works in study related problems and developing an initial

set of guidelines for optimizing inter-domain presence traffic and present DHT-based presence server architecture.

Recently, presence services are also integrated into mobile services. For example, 3GPP has defined the integration of presence service into its specification in UMTS. It is based on SIP protocol, and uses SIMPLE to manage presence information. Recently, some mobile devices also support mobile presence services. For example, the Instant Messaging and Presence Services (IMPS) was developed by the Wireless Village consortium and was united into Open Mobile Alliance (OMA) IMPS in 2005. In, Chen *et al.* proposed a weakly consistent scheme to reduce the number of updating messages in mobile presence services of IP Multimedia Subsystem (IMS). However, it also suffers scalability problem since it uses a central SIP server to perform presence update of mobile users. In, authors presented the server scalability and distributed management issues in IMS-based presence service.

III. SYSTEM DEVELOPMENT

Presence Cloud server overlay

The Presence Cloud server overlay construction algorithm organizes the PS nodes into a server-to-server overlay, which provides a good low-diameter overlay property. The low-diameter property ensures that a PS node only needs two hops to reach any other PS nodes.

One-hop caching strategy

To improve the efficiency of the search operation, Presence Cloud requires a caching strategy to replicate presence information of users. In order to adapt to changes in the presence of users, the caching strategy should be asynchronous and not require expensive mechanisms for distributed agreement. In Presence Cloud, each PS node maintains a *user list* of presence information of the attached users, and it is responsible for caching the *user list* of each node in its PS list, in other words, PS nodes only replicate the *user list* at most one hop away from itself. The cache is updated when neighbors establish connections to it, and periodically updated with its neighbors. Therefore, when a PS node receives a query, it can respond not only with matches from its own *user list*, but also provide matches from its caches that are the user lists offered by all of its neighbors.

Directed buddy search

We contend that minimizing searching response time is important to mobile presence services. Thus, the buddy list searching algorithm of Presence Cloud coupled with the two-hop overlay and one-hop caching strategy ensures that Presence Cloud can typically provide swift responses for a large number of mobile users. First, by organizing PS nodes in a server-to-server overlay network, we can therefore use one-hop search exactly for queries and thus reduce the network traffic without significant impact on the search results. Second, by capitalizing the one-hop caching that maintains the user lists of its neighbors, we improve response time by increasing the chances of finding buddies. Clearly, this mechanism both reduces the network traffic and response time. Based on the mechanism, the population of mobile users can be retrieved by a broadcasting operation in any PS node in the mobile presence service. Moreover, the broadcasting message can be piggybacked in a buddy search message for saving the cost.

IV. RELATED WORK

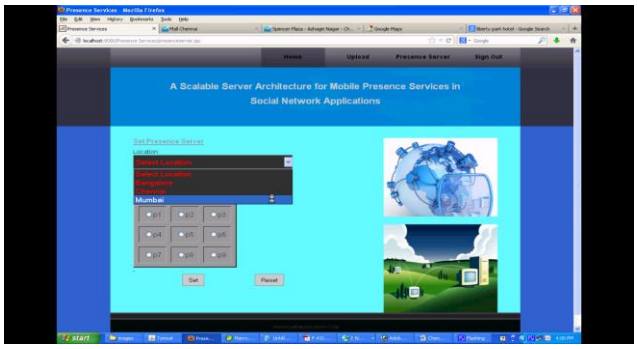
In this section, we describe previous researches on presence services, and survey the presence service of existing systems. Well known commercial IM systems leverage some form of centralized clusters to provide presence services [7]. Jennings III *et al.* [7] presented a taxonomy of different features and functions supported by the three most popular IM systems, AIM, Microsoft MSN and Yahoo! Messenger. The authors also provided an overview of the system architectures and observed that the systems use client-server-based architectures. Skype, a popular voice over IP application, utilizes the Global Index (GI) technology [8] to provide a presence service for users. GI is a multi-tiered network architecture where each node maintains full knowledge of all available users. Since Skype is not an open protocol, it is difficult to determine how GI technology is used exactly. Moreover, Xiao *et al.* [9] analyzed the traffic of MSN and AIM system. They found that the presence information is one of most messaging traffic in instant messaging systems. In [10], authors shown that the largest message traffic in existing presence services is buddy NOTIFY messages.

Several IETF charters [11]–[13] have addressed closely related topics and many RFC documents on instant messaging and presence services have been published, e.g.,XMPP [14], SIMPLE [15]. Jabber [16] is a well-known deployment of instant messaging technologies based on distributed architectures. It captures the distributed architecture of SMTP protocols. Since Jabber's

architecture is distributed, the result is a flexible network of servers that can be scaled much higher than the monolithic, centralized presence services. Recently, there is an increase amount of interest in how to design a peer-to-peer SIP [17]. P2PSIP [18] has been proposed to remove the centralized server, reduce maintenance costs, and prevent failures in server-based SIP deployment. To maintain presence information, P2PSIP clients are organized in a DHT system, rather than in a centralized server. However, the presence service architectures of Jabber and P2PSIP are distributed, the *buddy-list search problem* we defined later also could affect such distributed systems.

It is noted that few articles in [9]–[11] discuss the scalability issues of the distributed presence server architecture. Saint Andre [19] analyzes the traffic generated as a result of presence information between users of inter-domains that support the XMPP. Houri *et al.* [20] show that the amount of presence traffic in SIMPLE [13] can be extremely heavy, and they analyze the effect of a large presence system on the memory and CPU loading. Those works in [11], [12] study related problems and developing an initial set of guidelines for optimizing inter-domain presence traffic and present a DHT-based presence server architecture. Recently, presence services are also integrated into mobile services. For example, 3GPP has defined the integration of presence service into its specification in UMTS. It is based on SIP [13] protocol, and uses SIMPLE [15] to manage presence information. Recently, some mobile devices also support mobile presence services. For example, the Instant Messaging and Presence Services (IMPS) was developed by the Wireless Village consortium and was united into Open Mobile Alliance (OMA) IMPS [14] in 2005. In [15], Chen *et al.* proposed a weakly consistent scheme to reduce the number of updating messages in mobile presence services of IP Multimedia Subsystem (IMS). However, it also suffers scalability problem since it uses a central SIP server to perform presence update of mobile users [16]. In [17], authors presented the server scalability and distributed management issues in IMS-based presence service.

Results:



The Presence Cloud server overlay construction algorithm organizes the PS nodes into a server-to-server overlay, which provides a good low-diameter overlay property

V. CONCLUSION

In this paper, we explore a Cloud Computing new entity In this paper, we have presented Presence Cloud, a scalable server architecture that supports mobile presence services in large-scale social network services. We have shown that Presence Cloud achieves low search latency and enhances the performance of mobile presence services. In addition, we discussed the scalability problem in server architecture designs, and introduced the buddy-list search problem, which is a scalability problem in the distributed server architecture of mobile presence services. Through a simple mathematical model, we show that the total number of buddy search messages increases substantially with the user arrival rate and the number of presence servers. The results of simulations demonstrate that Presence Cloud achieves major performance gains in terms of the search cost and search satisfaction. Overall, Presence Cloud is shown to be a scalable mobile presence service in large-scale social network services.

REFERENCES

[1] Facebook, http://www.facebook.com.
[2] Twitter, http://twitter.com.
[3] Foursquare http://www.foursquare.com.
[4] Google latitude, http://www.google.com/intl/ enus/latitude/intro.html.
[5] Buddycloud, http://buddycloud.com.
[6] Mobile instant messaging, http://en.wikipedia.org/ wiki/Mobile instant messaging.
[7] R. B. Jennings, E. M. Nahum, D. P. Olshefski, D. Saha, Z.-Y. Shae, and C. Waters, "A study of internet instant messaging and chat protocols," *IEEE Network*, 2006.
[8] Gobalindex, http://www.skype.com/intl/en-us/support/user-guides/ p2pexplained/.
[9] Z. Xiao, L. Guo, and J. Tracey, "Understanding instant messaging traffic characteristics," *Proc. of IEEE ICDCS*, 2007.
[10] C. Chi, R. Hao, D. Wang, and Z.-Z. Cao, "Ims presence server: Traffic analysis and performance modelling," *Proc. of IEEE ICNP*, 2008.
[11] Instant messaging and presence protocol working group http://www.ietf.org/html.charters/imppcharter.html.
[12] Extensible messaging and presence protocol ietf working group http://www.ietf.org/html.charters/xmpp-charter.html.
[13] Sip for instant messaging and presence leveraging extensions working group. http://www.ietf.org/ html.charters/simplecharter. html.
[14] P. Saint-Andre, "Extensible messaging and presence protocol (xmpp): Instant messaging and presence describes instant messaging (im), the most common application of xmpp," *RFC 3921*, 2004.
[15] B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, and D. Gurle, "Session initiation protocol (sip) extension for instant messaging," *RFC 3428*, 2002.
[16] Jabber, http://www.jabber.org/.
[17] Peer-to-peer session initiation protocol ietf working group, http://www.ietf.org/html.charters/p2psip-charter.html.
[18] K. Singh and H. Schulzrinne, "Peer-to-peer internet telephony using sip," *Proc. of ACM NOSSDVA*, 2005.
[19] P. Saint-Andre, "Interdomain presence scaling analysis for the extensible messaging and presence protocol (xmpp)," *RFC Internet Draft*, 2008.
[20] A. Houri, T. Rang, and E. Aoki, "Problem statement for sip/simple," *RFC Internet-Draft*, 2009.

AUTHOR DETAILS:

First Author: ***D.Dedeepya Sailakshmi*** received B.Tech Degree in Computer Science and Engineering from Institute of Aeronautical Engineering in the year 2011. She is currently M.Tech student in the Computer Science and Engineering from MLR Institute of Technology. Her research interested areas are in the field of Cloud Computing, Mobile Computing and Data Ware Housing.

Second Author: ***P Dayaker,*** received his M.Tech (Software Engineering) in 2010 from JNTU, Hyderabad, India, Now he is working as Assistant Professor in Dept.of CSE in MLR Institute of Technology, Hyderabad, A.P., India. He has more than 5 years experience in academics. He published 4 papers in International Journal on wireless sensor networks and data mining. He presented a paper in international conference

Third Author: G Kiran Kumar is working as Associate Professor & HOD-CSE in MLR Institute of technology. He did M.Tech from Osmania University, Hyderabad, and submitted Ph.D from Nagarjuna University. His research areas include Data Mining, Spatial data mining, Software Engineering.