International Journal of Interdisciplinary Innovative Research &Development (IJIIRD)
ISSN: 2456-236X
Vol. **06** Special Issue **01** | **2021**

A Review on Scalable and Efficient Server for Mobile Presence Enable Services

[1]Mahesh Shastri, [2]Manjiri Karande

[1]Assistant Professor, Computer Science & Engineering, COE Malkapur, Maharashtra, India,
[2]Assistant Professor, Computer Science & Engineering, COE Malkapur, Maharashtra, India

ABSTRACT

The popularity of Social network applications are tremendously increasing because of easily and economically availability of mobile devices. A mobile device consist of mobile presence service which includes current status (online/offline), GPS location and network address, and also updates the user's online friends with the information continually. As the number of mobile users are increasing very rapidly the number of message delivered to mobile presence server may create a scalability problem in large scale presence service. To overcome with this problem, we propose a server architecture, called PresenceCloud which is more efficient and scalable. PresenceCloud uses a server-to-server architecture to organize presence servers. It also have a directed search algorithm and a one-hop caching strategy to achieve small constant search latency. Whenever a mobile user enters into the network, Presence cloud search for mobile user's friends and notify them about the arrival of the mobile user. On the basis of search satisfaction level and search cost the performance of Presence Cloud is analyzed. Search satisfaction level is the time taken by presence server to search friend list of the arrived user in the network. Search cost is the total number of messages generated by presence server after arrival of user. The result shows that presence cloud achieves performance gains in the search cost without compromising search satisfaction.

Index Terms - Presence Cloud, Mobile Presence Service, Cloud computing, Social networking

I. INTRODUCTION

Use of Presence-enabled applications is growing with the increasing use of internet. Mobile devices and cloud computing environment provides these applications such as social network applications/services, worldwide. The examples of presence enabled applications are Twitter, Facebook, Google Latitude [1], Foursquare, Mobile Instant Messaging (MIM) [2] and buddycloud .These applications have grown rapidly in the last decade. Various new applications are being developed to provide social network services to engage participants with their friends on the Internet. They share the information about the status of participants. So, because of the large use of mobile devices such as Smart phones that utilizes wireless mobile network technologies. Participants can share their live experiences instantly across great distances through social network services. Advantages of mobile devices are increasing day by day will become more powerful. Maintain user information of till today's date, list of presence information is generated and is main functionality of all mobile presence service .The presence information consists of the details about a mobile location of user , user's availability , capability of device , activity. The service should combine the user's ID along with presence information. It can also retrieve and subscribe to modifications in the presence information of the user's friends. In social network services, every mobile user has a list of friends, which is called as buddy list, which includes the contact information of other users that user wants to communicate with. When user switch from one state to another mobile user's status is broadcasted to other user's in its buddy list. Most presence services use server cluster technology [3] to maximize a mobile presence service's search speed and minimize the notification time. Social network services are used by more than 500 million people on the Internet.

Server-to-server architecture is proposed to improve scalability and efficiency of mobile services called Mobile Presence Cloud. First, the server architectures of existing presence services are examined, and introduced the buddy-list search problem in distributed presence architectures in large-scale geographically data centers. When distributed presence server is overloaded with buddy search message at that time scalability problem occurs and it called as buddy-list search problem. Basic building block for mobile presence service is scalable server-to-server architecture. Service-wide global information about all users should not maintain by single presence server in order to avoid single point of failure. Large-scaled social network service is supported by Presence Cloud. The

contribution of this paper consists of three things: First, Presence Cloud is among the architecture for mobile presence services, and it outperform based on distributed hash tables. Secondly, new problem called the buddy-list search problem is defined and scalability problem of distributed presence server architecture. Finally, performance complexity of presence-Cloud and their evaluation is done.

II. LITERATURE REVIEW

2.1 Extensible Messaging and Presence Protocol (XMPP): This is a technology for the near-real-time presence notifications, exchange of messages and, where data is done by XML. XMPP depends on some technologies which modifies their own data. For encryption it uses TLS and for authentication it uses Simple Authentication and Security Layer (SASL).For server-to-server stream, XMPP is deployed using TLS and the SASL EXTERNAL mechanism, where X.509 certificate is used to represent each peer [5].

2.2 Instant Messaging and Presence Protocol (Impp): Instant messaging is different from email and its main focus is instant end-user message delivery. IMPP is architecture for simple instant messaging and presence awareness/notification [7]. It provides features such as access control, encryption and message.

2.3 Instant Messaging (IM) System: Popularity of Instant Messaging (IM) increasing due to its ease of use, possibility of multitasking, quick response time. IM is used for various purposes like scheduling face to face meetings, simple requests and responses, or for checking availability of colleagues and friends. IM system routes text messages through central servers. Firewall traversal gives more control to instant messaging companies for leads to bottleneck. MSN/Windows Live Messenger and AOL Instant Messenger (AIM) these are two very popular instant messaging services. An IM server drops the extraneous traffic to maintain instantaneous nature of communication [4].

2.4 Skype: Skype is a popular VoIP application, in which Global Index (GI) technology is used to provide a presence service for users. At each node full knowledge of all users are maintained as Skype provides multi-tiered network architecture [3].

2.5 Jabber: Jabber is instant messaging technology which follows distributed architecture. 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.

2.6 P2PSIP: To remove the centralized server P2PSIP is proposed. It also reduces maintenance costs, and prevents failures in server-based SIP deployment. Examples of distributed presence service architectures are Jabber and P2PSIP; the buddy-list search problem could affect such distributed systems [9].

III. SYSTEM ARCHITECHTURE

The general architecture can be shown in Figure 1. In Figure 1, Base station establishes and controls the connections as mobile devices are connected to mobile networks via base station. Services provided to mobile user as authentication, accounting, authorization based on home agent. The requests delivered to a cloud through the Internet.

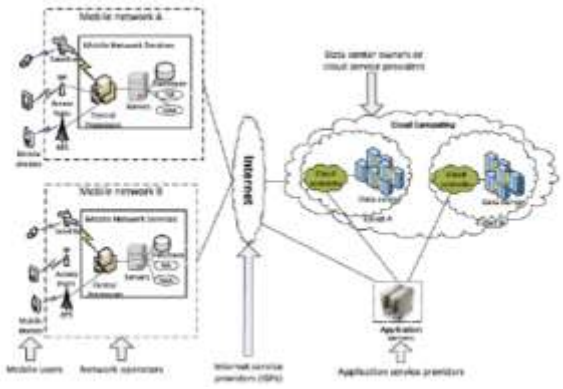


Figure 1 System Architecture

To compare cloud computing with grid computing four-layer architecture is introduced. Layered architecture of cloud computing is shown in Fig 2. Cloud computing model is designed in such way that it meets user's needs. By using Infrastructure as a Service (IaaS) which provided servers, storage, network as service, Platform as a Service (PaaS) gives computation platform to user, and Software as a Service (SaaS) we can use software which are located on cloud. These services are present at upper layer. Data centers layer: Hardware facility and infrastructure for clouds provided by this layer. To provide services for customer number of servers is linked with high-speed networks. Geographical area where population is less data center are built over there which leads high power supply and less risk. In the upper layers consists of Software as a Service (SaaS), and Infrastructure as a Service (IaaS), Platform as a Service (PaaS). Data centers layer: Infrastructure and hardware facility provided by this layer. In data center layer, to provide service for customers a number of servers are linked with high-speed networks.

Infrastructure as a Service (IaaS): IaaS provides storage, network and servers as a service. Client can buy can buy fully outsource services.The client typically pays on a per-use basis, so cost of payment is saved.

Platform as a Service (PaaS): In PaaS it provides computation platform as a service to user. It is mainly used by developers to deploy their code on public cloud. Examples of PaaS are Google App Engine, Amazon Map Reduce/Simple Storage Service.

Software as a Service (SaaS): SaaS software is built centrally by service provider and for end user's it is provided on demand. It provides any time anywhere access. Gmail provides software as service.

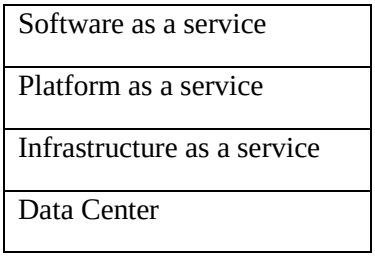


Figure 2 Service-oriented cloud computing architecture.

IV. COMPONENTS OF PRESENCE CLOUD

4.1 Presence Cloud Server Overlay

A good low-diameter overlay property is provided by the Presence Cloud server overlay construction algorithm. Server-to-server overlay is organized by the PS nodes. A PS node only needs two hop to reach any other PS nodes is ensured by the low diameter property.

4.2 One-hop Caching Strategy

Improvement of the efficiency of search operation is done by replicating presence information of users. User list is cached at each PS node and it is maintained by each node. Replication user list is done by Ps node only at most one hop away from itself. When neighbors establish connections to it the cache is updated, and periodically updated with its neighbors. When query arrives at PS node, it goes not only for its own user list, but also check for matches from its caches offered by all neighbors.

4.3 Directed Buddy Search

Important aspect of mobile presence services is minimizing searching response time. The buddy list searching algorithm of Presence Cloud combined with the two-hop overlay and one-hop caching strategy. First for queries one-hop search is used Second, by using the one-hop caching the user lists of its neighbors is maintained, by increasing the chances of finding buddies response time is improved. By using this mechanism the response time and network traffic is reduced.

V. ALGORITHM USED

Algorithm: Presence Cloud Stabilization Algorithm Basically use for periodic verification of PS node in Plist
Input: Set of the current PSlist of PS node
Output: Establish connection with PS node
for each node **do**
for each document in the corpus C **do**
Check for value of current node and node in plist
if no match found **then** refresh Plist entries
Again check correct node
if returns correct node **then** establish connection
else nil **then** pick random node from same column or row of failed node and establish connection
Send heartbeat message
If no connectivity between current node and node in plist i.e node n then n will pick random node from same column or row of failed node and establish connection
return connection of PS node with node in PSlist

VI. Result

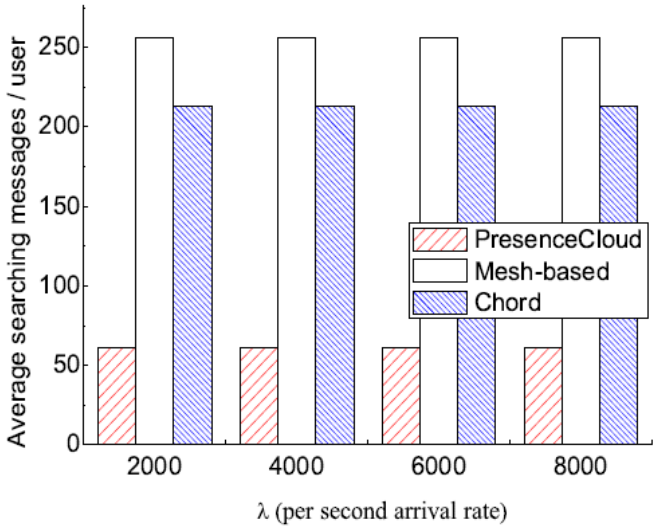
In this system, we have presented PresenceCloud, a scalable server architecture that supports mobile presence services in large-scale social network services. We have shown that PresenceCloud achieves low search latency and enhances the performance of mobile presence services. In addition, we scope the 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. Overall, PresenceCloud is shown to be a scalable mobile presence service in large-scale social network services.

6.1 Comparison of Presence Architecture

We analyze the performance of PresenceCloud in terms of the search cost and search satisfaction level. We summarize the comparison of different schemes in Table 4.1. The columns show the different schemes. The label ”Search” means the maximum number of messages sent by a PS node when a mobile user joins; label ”Replica” means the maximum number of buddy replicas in a PS node. label ”Latency” means the buddy search latency, we quantify this metric by the diameter of the server overlay. This is reasonable because, in general, the search latency is dominated by the diameter of the overlay. label ”Message Size” means the average number of required buddy in a message. label ”Message Reply Hops” means that the maximum hop counts of a message relayed by PS nodes. label ”Maintenance Overhead” means the number of PS nodes maintained by a PS node. In Table 4.1, none of the schemes is a clear winner. The mesh-based approach achieves good search latency at the expense of the other metrics. Our PresenceCloud yields a low communication cost in large-scale server architecture and small search latency. Note that u is denoted the average number of mobile users attached to a PS node and b is the number of buddy sizes. Meanwhile, the DHT-based method provides good features for low replica load, however it comes at a prices of increased searching latency.

	Mesh	PresenceCloud	DHT-based
Search	$O(n)$	$O(\sqrt{n})$	$O(b \times \log n)$
Replicas	$O(\|U\|)$	$O(\sqrt{n} \times u)$	$O(u)$
Latency	one hop	two hops	$\log n$ hops
Message Size	b	b/n	b/n
Message Reply Hops	$O(1)$	$O(1)$	$O(\log n)$
Maintenance Overhead	$O(n)$	$O(\sqrt{n})$	$O(\log n)$

Table 1: Presence Architecture Comparison



Graph 1: The average message transmissions per searching operation

VII. CONCLUSION

Mobile presence services are supported by scalable server Architecture called as Presence Cloud. It enhances the performance of mobile presence services and achieves low cost latency. Presence Cloud solves the Buddy List Problem. This problem sometimes referred to as Scalability problem. With the number of presence servers and the user arrival rate the total number of buddy search messages increases substantially. Major performance gain is in terms of search satisfaction and search cost. It is scalable presence service which provides services to the large network. In future scope, it is intended to extend the presence cloud system by providing the encryption between communications of presence servers in the Presence Cloud. This could be extended further by central authentication of every Presence server. Also, when it comes to Buddy List the search response time could be calculated. Failure handling of presence server will also be considered for efficiently increasing the performance.

REFERENCES

[1] Google latitude, http://www.google.com/intl/enus/latitude/intro.html.
[2] 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.
[3] Gobalindex,http://www.skype.com/intl/enus/support /user-guides/ p2pexplained/.
[4] Instant messaging and presence protocol ietf working group http://www.ietf.org/html.charters/impp-harter.html. [5] Extensible messaging and presence protocol ietf working group http://www.ietf.org/html.charters/xmpp-harter.html.
[6] Z. Xiao, L. Guo, and J. Tracey,”Understanding instant messagingtraffic characteristics,” *Proc. of IEEE ICDCS*, 2007.
[7] C. Chi, R. Hao, D. Wang, and Z.-Z. Cao,”Ims presence server: Traffic analysis and performance modelling,” *Proc. of IEEE ICNP*, 2008.
[8] Sip for instant messaging and presence leveraging extension ietf working group. http://www.ietf.org/ html.charters/simplecharter.html
[9] Peer-to-peer session initiation protocol ietf working group,http://www.ietf.org/html.charters/p2psip-charter.html.
[10] Open Mobile Alliance, OMA instant messaging and presence, service, 2005.
[11] W.-E. Chen, Y.-B. Lin, and R.-H. Liou, ”A weakly consistent scheme for ims presence service,” *IEEE Transactions on Wireless Communications*, 2009.
[12] D. Eastlake and P. Jones, ”Us secure hash algorithm 1 (SHA1),” RFC 3174, 2001.