International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Architecture for Scalable Problem in Mobile Presence Service Applications

K. Tarakeshwar[1], G. PavanKumar[2]

[1]Assistant Professor CS & IT, M. Tech, G. Pullaiah College of Engineering and Technology, Kurnool

[2]Department of CSE, G. Pullaiah College of Engineering and Technology, Kurnool

Abstract: ***Mobile devices are prevalent in the recent days with a lot of advanced applications. Social networking application is one of the applications used in the mobile devices. The social networking application involves the users who logged in to the system, their status updates and their presence, so the social networking application deals with this kind of features. In my project in order to explain this mobile presence service is used. A mobile presence service involves the presence of users and updates the user information manually. The updates regarding the user information is a "presence update", if this presence updates occurs continuously then more number of messages distributed by the servers may lead to scalability problem in an large scale network applications. In order to resolve the problem, a scalable architecture has been proposed which is "Presence cloud" which resolves the scalability problem in large scale network applications. When a user joins in an network the proposed architecture checks for the arrival of users and notifies then automatically which leads to resolve this scalability problem. The proposed scalable server architecture organizes the servers into a quorum based server to server architecture for the purpose of efficient searching. The performance of the architecture has been evaluated in terms of cost and search optimization level.***

Keywords: Social networks, mobile presence services, distributed presence servers, cloud computing.

1. Introduction

The main reason because of the iniquitousness of the web, mobile devices and cloud computing environments is to provide presence-enabled applications, i.e., social network applications/services, worldwide. Facebook [1], Twitter [2],Foursquare [3], Google Latitude [4], buddy cloud [5] and Mobile Instant electronic messaging (MIM) [6], square measure samples of presence-enabled applications that have big speedily in the last decade. Social network services square measure dynamic the ways within which participants have interaction with their friends on the net. They exploit the knowledge regarding the standing of participants as well as their appearances and activities to move with their friends. Moreover, thanks to the wide handiness of mobile devices (e.g., Smart phones) that utilize wireless mobile network technologies, social network services alter participants to share live experiences instantly across nice distances. As an example, Facebook receives over twenty five billion shared things monthly and Twitter receives quite fifty five million tweets day after day. In the future, mobile devices can become a lot of powerful sensing and media capture devices. Hence, we tend to believe it is inevitable that social network services are following generation of mobile web applications. A mobile presence service is a necessary part of social network services in cloud computing environments. The key perform of a mobile presence service is to maintain an up-to-date list of presence info of all mobile users. The presence info includes details about a mobile user's location, handiness, activity, device capability, and preferences. The service should additionally bind the user's ID to his/her current presence info, as well as retrieve and take changes within the presence information of the user's friends. In social network services, each mobile user features a friend list, usually referred to as a crony list that contains the contact info of different users that he/she desires to speak with.

The mobile user's status is broadcast mechanically to every person on the buddy list whenever he/she transits from one standing to the other. For instance, once a mobile user logs into a social network application, like AN IM system, through his/her mobile device, the mobile presence service searches for and notices everybody on the user's crony list. To maximize a mobile presence service's search speed and minimize the notification time, most presence services use server cluster technology [7]. Currently, quite five hundred million people use social network services on the web [1].Given the expansion of social network applications and mobile Network capability, it's expected that the quantity of mobile presence service users can increase considerably within the close to future. Thus, a scalable mobile presence service is deemed essential for future web applications.

In the last decade, several web services are deployed in distributed paradigms yet as cloud computing applications. For instance, the services developed by Google and Facebook square measure unfold among as several distributed servers as doable to support the massive variety of users worldwide. Thus, we have a tendency to explore the connection between distributed presence servers and server network topologies on the web, associate degreed propose an efficient and scalable server-to server overlay design known as Presence Cloud to boost the efficiency of mobile presence services for large-scale social network services. First, we have a tendency to examine the server architectures of existing presence services, and introduce the buddy-list search problem in distributed presence architectures in large-scale

Geographically knowledge centers. The buddy-list search drawback is a quantifiability drawback that happens once a distributed presence service is over laden with friend search messages. Then, we have a tendency to discuss the look of Presence Cloud, a scalable server-to-server design which will be used as a building block for mobile presence

services. The principle behind the design of Presence Cloud is to distribute the knowledge of lots of users among thousands of presence servers on the net. To avoid single purpose of failure, no single presence server is meant to keep up service-wide global data regarding all users. Presence Cloud organizes presence servers into a quorum-based server-to-server design to facilitate efficient friend list looking out. It also leverages the server overlay and a directed friend search algorithm to realize tiny constant search latency; and employs an active caching strategy that considerably reduces the number of messages generated by every rummage around for a list of buddies. We have a tendency to analyze the performance quality of Presence Cloud and two alternative architectures, a Mesh-based scheme and a Distributed Hash Table (DHT)-based theme. Through simulations, we have a tendency to additionally compare the performance of the three approaches in terms of the quantity of messages generated and also the search satisfaction that we have a tendency to use to denote the search latent period and also the friend notification time. The results demonstrate that Presence Cloud achieves major performance gains in terms of reducing the quantity of messages while not sacrificing search satisfaction. Thus, Presence Cloud will support a large-scale social network service distributed among thousands of servers on the Internet. The contribution of this paper is threefold. First, Presence Cloud is among the pioneering design for mobile presence services. To the simplest of our information, this is the first work that expressly styles a presence server architecture that significantly outperforms those primarily based distributed hash tables. Presence Cloud may also be utilized by Internet social network applications and services that require replicating or rummaging around for changeable and dynamic knowledge among distributed presence servers. The second contribution is that we analyze the quantifiability issues of distributed presence server architectures, and define a replacement drawback known as the buddy-list search drawback. Through our mathematical formulation, the quantifiability drawback within the distributed server architectures of mobile presence services is analyzed. Finally, we analyze the performance quality of Presence Cloud and completely different styles of distributed architectures, and evaluate them through empirical observation to demonstrate the benefits of Presence Cloud.

2. Design of Presence Cloud

The past few years has seen a veritable mania of analys is activity in Internet-scale object looking field, with many designed protocols and planned algorithms. Most of the previous algorithms are wont to address the fixed object searching downside in distributed systems for various intentions. However, folks ar unsettled, the mobile presence information is a lot of mutable and dynamic; a replacement style of mobile presence services is required to deal with the buddy list search downside, particularly for the demand of mobile social network applications.

Presence Cloud is employed to construct and maintain a distributed server design and might be wont to efficiently query the system for chum list searches. Presence Cloud consists of 3 main parts that are ran into a group of presence servers. Within the style of Presence Cloud, we refine the concepts of P2P systems and gift a selected style for mobile presence services. The 3 key parts of Presence Cloud are summarized below:

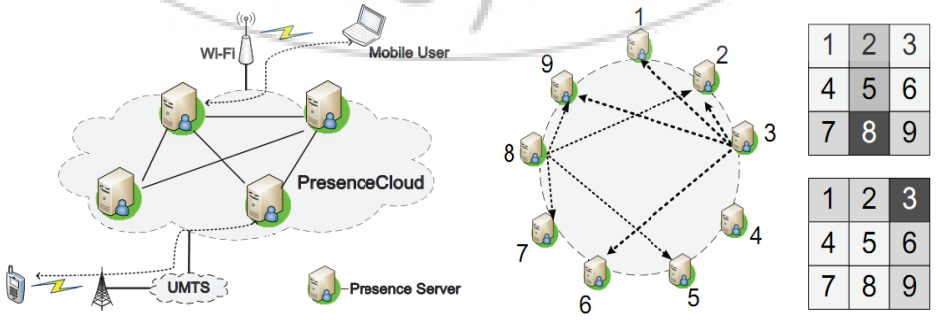
Presence Cloud server overlay organizes presence servers supported the conception of grid gathering system[29]. So, the server overlay of Presence Cloud has a balanced load property and a two-hop diameter with O(√n) node degrees, wherever n is that the variety of presence servers.

One-hop caching strategy is employed to scale back the number of transmitted messages and accelerate question speed. All presence servers maintain caches for the buddies offered by their immediate neighbors.

Directed chum search relies on the directed search strategy. Presence Cloud ensures Associate in Nursing one-hop search, it yields a little constant search latency on the average.

3. Presence Cloud Summary

The primary abstraction exported by our Presence Cloud is used to construct a scalable server design for mobile presence services, and might be wont to efficiently search the desired chum lists. We have a tendency to illustrate a straightforward summary of Presence Cloud in Fig. 2. Within the mobile web, a mobile user will access the web and create an information affiliation to Presence Cloud via 3G or Wi-Fi services. Once the mobile user joins and authenticates himself/herself to the mobile presence service, the mobile user is determinately directed to one of Presence Servers within the Presence Cloud by exploitation the Secure Hash formula, like SHA-1 [30]. The mobile user opens a communications protocol affiliation to the Presence Server(PS node) for management message transmission, significantly for the presence data. Once the management channel is established, the mobile user sends a call for participation to the connected postscript node for his/her chum list looking.



4. 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

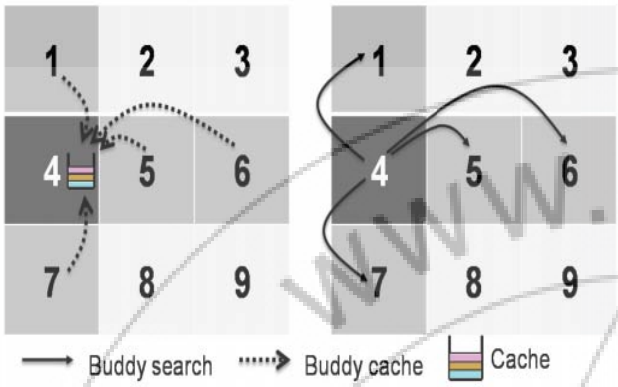


Figure An example of buddy list searching operations in Presence Cloud

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.

Our caching strategy does not require expensive overhead for presence consistency among PS nodes. When a mobile user changes its presence information, either because it leaves Presence Cloud, or due to failure, the responded PS node can disseminate its new presence to other neighboring PS nodes for getting updated quickly. Consequently, this one-hop caching strategy ensures that the user's presence information could remain mostly up-to-date and consistent throughout the session time of the user. 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.

5. Future Scope

"Scalable Mobile Presence Cloud with Communication Security". We analyze the performance of Presence Cloud in terms of the search cost and search satisfaction level. Our current PresenceCloud does not address the communication security problem, and the presence server authentication problem, we discuss the possible solutions as follows. The distributed presence service may make the mobile presence service more prone to communication security problems, such as malicious user attacks and the user privacy.

Several approaches are possible for addressing the communication security issues. For example, the Skype protocol offers private key mechanisms for end-to-end encryption. In PresenceCloud, the TCP connection between a presence server and users, or a presence server could be established over SSL to prohibit user impersonation and man-in-the-middle attacks. This end-to-end encryption approach is also used in XMPP/SIMPLE protocol.

6. Conclusion

Social networking applications are prevalent in many mobile devices. The presence cloud has been proposed and implemented to resolve the scalability problem and it uses an architecture which is scalable server architecture. The presence cloud achieves low search latency and enhances the performance of the mobile presence services. In this the scalability problem has been exercised and the buddy list search problem has been introduced. So, the presence cloud gains in terms of search cost and search satisfaction and enhances the performance of the system. The presence cloud is found to be scalable in large scale mobile devices.

References

[1] 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*.
[2] Z. Xiao, L. Guo, and J. Tracey, "Understanding instant messaging traffic characteristics," *Proc. of IEEE ICDCS*.
[3] C. Chi, Wang, and Z.-Z. Cao R. Hao, D.,"Ims presence server: Traffic analysis and performance modelling," *Proc. of IEEE ICNP*.
[4] M. Maekawa,"A*p n* algorithm for mutual exclusion in decentralized systems," *ACM Transactions on Computer Systems*.
[5] M. Steiner, T. En-Najjary, and E. W. Biersack,"Long term study of peer behavior in the kad DHT," *IEEE/ACM Trans. Netw.*
[6] Abdul-Rahman and S. Hailes., "A distributed trust model," *Proc. of the workshop on New security paradigms*