XMPP A Perfect Protocol for the New Era of Volunteer Cloud Computing

Kamlesh Lakhwani [1,] Ruchika Saini[1]
[1](Dept. of Computer Science and Engineering, Gyan Vihar University, Jaipur, Rajasthan, India)

ABSTRACT:
Volunteer Cloud Computing is based on the concept where highly distributed non-dedicated resources are harnessed to build a cloud so as to offer cloud services. By volunteer cloud computing, users and contributors can experience the impact of cloud computing by allowing users to share any type of services be it physical resources, software resources or applications. As volunteer clouds entities are allowed to communicate with each other and with other commercial clouds and clients also, it's necessary to implement an enhanced interoperable environment that can carry out effective communication between the communicating parties. This thesis contributes toward developing an enhanced interoperable environment where volunteer cloud entities are allowed to communicate with each other and also with other commercial clouds and clients via XMPP protocol (Extensible messaging and presence protocol). In this paper we propose an XMPP based messaging middleware architecture that can help in implementing such an environment

KEYWORDS: *cloud computing, commercial cloud, interoperability, middleware architecture, volunteer cloud, Service oriented architecture, security.*

1. INTRODUCTION

The concepts of cloud computing and volunteer computing are combined to form volunteer cloud computing. In volunteer cloud computing idle or non-dedicated resources [1] that are not designed to be cloud infrastructure are harnessed so as to provide cloud capabilities or services. By volunteer cloud computing [2] [3] [4] users and contributors can experience the impact of cloud computing by allowing users to share any type of service be it physical resources, software resources, applications.
It offers some advantages [3] [5]:

- Utilization of idle resources:
  Volunteer cloud computing makes use of non-dedicated idle resources to build a cloud and so as to deploy cloud services. Hence, it overall increases the efficiency of the system by exploiting these underused resources.
- Cost reduction :
  As volunteer cloud computing deals with the computing resources volunteered by individuals across the world, it prevents the organization, scientists and researchers from making any kind of investment in the resources. So it eliminates the requirement to have dedicated resources, as volunteer computing resources altogether generate a massive computing power which is sufficient to fulfill the needs and requirements of the projects and business.

- reduce overall power consumption**:**
- Presence of volunteer cloud decreases the need of particular framework for excessive power, cooling systems and battery backup etc.

Non dedicated resources are used to constitute volunteer contributors clouds that can communicate with each other and also with other commercial clouds. So in order to achieve interoperability between volunteer contributor cloud and commercial cloud, and to deal with moving and reallocating data, we propose an XMPP based messaging middleware architecture.

This paper is structured as follows: the next section presents what is the need to establish a communication between volunteer clouds and commercial clouds and challenges for achieving the same, than we discuss literature review. Thirdly we present our proposed work and some possible solutions to these challenges. Finally we end up with a conclusion and future work.
Scientific or research projects that cannot be able to have commercial cloud services can use volunteer clouds where contributors and users voluntarily share their resources. For short and long term projects, our traditional IT companies have to spend a lot of time in deploying new hardware and software resources, by procuring and purchasing these new resources into their infrastructure.
Importance of volunteer computing:

- Volunteer computing is designed to offer massive computing power which is a result of a large number of PCs that exist in the world. This massive computing power is required to carry out scientific research projects and development.

- A research project that cannot afford expensive computing resources can make best use of volunteer computing as computing power offered by volunteer computing cannot be purchased, it can only be earned.

- People are now taking interest in various scientific and research projects and volunteer computing has made it possible.

Volunteer cloud computing offers cloud services that are based on non dedicated resources without charging and a way to cut down IT cost so that companies can benefit from the well planned budget and can make the best use of it. We can make use of volunteer clouds, if a company wants to endeavor a project for a short duration of time. So it's time to move complex solutions to a volunteer cloud that can offer comparatively much faster response to the needs and requirements of business and can help companies lower overhead.
Consider few scenarios:

- where an organization utilizes the services of a cloud provider A, but if the cloud provider A's server suffers a failure, then organization can make use of volunteer computing services offered by volunteer cloud and can even bring their problems to the cloud. So communication is needed so that clients can easily establish a communication with the volunteer cloud and can take benefit from its computing services.

- To effectively manage the volunteer cloud architecture, there should exists a proper communication channel so that various volunteer cloud entities can communicate with each other and can efficiently provide a coordination amongst the various components of the volunteer cloud.

  An enhanced interoperable environment should be made within the volunteer cloud architecture so as to manage and control the volunteer cloud entities. In order to perform a specific task there should exists a proper coordination between the components so that they can carry out any task.

- Suppose a large number of computing resources are there in an organization which are kept unused for a long time, then organization can contribute their idle resources to the volunteer cloud so that these resources can be utilized in a proper way.

So there is a need to have a strong, reliable, secure communication to provide an interoperable environment between volunteer cloud entities and also between volunteer cloud and its clients.

II. CHALLENGES

Important challenges [6] that should be taken into consideration in order to achieve better communication between volunteer clouds and commercial clouds are as follows:
One directional communication: When volunteer cloud entities interact with each other and also with other clients and commercial clouds, one way communication is the biggest hurdle. Because if two parties wants to communicate with each other effectively then there should be a proper two way communication so that

consumers and contributors can easily access and provide their computing resources and also various components of volunteer cloud can coordinate with each other so as to carry out any specific task. So our middleware should be such that it can help to establish a proper and clean two way communication.

Latency: interaction between nodes is somewhat shorter than interaction between clouds, so a middleware is required that can perfectly handle such delays. As we know all the clouds are different, but still there's one thing which is common among all of them & it's a general concept which says "Data is centralized but Users are distributed".
This concept therefore highlights the need of properly planned deployment which can thus avoid any significant latency issue between user & the application server. If we won't plan a proper deployment, it can increase the latency issue for the user

Availability: Resources should be made available easily as per the needs of the users. Enterprises concerns about whether volunteer computing resources will have sufficient availability or not and also find out for how much time the system is running perfectly by making use of presence information. So there should be a mechanism that will allow an entity to broadcast its presence information or availability to other entities through communication. So to manage the availability of volunteer cloud entities, presence information can be used.

Security: So when volunteer cloud entities communicate with each other or with other commercial clouds, clients, security is a matter of concern because volunteer cloud deals with highly distributed resources and it is also known as distributed cloud, so in order to achieve communication we need some effective mechanism that will provide support for authentication, encryption, data protection and integrity.

Compatibility If we analyze today's scenario, no 2 clouds are similar when talk about both the aspect:
1. Nature
2. IT

For example if we talk about Google's Cloud it is way too different than that of Microsoft's Cloud, which in turn different from any other cloud which exists today.Today users expect higher magnitude of cross compatibility between devices, between applications, between platforms & environment. But still there is something which makes each cloud entirely different, & surely it isn't much about the way the work on the inside, but the way interaction happens with the cloud, the way data gets into & out of cloud of each cloud, and management of functionality of each cloud is done very differently.

III. LITERATURE REVIEW

This section presents the following related work: volunteer cloud computing architecture and cloud@home architecture. Volunteer cloud computing architecture [3] contains three layers: a service layer, virtual layer and physical layer. Service layer, in this layer services are provided to customers via an interface which is based on SOA approach. Next layer is the virtual layer it provides multiple functionalities like task management and QoS management. Last layer is the physical layer that deals with resource aggregation, allocation and monitoring.

Cloud@home [7] [8] deals with the reuse of domestic computing resources in order to deploy voluntary clouds. By cloud@home users and contributors can share their resources and services of any kind. This logical abstract model defines some layers: software environment, software infrastructure, software kernel and firmware/hardware. Software environment layer deals with interoperability among clouds and also responsible for checking services like availability and reliability. Next layer, software infrastructure, it provides two basic services to end users: execution and storage services. Third layer software kernel, in order to manage execution and storage services at the above layer software kernel provides some mechanism and tools. Final layer firmware/hardware, for the implementation of execution and software services this layer provides the physical hardware resources to the upper layer. Even after these approaches, some challenges are still left that should be taken into account in order to achieve interoperability.

IV. PROPOSED WORK

A working procedure is presented over here to achieve better interoperable environment where volunteer cloud entities are free to communicate with each other and with other commercial clouds and clients also via XMPP protocol. In this regard, two different scenarios will come into play: first one deal with the communication between volunteer cloud entities using XMPP and second scenario deals with the

communication between clients and volunteer cloud. We now implement an XMPP protocol over middleware layer so as to form XMPP based messaging middleware layer that can provide enhanced interoperability and can help towards a solution for the challenges mentioned above.

4.1.XMPP: Extensible messaging and presence protocol (XMPP) is an open protocol for real time communications based on XML (extensible markup language) that works as a communication protocol over message oriented middleware. XMPP based messaging middleware architecture provides a wide range of services that deals with the issues that we have discussed earlier are interoperability, availability, faster two way communication, flexibility, security.

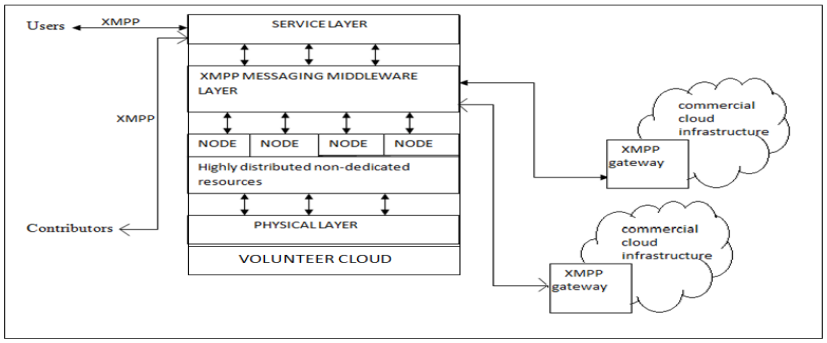

Fig. XMPP based messaging middleware architecture

4.2.First Scenario

XMPP protocol is implemented over middleware layer so as to form XMPP messaging oriented middleware layer that allow volunteer cloud entities or components to interact with each other by exchanging xml stanzas between components or entities. XMPP provides scalability which makes it easier for an infrastructure to extend by adding multiple resources, services and nodes to the network. It also overcomes the problem of single point of failure, as our volunteer cloud infrastructure can have multiple XMPP servers.

Volunteer cloud infrastructure contains various components or entities that further incorporate multiple sub-components, each of them is responsible for performing a particular task. These entities perform task scheduling, performance monitor, resource management, data management, network management etc. All the communicating entities of volunteer cloud can make use of XMPP </presence> stanza which is based on publish-subscribe method as it is responsible for managing and reporting the presence information. It allows an entity to broadcast its network availability to other entities through communication. This can be made possible through XMPP based messaging oriented middleware.

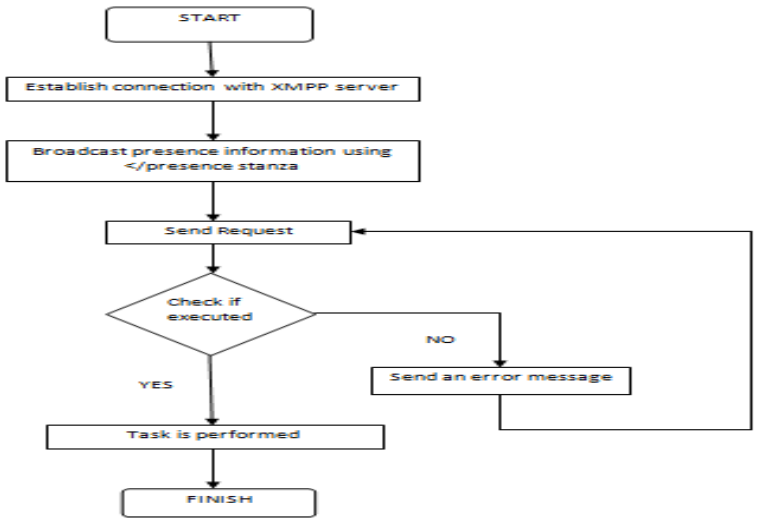

Fig.2. Flow chart for internal communication using XMPP

4.3. Second Scenario

In this scenario, a communication process is defined between volunteer cloud and clients by using XMPP. It describes how the clients can interact with the volunteer cloud so as to fulfill their requirements and needs. A scenario is defined with which clients can effectively consume and contribute their resources to the volunteer cloud.

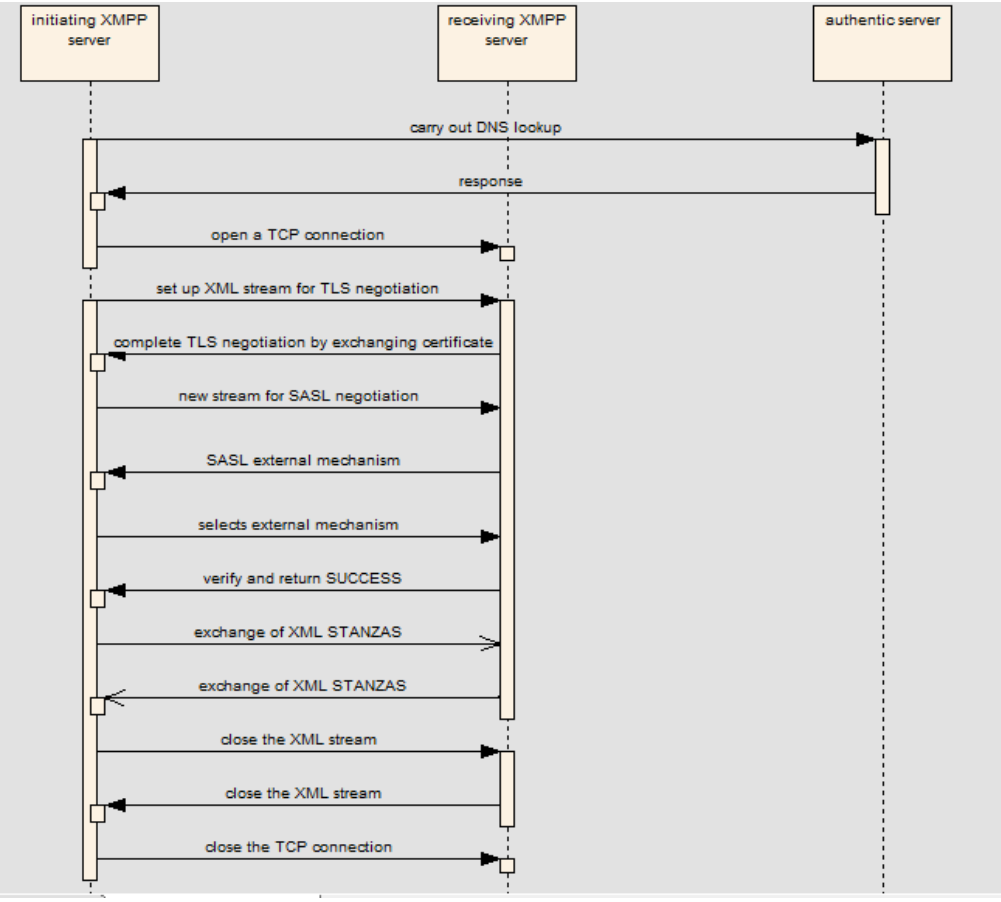

Fig.3. Sequence Diagram

One object will be our initiating XMPP server, it is considered to be client's server which is responsible for initiating the communication, second one is the receiving XMPP server, it is considered to be volunteer cloud server with which the interaction is being setup, and the last one is the authentic server which is equivalent to the Domain Name System.

- It will start with the initiating XMPP server carrying out DNS lookup with the authentic XMPP server.
- After receiving the response from the authentic server, initiating server establish a TCP connection with the server (receiving XMPP server) with which it wants to communicate.
- When the TCP connection is established its time to setup XML stream for TLS negotiation between the two servers.
- The two communicating server exchange certificates with each other in order to complete the TLS negotiation.
- Now a new stream is setup for the SASL negotiation then receiving XMPP server send a request for SASL External Mechanism to the initiating server, now the initiating server will select an External Mechanism and send it back to the receiving server. After the successful completion of the authentication process, the receiving server will send a SUCCESS response to the initiating server.
- After the authentication process both the servers are free to communicate with each other via exchanging XML stanzas.
- When they are done with the exchanging information and when their work is done, the initiating XMPP server can close the XML stream and TCP connection.

4.4. Implementation

Suppose ab.domainA.net is the initiating XMPP server or we can refer to it as server1 and domainB.net is the receiving XMPP server or we can call it as server2. Initially server1 has to perform DNS lookup with the authentic server and after clarifying a Service Record of _xmpp-server._tcp.domainB.net and then a request is sent from server1 to server2 in order to establish a TCP connection.

Now server1 set up an XML stream with server2 in order to carry out TLS negotiation.

```
S1: <stream:stream
      from='ab.domainA.net'
      to= 'domainB.net'
      version='1.0'
      xmlns='jabber:server'
      xmlns:stream='http://etherx.jabber.org/streams'>
```

Features of stream along with a stream header are sent as a response from server2 to server1

```
S2: <stream:stream
      from='domainB.net'
      id='hTiXkW+ih9k2SqdGkk/AZi0OJ/Q='
      to='ab.domainA.net'
      version='1.0'
      xmlns='jabber:server'
      xmlns:stream='http://etherx.jabber.org/streams'>
```

```
S2: <stream:features>
      <starttls xmlns='urn:ietf:params:xml:ns:xmpp-tls'>
        <required/>
      </starttls>
    </stream:features>
```

In order to carry out TLS negotiation, a STARTTLS command is transmitted from server1 to server2.

Now both the servers exchange certificates so as to complete TLS negotiation. After the successful completion of TLS negotiation, a new stream is set up by server1 for SASL authentication.

```
S1: <stream:stream
      from='ab.domainA.net'
      to='domainB.net'
      version='1.0'
      xmlns='jabber:server'
      xmlns:stream='http://etherx.jabber.org/streams'>
```

Features of stream (SASL External Mechanism) along with a stream header are sent as a response from server2 to server1.

```
S2: <stream:stream
      from='domainB.net'
      id='RChdjlgj/TIBcbT9Keu31zDihH4='
      to='ab.domainA.net'
      version='1.0'
      xmlns='jabber:server'
      xmlns:stream='http://etherx.jabber.org/streams'>

S2: <stream:features>
      <mechanisms xmlns='urn:ietf:params:xml:ns:xmpp-sasl'>
        <mechanism>EXTERNAL</mechanism>
        <required/>
      </mechanisms>
    </stream:features>
```

An external mechanism is selected by server1, this selected external mechanism is sent as a response to server2 along with an encoded identity.

```
S1: <auth xmlns='urn:ietf:params:xml:ns:xmpp-sasl'
          mechanism='EXTERNAL'/>eG1wcC5leGFtcGxlLmNvbQ</auth>
```

If the information given in the certificate matches with the encoded identity sent by server1, then server2 returns SUCCESS in response.After the authentication process both the servers (client's server and volunteer cloud server) are free to communicate with each other via exchanging XML stanzas. The communicating servers now can send any number of XML stanzas with each other.Suppose abc@domainA.net of domainA.net wants to establish an interaction with xyz@domainB.net of domainB.net for consuming resources that are present on the volunteer cloud's server which we denote here as domainB.net. Now domainB.net check if the requested resources are available, if available then it grants the resources to the requesting party. It means that these resources are logically present in domainA.net but are physically located on domainB.net.

If the servers of clients and volunteer cloud don't want to communicate any further, initiating XMPP server (server1) can close the XML stream by using handshake procedure.

```
S1: </stream:stream>
```

The stream is closed by volunteer cloud server (server2) as well.

```
S2: </stream:stream>
```

Finally, underlying TCP connecting can be closed by server1.

For clouds to interact with each other, a common format for messaging is used to allow the resources to **interoperate** with each other and manifest how their services can be utilized. On different cloud network, if resources are not implemented by XMPP and if the communication is to be established, then there is XMPP gateway that can convert XMPP to a foreign protocol. Any user who wants to communicate with the user on the other network, that user will first have to register with one of the gateways for the authentication purpose and then only they can start the communication.With emerging cloud computing technologies, XMPP is an ideal middleware protocol. Cloud services often use SOAP and some other HTTP-based protocols, but the problem with these protocols, is that they all offer one way communication that makes the services non real time, won't extend. There is another problem of long polling with HTTP based protocols, it means that server has to wait until it receives an update and as soon as it receives an update, the server sends the response and then only client can send further request. In this regard, we make use of XMPP as it offers faster and easy **two way communication** and also eliminates long polling. XMPP is also designed to be **scaled** and it provides SASL and TLS mechanism in order to provide robust **security.**

V. CONCLUSION

The present study aimed at developing an effective interoperable environment so that volunteer cloud entities can interact with each other and also with other clients via XMPP. It also aimed at providing solutions to the challenges that should be taken into consideration in order to achieve effective communication and also provides a well defined sequence of steps that are required to carry out communication. This study also brings out the magnificent role XMPP play so as to establish effective communication between the communicating entities. XMPP has proved to be the most suitable protocol and has emerged to offer solutions for the challenges and also for carrying out a better communication between volunteer cloud entities and also with other clients as it offers a wide variety of services that includes scalability, more security, implemented on a large scale, internet-scale technology, stateful, support a large number of end users, offer two way communication, enhance interoperability, extensible, reduce bandwidth and thousands of interoperable collection of code.

Feature:	XMPP	HTTP+ REST	HTTP+ SOAP
Security	TLS	SSL	SSL
Reliability	YES	NO	NO
Authentication	SASL	NO	NO
P2P Support	YES	YES	YES
Easy to Integrate with Web	YES	YES	YES
Easy to Integrate with Operators	EASY	N/A	N/A
Identity Management	YES	NO	NO
Bi-Directional Communication	YES	NO	NO
Overhead	HIGH	HIGH	HIGHEST

Fig.4. Feature Comparison

REFERENCES

[1] A. Andrzejak, D. Kondo, and D. P. Anderson, "Exploiting non dedicated resources for cloud computing," *2010 IEEE Network Operations and Management Symposium - NOMS 2010*, pp. 341-348, 2010.

[2] A. Marosi, J. Kovács, and P. Kacsuk, "Towards a volunteer cloud system," *Future Generation Computer Systems*, Mar. 2012

[3] Abdulelah Alwabel, Robert Walters and Gary Wills "towards an architecture for IaaS volunteer cloud" digital research 2012.

[4] Fernando Costa, Luis Silva, Michael Dahlin, " volunteer cloud computing: mapreduce over the internet", Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on.

[5] D. Kondo, B. Javadi, P. Malecot, F. Cappello, and D. Anderson, "Costbenefit analysis of Cloud Computing versus desktop grids", in *Proceedings of the 2009 IEEE international Symposium on Parallel & Distributed Processing*, pp. 1-12, May 2009

[6] T. Dillon, C. Wu, and E. Chang, "Cloud computing: Issues and challenges," in *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, 2010, pp. 27–33.

[7] Vincenzo D. Cunsolo, Salvatore Distefano, Antonio Puliafito and Marco Scarpa, "Cloud@Home: Bridging the Gap between Volunteer and Cloud Computing", in *ICIC'09 Proceedings of the 5th international conference on Emerging intelligent computing technology and applications*, 2009.

[8] V. D. Cunsolo, S. Distefano, A. Puliafito, and M. Scarpa, "Applying Software Engineering Principles for Designing Cloud@Home," *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pp. 618-624, 2010.