Evaluation of Scalability and Communication in MMOGs

Sarmad A. Abdulazeez, Abdennour El Rhalibi and Dhiya Al-Jumeily
Department of Computer Science, Faculty of Engineering and Technology
Liverpool John Moores University
Liverpool, UK
S.A.Abdulazeez@2013.ljmu.ac.uk, {A.Elrhalibi, D.Aljumeily}@ljmu.ac.uk

***Abstract*— Massively Multiplayer Online Games (MMOGs) can involve millions of synchronous players scattered across the world and participating with each other within a single shared game. One of the most significant issues in MMOGs is scalability and it is impact on the responsiveness and the quality of the game. In this paper, we propose a new architecture to increase the scalability without affecting the responsiveness of the game, using a hybrid Peer-to-Peer system. This mechanism consists of central servers to control and manage the game state, as well as super-peer and clone-super-peer to control and manage sub-networks of nodes sharing common regions of the game world. We use the OPNET Modeler to simulate the system and compare the results with client/server system to show the difference in delay and traffic received for various applications such as remote login, database, HTTP, and FTP sessions which are all part of an MMOG system. We use four scenarios for each system to evaluate the scalability of the system with different number of peers (i.e.125, 250, 500, and 1000 peers). The results show that the hybrid P2P system is more scalable for MMOGs when compared with client/server system.**

Keywords— MMOGs; Client/Server; Scalability; OPNET; P2P;

I. INTRODUCTION

Nowadays, Massively Multiplayer Online Games (MMOGs) are becoming more and more popular in the game industry. The key important feature in these games is the large number of players, involving millions of players, who play concurrently over the Internet. The main reason that makes MMOG's popularity is rapidly increasing a few years ago is the experience of playing with other human players. A good example of successful MMOG is World of Warcraft (WoW) [1]. Games' companies suffer from scalability issues that must be considered in all design phases. The scalability level of games is defined as the ability to manage the increasing level of interaction demands without dramatically deteriorating the quality of the playing experience. Thus, there is a need for processes and network architectures that can deal with the resulting overload, whilst preserving an accurate game state, decreasing latency, and minimizing the bandwidth consumption. Traditionally, MMOGs are distributed systems using a client-server architecture, in which a centralised server has the authority to perform the majority of the game logic and then send a game state update to the clients for rendering. The responsibility of client is collecting the user commands and sends them to the server for processing. In this architecture, the requirement of computational power and bandwidth for a single client is minimal. However, considerable resources and consecrated support staff are required for a large number of players at the server side. In most case this can lead to a bottleneck restricting the scalability of the game. In Client-Server architecture, the scalability is achieved by using server clusters or multi servers. For example, the game world is divided into several zones and each zone controlled by one server. However, the possibility of congestion for bandwidth stress at the server side in addition to a central point of failure cannot be avoided. Furthermore, this architecture lacks flexibility and the resources used by the servers have to be over-provisioned to deal with peak loads. In this paper we propose a new hybrid architecture using peer-to-peer and server to alleviate the scalability issues involved in MMOGs. The paper starts by proposing new architecture and carry out experiments to compare it with the corresponding client server architecture. The remaining of the paper is organised as follows. Section II introduces the background of subjects related to the research area. Section III presents a review of the state-of-the-art. Section IV discusses P2P MMOG peer distribution and communication protocols. Sections V, VI, and VII describe the scalability, responsiveness, and reducing the traffic volume respectively. Section VIII illustrates the simulation results. Finally, section IX dedicated for the conclusion.

II. BACKGROUND

In this section we discuss a number of concepts related to this research.

A. Client-Server Architecture

The client-server model has been used as a traditional method of designing networked multiplayer games, due to the simplicity and efficacy of the approach, with many examples of its usage. It consists of one or more authoritative game state servers, and one or more clients. The client-server mechanism have been explained in detail in [2].

B. Peer-to-Peer Architecture

Peer-to-Peer (P2P) system refers to a distributed system whereby every peer in the network takes equal status and control. All machines are treated as clients and servers. Recently, the use of P2P networks have been growing fast and also more widespread in different applications of the Internet. Peer-to-peer architecture have been explained in detail in [2].

C. Hybrid Peer-to-Peer Architectures

Hybrid Peer-to-Peer system is a combination of a P2P architecture and a server-based architecture. This means that a certain quantity of workload can be assigned to the server or servers depending on the infrastructure of the game, however, other computations will be deployed to peers. Hybrid P2P architecture does not have the single point of failure compare with pure P2P architecture. This system can be divided into three categories based on what are handled by the P2P system [3].

- Cooperative Message Dissemination: The game state is maintained by one server or multiple depending on the server-based architecture. However, it uses a P2P approach to update game dissemination. Usually, players send their interactions and queries directly to the server.

After execution, the server uses a P2P multicasting technique for sending the updates to all players.

- State Distribution: Distribute the game state between peers. Peers responsible for execution of player actions because they hold primary copies of objects, but, all or part of the communication among peers might be managed by the one server or more. Furthermore, the server is responsible for centralised processes such as authentication, and control of joining and leaving of players. This mechanism achieves good scalability by distributing the cost of state execution among peers.
- Basic Server Control: P2P overlay responsible to do both message dissemination and state distribution. The essential role of the servers is to keep highly sensitive data, such as players' logins, payment information, as well as players' progress and game state. The responsibility of the server is to control joining and leaving of players. However, servers do not preserve games state or execute state dissemination.

III. Related works

Scalability is a phenomenon observed in many natural and artificial systems such as MMOGs. With huge number of players, MMOGs are becoming increasingly common in the game world. However, they still suffer from scalability problem. Some researches tried to solve this problem. Hu and Liao [4] proposed a distributed peer-to-peer (P2P) architecture to deal with the scalability issue based on Voronoi diagram [5]. The scalability can be achieved by dramatically reducing server load with a single lightweight server. In fact, the server is used just for authentication, and is not required after login [4]. Hu et al. [6] extended the previous work and proposed the Voronoi-based Overlay Network (VON) scheme to preserve the P2P architecture with small overhead, and achieves high scalability through bounding resource use at each player peer. Glinka et al. [7] proposed a new middleware system called RTF (Real-Time Framework) for developing real-time MMOGs. RTF can be used for automatically distributing the game state among participating servers, and provides effective communication and concurrence between game servers and clients [7]. Gupta et al. [8] propose a novel method for distributing consistency protocol and system architecture called SEMMO (Scalable Engine for MMOs). In this research, the engine authorises the clients to do all computations in local computer, however, the central server only determines the serialisation order of game actions [8]. Mildner et al. [9] propose a design for real-time massively multiplayer games to satisfy the high demands on scalability and responsiveness. In this research, the authors used a fully distributed P2P overlay system with a dynamic connection scheme. This system supports a big number of players by utilising local interests in the virtual world. As well as using a Geocast algorithm for sending messages to an arbitrary set of players based on their positions in the networked virtual environments (NVE) [9].

IV. P2P Communication Protocol for MMOGs

Over the last few years, there have been different P2P communication protocols for MMOGs and also for virtual worlds. These protocols have been developed in many active projects. The following sections will describe these categories: Application Layer Multicast (ALM) based protocols, super-peer based protocols and mutual notification based protocols.

A. Application Layer Multicast Based Protocols

Standard Application Layer Multicast (ALM) methods are used for distributing game events and messages [10]. ALM provides the mechanism to multicast as an application level service instead of a service at lower levels of the network stack [11]. Commonly, the game space is divided into a number of subspaces. Each subspace is represented by a special multicast group. When an event occurs inside one subspace, the corresponding message is sent through the subspace's multicast group just for players interested in events in this subspace. A player is usually only interested in events that happen inside an Area of Interest (AoI). A typical example for an ALM based protocol is SimMUD [12]. The game space in SimMUD is divided into fixed zones with unique IDs.

B. Super-peer Based Protocols

Similarly to ALM based protocols, the game space is commonly divided into subspaces. However, the subspaces can be a fixed size [13] or sometimes dynamically based on player density [14]. Each subspace is assigned to a responsible peer called super-peer. The responsibility of super-peer is receiving all game event messages related for his subspace and distributed to all subspace players. Players can be registered at any interest super-peer of the subspaces. In dynamic subspace sizes, super-peer overloading is prevented by determining the number of players per subspace. If the number of players inside a certain subspace is too high, it will be divided into smaller subspaces. A useful example for this type of protocols is the publish/subscribe based technique [13].

V. Scalability

Scalability is a desirable feature of network, system, or process. It is one of the most important challenges in MMOGs development and deployment. Scalability indicates the ability of a system to accommodate an increasing number of players in the game world, without affecting the performance of the game. The scalability is a crucial aspect to long term success when the system is subject to increasing demand. Scalability in computer science field generally divided into two main aspects; structural scalability and load scalability [15]. Structural scalability is the ability of the system to expand in the selected dimension without main modifications to the architecture of the system such as adding nodes (peers) to a system without the need to modify the system. Load scalability means using of available resources in a way that allows dealing with load increase, such as the increase of number of players in game world. In our research, we have used the load scalability to increase the number of players in the region of game world. Some factors that can effect load scalability are shared resource scheduling, scheduling the category of resources in ways that lead to an increase own usage, the increase of interconnection of player inside the region, wrong selection of region's super-peer and clone-super-peer, and lack of exploitation of parallelism in efficient way.

VI. Responsiveness

In MMOG, responsiveness indicates to the ability of the game applications to deploy updates among players in an efficient manner without affecting the gameplay, without prejudice in the virtual world such as stop the game a little bit whilst waiting for update. In other words, "Responsiveness means the delay that it takes for an update event to be registered" [16]. In traditionally architecture for MMOGs, players wait until received updates from the server. This will lead to high latency, especially when the position of the player is far from the game server, and thus effect on the performance of the game. Instead, the client locally predicted the position of the player whilst waiting for a response from the server, applying corrective state rewinds to the local view of the world so that the player's view was still authoritatively controlled by the server. To achieve good responsiveness, the game updates must be received to players quickly. In this case, the peers must include some computation to decrease the bandwidth and latency requirements such as dead reckoning, message aggregation, and message compression.

VII. Reducing the Traffic volume

Reducing traffic of the network have been concentrated on two prime dimensions: Reducing the number of parties (peers, super-peer, or clone-super-peer) that sent a message to it, and reducing the number of messages required sent from the server to the parties in the game world. MMOGs are a group of state information, and a process for updating those state. There are many state changes when for example two avatar are interacting with each other. The game server can be changing just the state of the two avatars. Moreover, decision of the combat among those avatars is possibly not relevant and effect to other avatars in the game world. There are two key solutions to exploit this behaviour: region-based computation and message deployment, and Area of Interest Management (AoIM). In region-based mechanism, the game world is divided into regions. In our research, we use the geographical based region to divide the game world, and each region manages the state and control the behaviour of players by super-peer and clone-super-peer. The message deployment and game states are done through both super-peer and clone-super-peer. This mechanism will reduce the network traffic compare with sending the message and game states to each player in the game world. However, in Area of Interest Management (AoIM) mechanism, a message is only sent to the player that needs this information, in order to display an accurate state of the world to the players. In this research, two types of traffic are used: explicit traffic and background traffic as detailed in the next subsections.

A. Explicit traffic

Explicit traffic is packet-by-packet basis traffic. It models for packet creation, queuing, transmission, and demolition explicitly traffic through a discrete-event simulation process. The packet is created by using several applications such as (remote login, database, web services, HTTP etc.). Therefore, explicit traffic modelling is one of the most important models that provide accurate results due to it is provided the details of protocols. In addition, explicit traffic can be used to model all the details in discrete-event simulation, but it takes more time and resources. Using background traffic in order to reduce the computational burden, this will be introduced in the next section. There are three main mechanisms for explicitly traffic modelling [17]:

- Packet generation: The basic method to generate streams of generic packets is configuring certain node objects.
- Application requests: Create application requests to represent the traffic flowing among two nodes in the network.
- Application traffic models: Include a group of models for generating traffic based on standard applications such as (remote login, database, web services, HTTP etc.).

B. Background Traffic

Explicit traffic provides an accurate result but it is not useful for large network because it allows detailed control of the modelled protocol and takes a very long time and require further computing resources. Therefore, use background traffic has been introduced to solve these problems. Background traffic described as analytically modelled traffic rather than discrete-event simulation. Therefore, it consumes a very small amount of computing resources and takes considerably less time. Use hybrid simulation system to combine explicit traffic and background traffic. The performance of explicit traffic at this case is really influenced by the extra delays analytically calculated and produced by the background traffic load. However, background traffic technique is not beneficial for all simulation scenarios because it is based on analytical models. The implementation of background traffic is restricted to statistics of protocols at higher layers than IP, such as TCP, UDP, IP traffic, and applications. Background traffic can takes three different forms [18]:

- Traffic flows: Describes an end-to-end flow of traffic from a source to one or more destination nodes. In this type of traffic, you can create traffic flows manually by using traffic flow objects. As well as, you can also import traffic flows from text files and spreadsheets.
- Baseline loads or sometime called "static background utilization": The traffic is represented as a background load on a link, node, and connection. In this type of traffic, a traffic load is "static" and applies to one object, on the contrary a traffic flow, which can extend to multiple nodes and links.
- Application demands: Use application demands to represent background traffic flowing between two nodes. Furthermore, you can also configure application demands to be purely discrete- event simulation (explicit traffic), or hybrid simulation.

VIII. Simulation

OPNET Modeller 14.5 is used to model and simulate the system. OPNET has possibly the largest selection of ready-made protocol models [19].

A. Network Topology of Scenarios 1

We have used two architectures to simulate Client/Server system and Hybrid P2P system. This first architecture consists of the following devices: game server, two Cisco C400 Router, IP Cloud, Ethernet Switch, and Ethernet Workstations as shown in figure 1. The connection mechanism between these devises is described as follow: The game server connects with the IP Cloud through Cisco C400 Router (server gateway). The IP Cloud connects with the Ethernet Workstations (Peers) through both Cisco C400 Router (Client gateway) and Ethernet Switch. Using Ethernet 100BaseT link to connect between server and server gateway, also between the client gateway and Ethernet switch. Using PPP DS1 link to connect between Router and IP Cloud, also between IP Cloud and Client gateway. Use Ethernet 10BaseT link to connect between the Ethernet Workstations and Ethernet Switch. In client/server architecture, all the applications that showed in table 2 are manage and control by the server. The number of Ethernet Workstations was (125, 250, 500, and 1000) at each scenario.

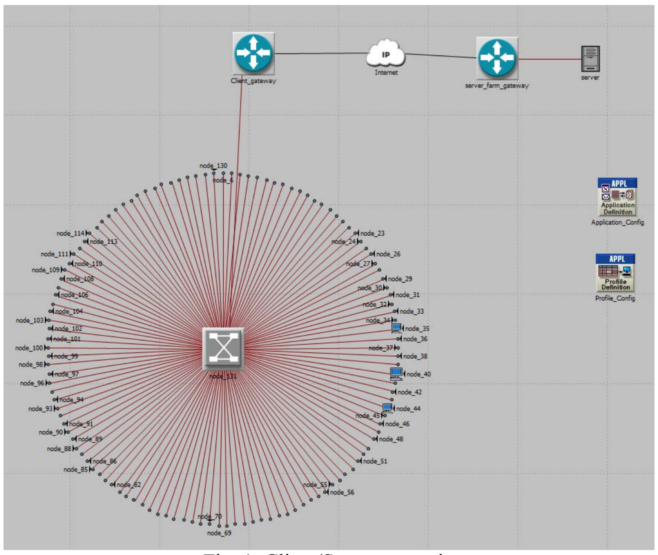

Fig. 1. Client/Server scenario

B. Network Topology of Scenarios 2

The second scenario has the same devices in scenario 1 but with a little bit change in the zone area by adding super-peer and clone-super-peer to the region. Each region has 60 peers connected to both super-peer and clone-super-peer using the super-peer gateway as shown in figure 2. In Hybrid P2P architecture, only the remote login manages and controls by the server, the others applications control and manage by the region super-peer and clone-super-peer.

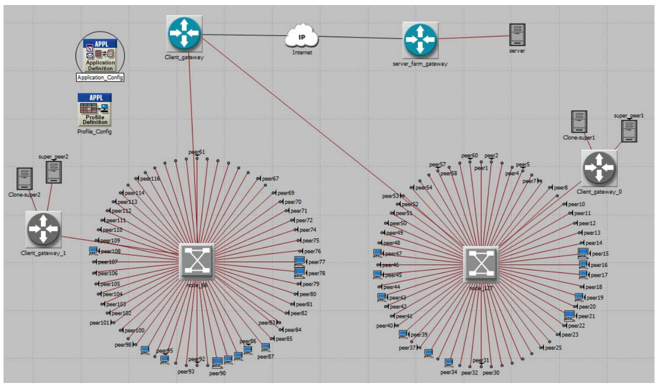

Fig. 2. Hybrid P2P scenario

The applications that were used during the simulation for all scenarios are Remote Login, Database, HTTP, and FTP as shown in the table 1.

TABLE 1: THE APPLICATIONS USED IN SIMULATION

Name of Applications	Start Time Offset (second)	Duration (second)	Repeatedly
Remote login	Constant (1)	Constant (300)	Once at Start Time
Database	Constant (1)	Constant (300)	Once at Start Time
HTTP	Constant (1)	Constant (300)	Once at Start Time
FTP	Constant (1)	Constant (300)	Once at Start Time

The duration of the simulation is set to 1200 seconds for all the profiles used in the system for the scenario with 125 peers, however, the duration of the simulation duplicated when the number of peers are duplicated for example when the number of peers is 250 peers the simulation duration will be 2400 second and so on. The simulation parameters that used in the simulation system described in table 2.

TABLE2: SIMULATION PARAMETERS

Applications	Parameters	Unite
Remote login	Delay Traffic Received	Seconds Bytes/sec
Database	Traffic Received	Bytes/sec
HTTP	Delay Traffic Received	Seconds Bytes/sec
FTP	Delay Traffic Received	Seconds Bytes/sec

C. Analysis of the Results

In this section, we have analysed network performance under simulation scenarios from the view of five initial parameters.

a) Overall Delay

This parameter is defined as the overall end-to-end delay for all packets received by the station. End- to-end delay for the application that used during the simulation is measured from the time from source to destination. Figure 3 shows the overall delay for the client/server architecture compare with the overall delay of the hybrid P2P architecture with 125, 250, 500, and 1000 peers without adding any background traffic to the connection link. Figure 3 illustrates that the delay for the client/server system is greater than the delay of the hybrid P2P system for all application used in the simulation except the remote login application because this application is the same for both architectures. All the players must login in the game server at the first time. This is the reason that make the delay nearly equal for the remote login application for all four scenarios. However, database query delay for client/server system is four times greater than the delay of database query in hybrid P2P system. due to all players in client/server architecture send their queries of database directly to the server, however, the players in hybrid P2P architecture sent queries of databases to the region's super-peer and clone-super-peer, and this thing helps to decrease the overall delay for hybrid P2P system.

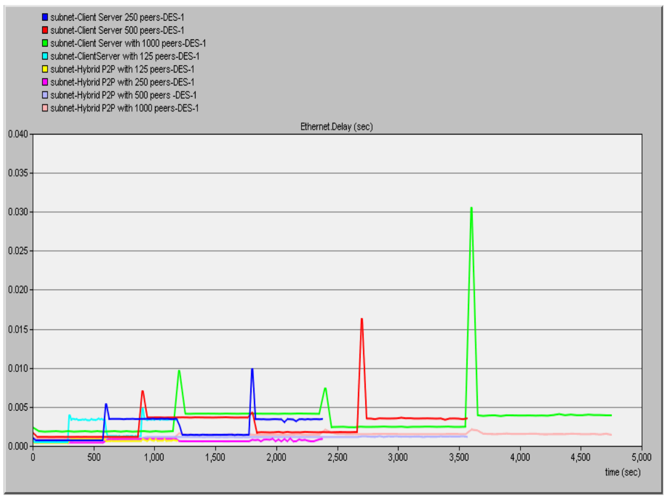

Fig. 3. The overall delay for client/server and hybrid P2P architecture with 125, 250, 500, and 1000 peers

b) Traffic Received for Remote Login

The Remote Login application allows the user to connect with a remote server and implement different operations on it by issuing commands from a local machine. However, traffic received is the average number of bytes per second forwarded to all remote login application by the transport layers in the network. Figure 4 illustrates the traffic received for the remote login application with 125, 250, 500, and 1000 peers for both client/server and hybrid P2P architecture. The figure showed that the traffic received for the remote login application is nearly equal. Due to the remote login is managed by the server for both scenario. However, there is a very large variation between the traffic Received for Remote Login for scenario with 500 peers and the scenario with 1000 peers for both architectures up to five times.

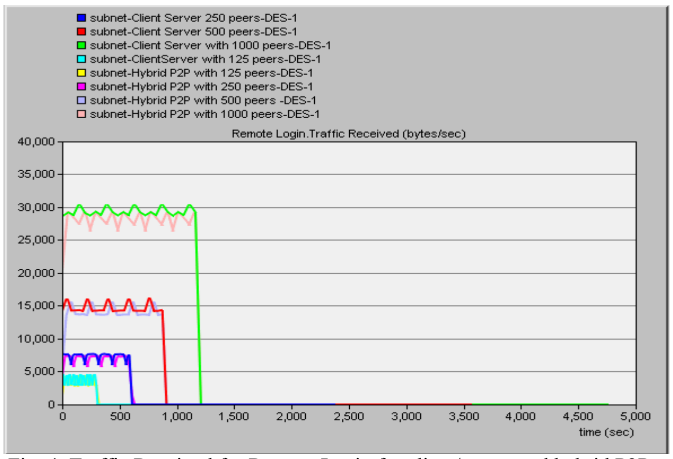

Fig. 4. Traffic Received for Remote Login for client/server and hybrid P2P architecture

c) Traffic Received for Database Query

The database query operates on retrieving data from the database. It consists of a query message that holds the database request and a response message that holds the data. The traffic received for database query defined as the average number of bytes per second forwarded to all database query application by the transport layers in the network. Figure 5 displays the traffic received of database query with 125, 250, 500, and 1000 peers for both client/server and hybrid P2P architecture. The figure explains the traffic received of database query for client/server system is much greater than the traffic received of database query for hybrid P2P system up to nearly five times with 500 peers, however, traffic received for database query up to highest level more than ten times in client/server architecture with 1000 peers compare with hybrid P2P with same number of peers. This shows that when the number of players increased, then the traffic received of database query will increased while it remains stable in hybrid P2P architecture. Due to the database query in hybrid P2P system is manage and control by the region super-peer and clone-super-peer.

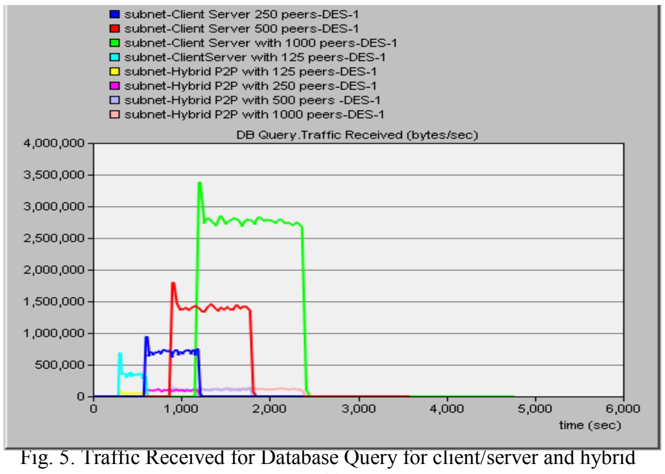

Fig. 5. Traffic Received for Database Query for client/server and hybrid P2P architecture

d) Traffic Received for HTTP

The HyperText Transfer Protocol (HTTP) is the Internet browsing activity where a client node periodically contacts web servers to get web pages. HTTP traffic received is the average number of bytes per second forwarded to the HTTP application by the transport layers in the network. Figure 6 illustrates the difference between the HTTP traffic received for client/server and hybrid P2P system with 125, 250, 500, and 1000 peers. The figure shows the clearly increase of HTTP traffic received to reach the highest peak with 340.000 bytes/second in the scenario four with 1000 peers, while it remains stable in hybrid P2P architecture for all scenarios by less than 20.000 bytes/second.

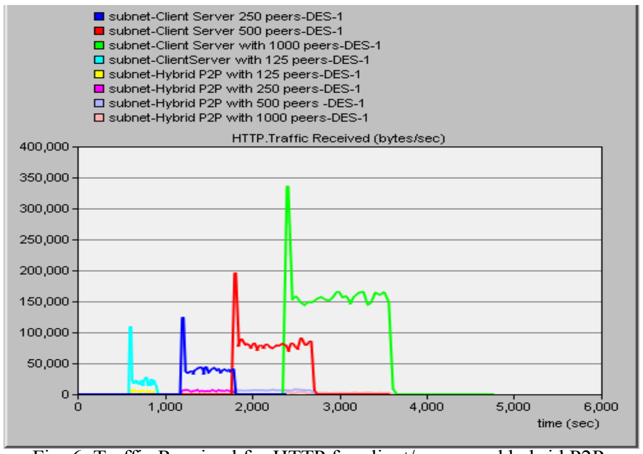

Fig. 6. Traffic Received for HTTP for client/server and hybrid P2P architecture

Figure 6 shows that the delay of HTTP application for hybrid P2P system is slightly less than the delay of HTTP for client/server system. Due to the fact that the amount of data used in HTTP application is few in comparison with the enormous amount of data in FTP application, which made the contrast in overall delay is comparatively few.

e) Traffic Received for FTP

File Transfer Protocol (FTP) has two primary operations for data transfer. The first one is FTP put operation uploads a file onto the FTP server; however the FTP get operation downloads a file from the FTP server onto the client node. Figure 7 shows the FTP Traffic Received for client/server and hybrid P2P system. The figure indicates the big difference between the FTP traffic received for client server and the FTP traffic received for hybrid P2P system at the start time of the FTP application. The FTP traffic received for hybrid P2P is nearly equal to 50.000 bytes/second at the starting time of the application, while rapidly increasing even up to the highest level with at the fourth scenario with nearly 340.000 bytes/second. These results will make the client/server architecture inappropriate to some extent to design MMOGs.

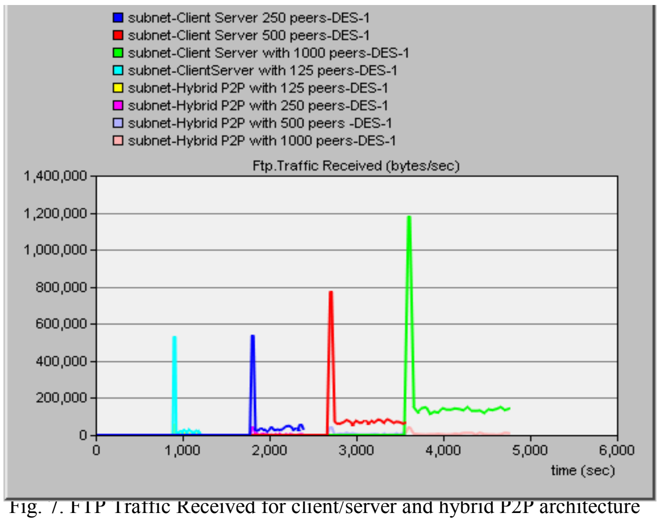

Fig. 7. FTP Traffic Received for client/server and hybrid P2P architecture

D. Background traffic

Figure 8 shows the effect of the background traffic on the link connection in client/server system with 125, 250, 500, and 100 respectively. Table 3 shows the background traffic information that used on the all link connection between the devices in both scenarios, using uniform X interval 300 seconds/step for both sides.

TABLE3: THE BACKGROUND TRAFFIC INFORMATION

Seconds	Bits/second
0.0	500.000
300	100.000
600	500.000
900	100.000
1200	500.000

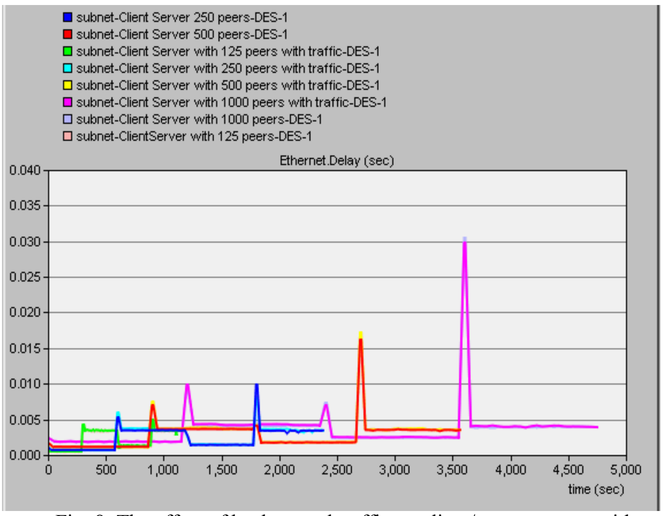

Fig. 8. The effect of background traffic on client/server system with 125 and 250 clients

However, the effect of the background traffic on hybrid P2P system. Figure 9 shows the effect of the background traffic on the hybrid P2P system with 125, 250, 500, and 100 respectively.

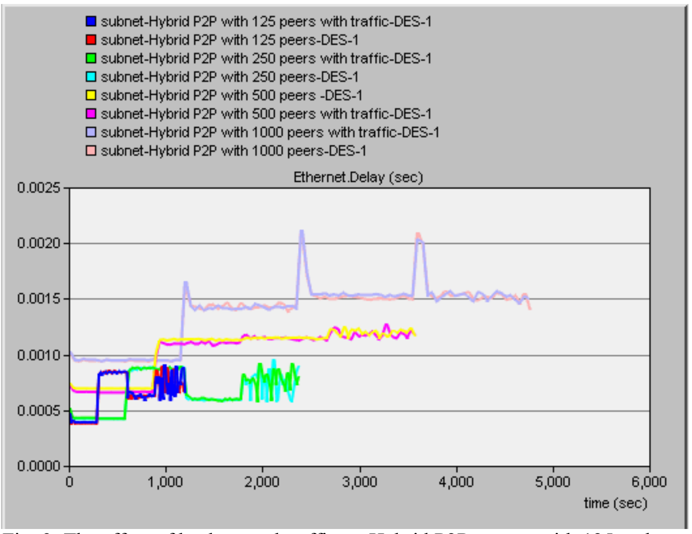

Fig. 9. The effect of background traffic on Hybrid P2P system with 125 and 250 peers

IX. CONCLUSION

The increasing number of players in MMOGs that support millions of players make the game has several problems that relate to the scalability. In this paper, we have discussed the MMOGs architectures and some related works to deal with the scalability issue in MMOGs. We have proposed a new method to cope with the scalability problem based on hybrid P2P system. The systems have been simulated using OPNET Modeller with different number of peers to validate the scalability of the proposed system. The results have been compared with the client/server system to show the difference in delay and traffic received for various applications such as remote login, database, HTTP, and FTP. The results show that the hybrid system is more scalable for MMOGs compare with client/server system.

REFERENCES

[1] M. T. Wyman, “World of Warcraft.” pp. 23–38, 2011.

[2] S. A. Abdulazeez, A. El Rhalibi, M. Merabti, and D. Al-jumeily, “Load Balancing Techniques for Massively Multiplayer Online Games□: A Survey,” PGNet, no. Liverpool, UK, 2014.

[3] A. Yahyavi, B. Kemme, and F. Fantasy, “Peer-to-Peer Architectures for Massively Multiplayer Online Games□: A Survey,” vol. 46, no. 1, 2013.

[4] S.-Y. Hu and G.-M. Liao, “Scalable peer-to-peer networked virtual environment,” Proc. ACM SIGCOMM 2004 Work. NetGames ’04 Netw. Syst. Support games - SIGCOMM 2004 Work., p. 129, 2004.

[5] S.-Y. Hu, S.-C. Chang, and J.-R. Jiang, “Voronoi State Management for Peer-to-Peer Massively Multiplayer Online Games,” 2008 5th IEEE Consum. Commun. Netw. Conf., pp. 1134–1138, 2008.

[6] J. Chen and T. Chen, “VON: A Scalable Peer-to-Peer Network for Virtual Environments,” no. August, pp. 22–31, 2006.

[7] F. Glinka, A. Ploß, J. Müller-iden, and S. Gorlatch, “RTF□: A Real-Time Framework for Developing Scalable Multiplayer Online Games,” pp. 81–86.

[8] N. Gupta, A. Demers, and J. Gehrke, “SEMMO□: A Scalable Engine for Massively Multiplayer Online Games [Demonstration Paper],” pp. 1235–1238, 2008.

[9] P. Mildner, T. Triebel, S. Kopf, and W. Effelsberg, “A scalable Peer-to-Peer-overlay for real-time massively multiplayer online games,” Proc. 4th Int. ICST Conf. Simul. Tools Tech., 2011.

[10] D. T. Ahmed and S. Shirmohammadi, “A fault tolerance procedure for P2P online games,” 10th Int. Conf. Inf. Sci. Signal Process. their Appl. ISSPA 2010, no. Figure 1, pp. 614–617, 2010.

[11] M. Hossein, D. T. Ahmed, S. Shirmohammadi, and N. Georganas, “A Survey of Application-Layer Multicast Protocols,” vol. No.3, pp. 58–74.

[12] B. Knutsson, H. Lu, W. Xu, and B. Hopkins, “Peer-to-Peer Support for Massively Multiplayer Games,” IEEE INFOCOM, 2004.

[13] S. Yamamoto, Y. Murata, K. Yasumoto, and M. Ito, “A distributed event delivery method with load balancing for MMORPG,” Proc. 4th ACM SIGCOMM Work. Netw. Syst. Support games - NetGames ’05, p. 1, 2005.

[14] C. GauthierDickey, V. Lo, and D. Zappala, “Using n-trees for scalable event ordering in peer-to-peer games,” Proc. Int. Work. Netw. Oper. Syst. Support Digit. audio video - NOSSDAV ’05, p. 87, 2005.

[15] a Bondi, “Characteristics of Scalability and Their Impact on Performance,” Proc. 2nd Int. Work. Softw. Perform., pp. 195–203, 2000.

[16] J. Smed, T. Kaukoranta, and H. Hakonen, “A Review on Networking and Multiplayer Computer Games Timo Kaukoranta,” Turku Cent. Comput. Sci. TUCS Tech. Rep. No 454, no. Finland.

[17] Z. Lu and H. Yang, Unlocking the Power of OPNET Modeler. 2012.

[18] S. S. Sethi and V. Y. Huatyshin, The Practical OPNET® User Guide for Computer Network Simulation. CRC Press, Taylor & Francis Group, 2013.

[19] “OPNET Modeler Simulation Environment.” [Online]. Available: http://www.riverbed.com/products/performance-management-control/opnet.html?redirect=opnet. [Accessed: 26-Sep-2014].