Collaborative Applications Platform Based on Secure P2P Networks

Chun-Hsin Wang, Chia-Chun Lien and Shao-Hua Lin
Department of Computer Science and Information Engineering
Chung Hua University, Hsinchu, Taiwan, R.O.C.
E-mail: chwang@chu.edu.tw; m10102025@chu.edu.tw; m10102026@chu.edu.tw

***Abstract*—The feature of collaborative applications is to utilize various resources distributed over Internet (or intranet). It is a good choice to develop collaborative applications based on Peer-to-Peer (P2P) networks, which can be applied to integrate resources over peers. Most of popular P2P networks focus on files or content sharing and security problems are not seriously considered, so they are not good enough for developing collaborative applications. In this paper, secure P2P networks are designed by authentication of joining peers and encrypted data communication. To prohibit misuse of resources, peers are classified into three levels with different priorities. Based on secure P2P networks, a scalable and flexible collaborative application platform composed of core services and user defined services is built. Various resources provided by peers can be easily used by service execution. Two collaborative applications are designed to demonstrate the use of services over peers. It can be expected that more creative collaborative applications will be designed based on the proposed platform.**

Keywords-Collaborative applications platform; Secure P2P Networks.

I. INTRODUCTION

Network applications have great variation from conventional client-server model to P2P networks. In client-server model, direct communications are not allowed between any two clients, but they may occur between any two peers in P2P networks. The P2P technology has been widely applied to the integration and sharing of network resources. The shared resources can be provided by any peers joining the system instead of few dedicated servers. More creative network applications need resources distributed over peers to work well. For example, some collaborative approaches [1-4] defend Distributed Denial of Service (DDoS) attacks and worm containment from Internet. In [5], collaborative application is applied for data fusion, in which data are from different radars, sensors, and processing nodes. P2P-assisted cloud [6-7] or cloud-assisted P2P [8-10] collaborative models are trying to integrate the services provided by P2P network systems and cloud computing systems. Theses examples reveal that network applications tend to prefer collaboration of joining nodes and require multi-types resources support such as files, content, storage, or computing power.

It is a good choice to develop collaborative applications based on P2P networks. Most of popular P2P network systems focus on files, content sharing, or computation power. For example, BitTorrent-like P2P systems [11] are implemented for file sharing, while PPLive [12] and PPStream [13] are designed for streaming content sharing. The well-known SETI@home [14] project tries to find intelligent life outside Earth by stealing computation power over peers. Security problems in P2P networks are not seriously considered. Peers can join in these P2P networks without any authentication. Malicious nodes can easily launch attacks, such as sybil attacks [15], which generate a large number of shadow identifies that control system operations. Due to the weak of security problems and limited types of resources sharing, current P2P networks are not good enough for developing collaborative applications.

For development of collaborative applications, it is important for peers to easily provide services. The services provided by peers can be defined as executable software modules which can be executed to satisfy requests from other peers. This feature is to fit the collaborative applications which can request some selected peers to provide different types of services for approaching purposes of applications. As a scenario may occur, peers have resources, but can not provide requested services. Some peers in the system should have the ability to publish new services to the requested peers. The available services must be scalable and allowed to be defined by users. The open source vuze P2P system [16] has the similar concept, which is a BitTorrent-like file sharing system implemented in Java. Users can develop and plug their software modules into the system, but current plug-in software modules still focus on enhancement of files, content sharing, or improvement of user friendly interfaces.

In this paper, P2P network systems are enhanced by authentication of joining peers with three levels of priorities and encrypted data communication. Based on secure P2P networks, a scalable and flexible collaborative application platform composed of core services and user defined services is built. Various resources provided by peers can be easily used for service execution. The system model is shown in Fig. 1. Graphic user interface of peers is implemented in shell-like commands to request other peers to execute services. Two collaborative applications are designed to demonstrate the use of services distributed over peers. More creative collaborative applications based on the proposed platform can be easily designed, such as distributed computing, location-aware applications, etc.

The rest of the paper is organized as follows. The proposed secure P2P networks are described in Section II. The collaborative applications platform and its services are described in Section III. Some possible collaborative applications based on the proposed platform are presented in Section IV. Web-based online management system is introduced in Section V. Finally, some concluding remarks and future work are given in Section VI.

 ISBN: 978-1-61208-300-1

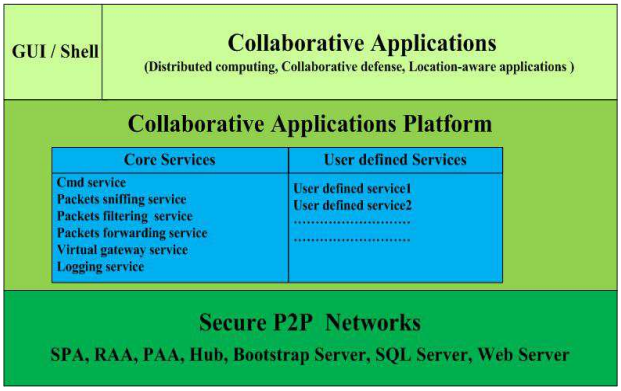

Figure 1. System model

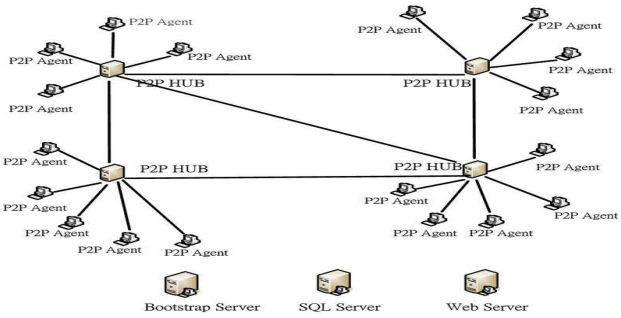

Figure 2. Overlay networks of secure P2P networks

II. SECURE P2P NETWORKS

In this section, we first introduce the model of the proposed secure P2P networks. Then, we describe the functions of components in it, system operation and implementation in detail. The proposed secure P2P networks (Fig. 2) are similar to KaZaA [17]. The main members of system are bootstrap server, P2P Hub nodes, and P2P agent nodes. To improve security of P2P networks, anonymous peers are not allowed. A database server (microsoft SQL) and web server are set up to maintain and manage related information of peers.

Peers are classified into three categories, namely Service Passive Agent (SPA), Request Active Agent (RAA), and Publish Active Agent (PAA). SPAs can provide services for RAAs and PAAs which they can issue requests to other peers. Besides the functions of SPA and RAA, PAA can publish new services to extend core services. The role of P2P Hub node is similar to cluster leader in KaZaA, which maintain the IP addresses of its members and associated resources-sharing. Hub nodes are selected from agents instead of hardware appliances. It means that some of agents have also to be roles of Hub nodes. Based on the open source library Lidgren [18], agents, Hub nodes, and bootstrap server are implemented in .net framework 4.0.

A. Bootstrap Server

The software architecture of bootstrap server and relationship of agents are shown in Fig. 3. Two components, "bootstrap controller" and "website controller", can startup bootstrap service by graphic user interface in bootstrap server and by a online browser respectively. The Hub module will be started in a agent only when it is selected to be a Hub node. To maintain information of the owned agents(members)

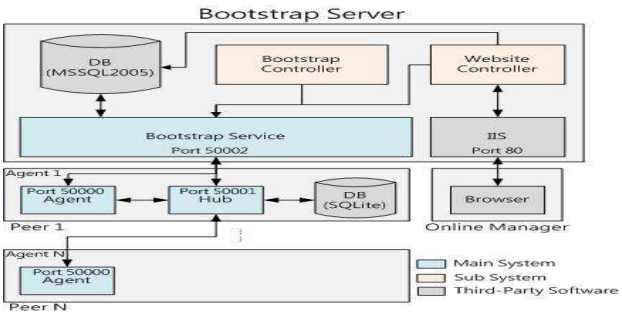

Figure 3. Software architecture of bootstrap server

and how much data traffic is exchanged, each Hub node will set up a file-based SQLite [19] database. The maintained information in Hub nodes will be updated to the database in bootstrap server periodically. In our system, bootstrap server maintains the authorities and locations of joining peers, the list of existing Hub nodes, log information of peers, private/public keys for secure communication, and how much data traffic is exchanged in each Hub node. Note agents and Hub nodes connect to bootstrap server only for authentication or reporting information.

B. System Operation

The bootstrap server is initialized when secure P2P networks are starting. One peer can request it to acquire an account to join the system and becomes a SPA by default. Bootstrap server will give a list of Hub nodes according SPA's location where its country and city are. To approach this purpose, the database of mapping IP address to geographic location [20] is adopted. If none of Hub nodes exists in the system, the SPA will become the first Hub node. Otherwise, SPA will measure Round Trip Times (RTTs) between it and Hub nodes in the given list. Then SPA will select the Hub node with minimum measured RTT to connect. When the selected Hub node is too far from the SPA (ex. RTT $\leq$ 50ms) or the members of it are too many (ex. over than 50), the SPA will become a new Hub node under performance and load balance consideration. Once a new Hub node is created, it will request bootstrap server to get five numbers of Hub nodes as their neighbors including three local neighbors the same country as it and two random Hub nodes in foreign countries if they exist. The new Hub node will connect their neighbors to join the system. When Hub node leaves system, members of it will automatically rejoin the system after random time. SPA can request bootstrap server to be a RAA or PAA and get related keys for secure communication.

C. Secure Communication

To construct secure P2P networks, message and data exchange among members of P2P system are encrypted. The communication between bootstrap server and the other agents is protected by AES [21]. The RSA [22] is implemented to protect the communication between two Hub nodes, between Hub node and its member, and between two agents. All kinds of agents and Hub nodes have to register in bootstrap server and then get the corresponding keys from it as follows.

- The private keys of AES for communication between bootstrap server and the others (agents and Hub nodes)

 ISBN: 978-1-61208-300-1

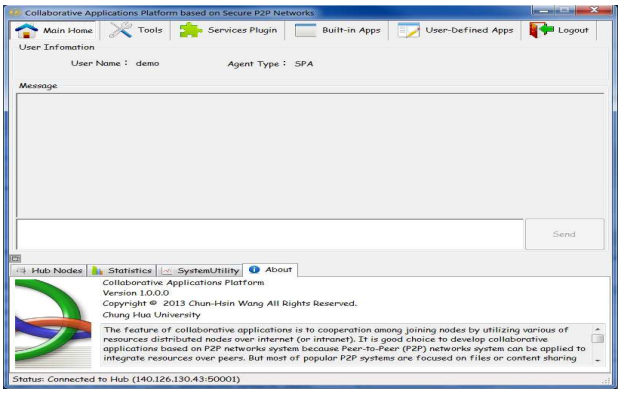

Figure 4. GUI of agent

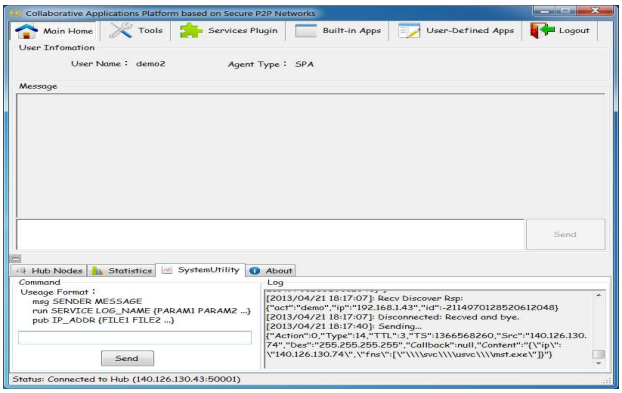

Figure 6. System utilities of agent

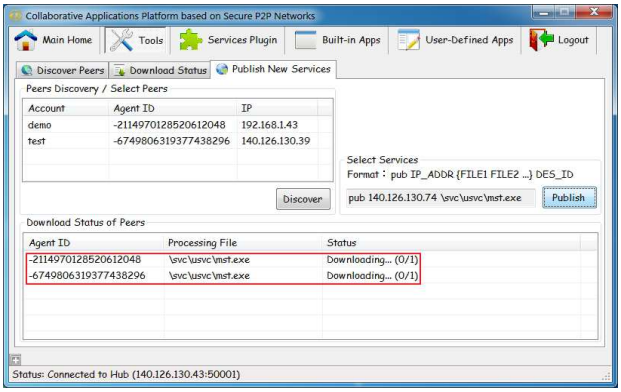

Figure 5. Publish new service to selected peers

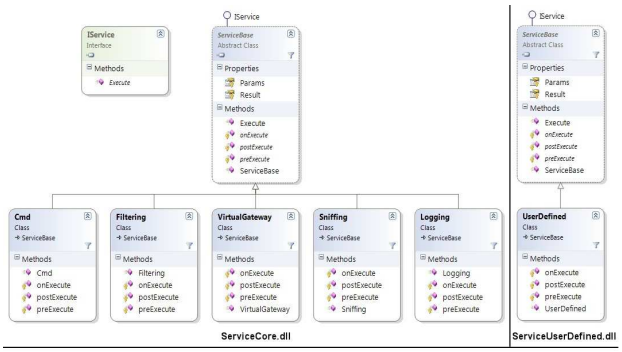

Figure 7. Class diagrams of core and user defined services

are generated by bootstrap server and given to agents when they log in the system.

- When agents or Hub nodes register to bootstrap server, their associated public and private keys will be generated and saved in database by bootstrap server for preparation of RSA communication.
- When agent want to connect a Hub node, it will query bootstrap sever to get public key of the Hub node first and then use it to encrypt a connection request. The Hub node can decrypt the connection request by its private key and send a request to bootstrap server for verifying the agent. If the agent exists, bootstrap server will response its public key. Then, the Hub node will accept the connection request and response message encrypted by public key of the agent. Otherwise, the connection request will be rejected.
- The connection between two Hub nodes are protected by RSA communication. It occurs when a new Hub node wants to connect its neighboring Hub nodes. In a similar way as mentioned above, the public keys of Hub nodes and verification of Hub nodes are also provided by bootstrap server.
- PAA or RAA agent can request other agents over the system to provide their services. To approach it, the first thing is to discover the associated agents by overlay networks. The public key of requesting agent can be sent with the request of discovering agents. Then, the found agents will response with their public keys back. The communication between two agents can be protected by RSA encryption.

D. Implementation of Agents and Hub nodes

Although we have three different types of agents and Hub nodes, a universal Graphic User Interface (GUI) in Fig. 4 is designed for simplification of operations. User name, type of agent, and the connected Hub node are displayed when user logs in the system. A simple chatting function to send messages to all of peers is implemented. Unavailable functions can be seen but can not be used for agents without the priority. For example, the SPA agents can see the function of ”publish new services” but they are not granted to use it.

The menu bar on the top consists of ”Tools”, ”Services Plugin”, ”Built-in APPS”, ”User-Defined APPS”, and ”Logout”. Three frequently used tools are designed. The first tool is discovery of peers in a Time-to-Live range. The second tool is used to observe the current download status of services (or files) from other peers. The third tool is used to publish new services which only PAA agents have right to do it. Fig. 5 shows the tool of publishing new services defined by users to the selected peers. The function of ”Services Plugin” is used to check what core services and user defined services are plugged in. ”Built-in APPS” defines bulit-in collaborative applications we have designed. We will discuss it in next two section. ”User-Defined APPs” is reserved for embedding user defined collaborative applications in the future.

The sign of plus (”+”) down the middle of GUI can be clicked to show the optional utilities as in Fig. 4. We

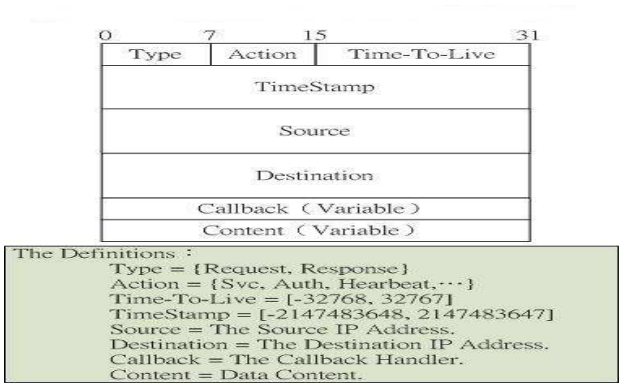

Figure 8. Application layer protocol between two peers

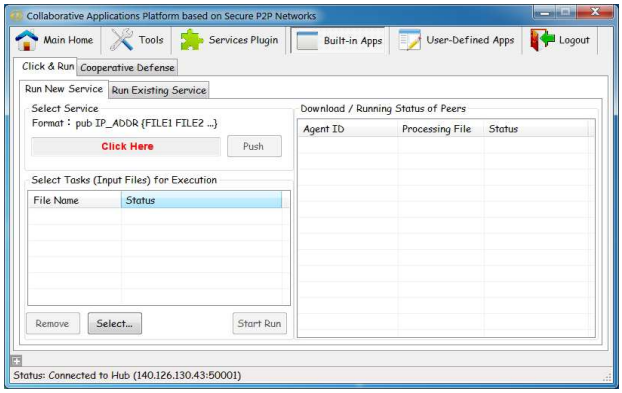

Figure 9. Click and open file browser to select program

can see the information of the proposed collaborative applications platform such as software version, introduction, etc. In addition, information of the connected Hub node can be seen such as its IP address, port number, etc. If the agent is also a Hub node, optional functions can be enabled to observe the statistical information how number of data bytes are received and transmitted. ”System Utility” (Fig. 6) can enable some manual commands to request other peers and show log information of history. Due to the page limitation, only few figures are shown in this paper.

III. Collaborative Applications Platform

Collaborative applications platform is constructed by services distributed over agents in secure P2P networks. It consists of two classes (Fig. 1), core services and user defined services which can be published to be plugged in peers. The core services are composed of basic and useful services we have implemented and will be keeping extension. For example, simple *cmd* service can ask peers to execute services (programs) in background by command shell. *Logging* service can save result of running services and send it back to requesting agents (RAA or PAA).

Some useful services for defending network security problems are implemented such as packet sniffing, filtering, and virtual gateway services. Peers can be asked to monitor and drop harmful packets according to signatures of packets by packet sniffing and filtering services respectively. Since peers can only verify packets they have received, the area of defending network security is limited. It motivates us to design

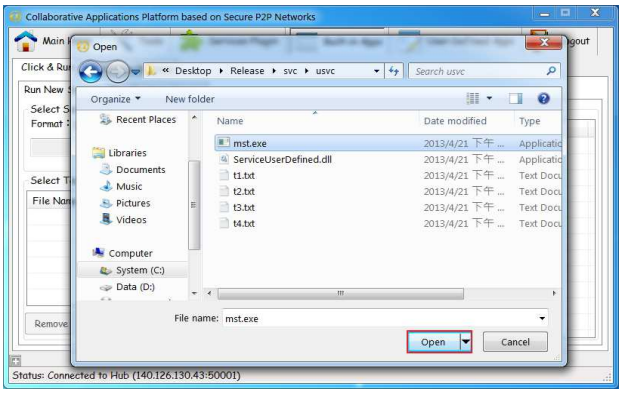

Figure 10. Select a program to be published

the virtual gateway service which can redirect packets passing through neighboring nodes to the peer running the service. A peer running virtual gateway service will generate ARP reply packets [23] to cheat their neighboring nodes that the MAC address of the real gateway is the MAC address of it. Without the assistance of network switches, all packets from neighboring nodes will be forwarded to the virtual gateway. These core services can be applied to develop collaborative applications to defend network security problems.

Besides core services we have designed, user defined services are allowed and easily plugged in peers. The services are packaged into two dynamic linking libraries. The class diagrams of core services and user defined services are shown in Fig. 7. Core services are provided by ServiceCore.dll and user defined services can be designed and implemented into ServiceUserdefined.dll. The library ServiceSDK.dll including two main components (Iservice and ServiceBase) is provided to let users define their services in ServiceUserdefined.dll and develop collaborative applications. PAA with highest priority can publish new services embedded in ServiceUserdefined.dll to the associated peers. In this way, the collaborative applications platform can be easily scalable. It can be expected that more creative collaborative applications based on our proposed platform can be developed.

After services in platform have been designed, application layer protocol is required to communicate between two peers. The format of protocol and related definition are shown in Fig. 8. Two types, request and response, are defined basically. The field of Time-to-Live is used to limit broadcast area of messages in overlay networks, which is defined by how number of Hub nodes is passing through. To avoid loop of message transmission, the time stamp is adopted to recognize whether the received messages are repeated or not. The filed of ”Action” is trying to define possible requests or responses of peers such as running services (*SVC*), authentication (*Auth*), Hello packets between two Hub nodes (*Heartbeat*), etc. The associated parameters and data with ”Action” can be set in the field of ”Content”. In addition, the filed of ”Callback” is used to assign a predefined program to handle the response of a corresponding request. New requests or responses of peers can be easily and flexibly added in our application layer protocol.

IV. Examples of Collaborative Applications

Based on the proposed collaborative applications platform, collaborative applications can be designed by use of resources

 ISBN: 978-1-61208-300-1

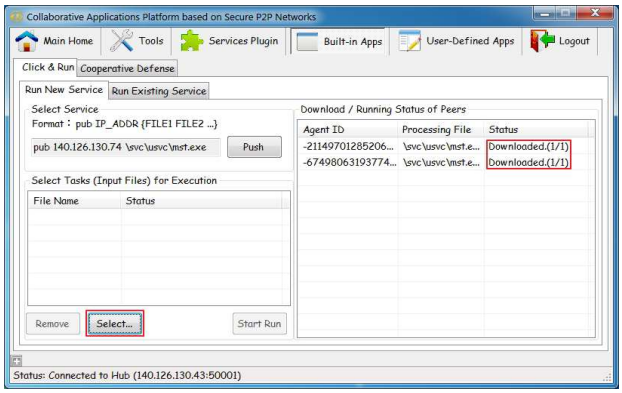

Figure 11. Protocol format is automatically generated.

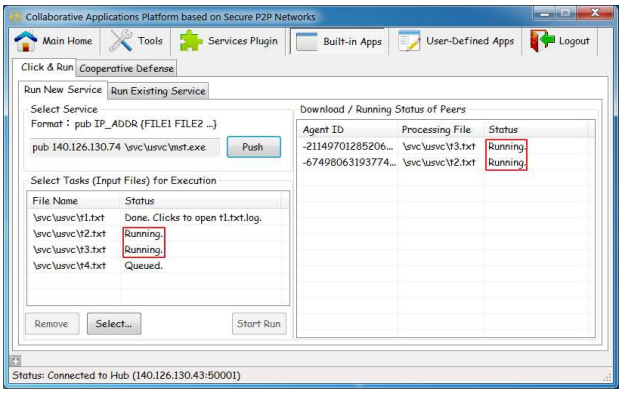

Figure 13. Execution status of each task

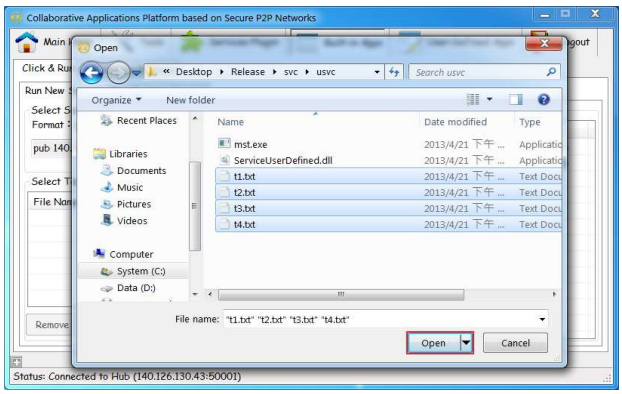

Figure 12. Select input files by a file browser

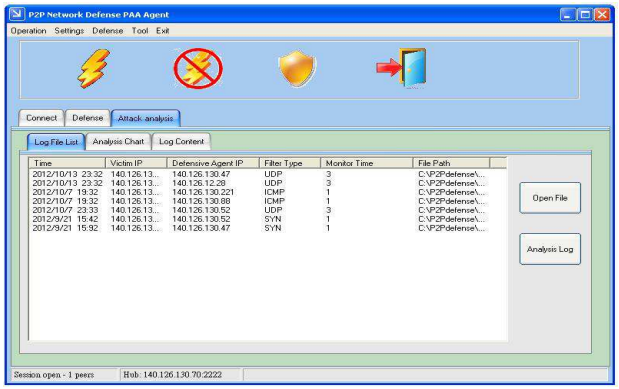

Figure 14. Attack logs from peers

distributed over peers in secure P2P networks. Applications on P2P networks are not limited to file sharing anymore. For demonstration of our platform, two built-in collaborative applications will be introduced and some possible useful collaborative applications will be further studied in the future.

The first buit-in collaborative application is a simple distributed computing application named click&Run. It can request all of peers to execute a new service or existing service. A new service can be a simple execution program. Users can focus on developing their execution programs and request other peers to execute them easily without knowledge of our platform. In our example, we want to request peers to compute minimum spanning trees of random graphs. The execution program, "mst.exe", can be viewed as a new service because the program is not available at other peers. We can simply use this application to select it from a file browser and then send to peers. The operations are shown in Fig. 9 and Fig. 10. The protocol format will be automatically generated instead of typing manually. After that, the "push" button is used to send the program to all of peers. We can observe which peers have already downloaded the program as shown in Fig. 11. In our experiment, there are two peers in the system. Next thing we have to do is to select input files of the published program by a file browser (Fig. 12). When the "start Run" button is clicked, each input file will be automatically assigned to one of peers for execution. Once peer has finished computing, executing result will be sent back and peer will be given the next input file till no more input files exist. The execution status of each task (input file) can be monitored and executing result can be opened by double clicks as shown in Fig. 13. Running existing services of peers can work in a similar way. This simple built-in collaborative application shows computing resource sharing.

The second buit-in collaborative application is trying to integrate various resources over Internet (or intranet) to defend network security. We have designed defensible services in our core services including the virtual gateway service, packet sniffing, filtering, and logging services. Without the assistance of network switches, peers can be the virtual gateway to investigate the packets from other nodes and then filter malicious packets. The investigated packets will be forwarded to the real gateway if they are valid. This application is migrated from our pervious work [24]. The integration of this collaborative application with our platform is still under working. We only show the idea here. Fig. 14 shows attack logs from virtual gateways which they sniff and send log files back to PAA by core services. The content of log files contains time stamp, IP addresses of victim and collaborative agents, traffic type, duration time of monitoring, and file names of logs. Another tool can display statistical volume of attack traffic according to source IP address as shown in Fig. 15. We can observe what IP addresses are suspects of attack origins. In the future, we will finish integrating this collaborative application into our platform.

V. WEB-BASED MANAGEMENT SYSTEM

For convenient management of our platform, a web server is set up. From the web server, users are allowed to register accounts and download software of agent and development libraries. A general user can get information of logging time,

 ISBN: 978-1-61208-300-1

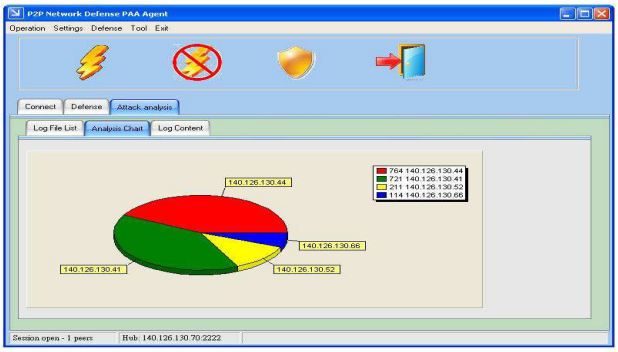

Figure 15. Analysis of attack logs

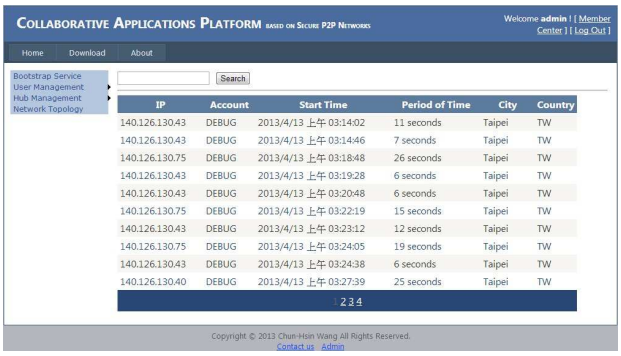

Figure 16. Web-based administration

volume of data traffic exchanged, and personal information. Administrator can startup bootstrap service, manage accounts of users, Hub nodes, and observe where users are from (Fig. 16). Statical volume of data traffic in each Hub node will be considered to design how the proper number of members is and how to select agents to be Hub nodes in the future.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, secure P2P networks are set up by authentication of joining peers and encrypted data communication. Although limitation of users is added, it is necessary for collaborative applications sensible of network security and resource sharing. Based on secure P2P networks, a scalable and flexible collaborative applications platform composed of core services and user defined services is built. Two collaborative applications based on the proposed platform are designed to demonstrate the use of services over peers. In the future, more core services will be added. The services may be combined with cloud services and location where peers are. It can be expected that more creative collaborative applications will be present.

ACKNOWLEDGMENT

This study was supported by the National Science Council of Taiwan, under the Grant No. NSC 101-2221-E-216-038.

REFERENCES

[1] M. Cai, K. Hwang, Y. K. Kwok, S. Song, and Y. Chen, "Collaborative Internet Worm Containment,", *IEEE Security and Privacy*, May/June, 2005, pp. 25-33.

[2] M. E. Locasto, J. J. Parekh, A. D. Keromytis, and S. J. Stolfo, "Toward Collaborative Security and P2P Intrusion Detection," IEEE Workshop on Information Assurance and Security, United States Military Academy, West Point, NY, 2005, pp. 333-339.

[3] Y. Chen, K, Hwang, and W. S. Ku, "Collaborative Detection of DDoS Attacks over Multiple Network Domains," IEEE Transactions on Parallel and Distributed Systems, Vol. 18, No. 12, December 2007, pp. 1649-1661.

[4] S. Radwane, N. A. Farid, and S. Ahmed, "A collaborative peer-to-peer architecture to defend against DDoS attacks," the 33rd IEEE Conference on Local Computer Networks, Oct. 2008, pp. 427-434.

[5] P. Lee, A. P. Jayasumana, H. D. Bandara, S. Lim, and V. Chandrasekar, "A peer-to-peer collaboration framework for multi-sensor data fusion," Journal of Network and Computer Applications, May 2012, pp. 1052-1066.

[6] H. M. Xu, Y. J. Shi, Y. L. Liu, F. B. Gao, and T. Wan, "Integration of Cloud Computing and P2P: A Future Storage Infrastructure," International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (ICQR2MSE), June 2012, pp. 1489-1492.

[7] J. Xu, J. Yan, L. He, P. Su, and D. Feng, "CloudSEC: A Cloud Architecture for Composing Collaborative Security Services," in 2nd IEEE International Conference on Cloud Computing Technology and Science, Dec. 2010, pp. 703-711.

[8] A. H. Payberah, H. Kavalionak, V. Kumaresan, A. Montresor, and S. Haridi, "CLive: Cloud-Assisted P2P Live Streaming," IEEE 12th International Conference on peer-to-peer Computing, Step. 2012, pp. 79-90.

[9] K. Graffi et al., "Towards a P2P Cloud: Reliable Resource Reservations in Unreliable P2P Systems," in 16th International Conference on Parallel and Distributed Systems, Dec. 2010, pp. 27-34.

[10] J. Dharanipragada and H. Haridas, "Stabilizing peer-to-peer systems using public cloud: A case study of peer-to-peer search," in 11th International Symposium on Parallel and Distributed Computing, June 2012, pp. 135-142.

[11] D. Harrison, BitTorrent homepage, http://www.bittorrent.org/, retrieved: July., 2013.

[12] PPLive Inc., PPLive homepage, http://www.pplive.com/, retrieved: July., 2013.

[13] PPStream Inc., PPStream homepage, http://www.ppstream.com/, retrieved: July., 2013.

[14] SETI, http://setiathome.berkeley.edu/index.php, retrieved: July, 2013.

[15] J. R. Douceur, The sybil attack, in First International workshop on peerto-peer systems, 2002, pp. 251-260.

[16] Vuze, http://wiki.vuze.com/, retrieved: July, 2013.

[17] J. Shi, J. Liang, and J. You, "Measurements and Understanding of the KaZaA P2P Network,", Current Trends in High Performance Computing and Its Applications, 2005, pp. 425-429.

[18] Lidgren networking library, http://code.google.com/p/lidgren-network-gen3/, retrieved: July, 2013.

[19] SQLite, SQL database engine, http://www.sqlite.org/, retrieved: July, 2013.

[20] MaxMind, Inc., http://dev.maxmind.com/geoip/geolite, retrieved: July, 2013.

[21] J. Daemen and V. Rijmen, "The Design of Rijndael: AES V The Advanced Encryption Standard," Springer, 2002.

[22] A. J. Menezes, P. C. V. Oorschot, and S. A. Vanstone, "Handbook of Applied Cryptography," CRC Press, the 5th printing in August 2001.

[23] Network Working Group ARP, RFC 826, 1982.

[24] C. H. Wang and C. W. Huang, "A Collaborative Network Security Platform in P2P Networks," International Conference on New Trends in Information and Service Science, June/July, 2009, pp. 1251-1256.

 ISBN: 978-1-61208-300-1