1-1-2008

P2P distributed intrusion detections by using mobile agents

Dayong Ye
University of Wollongong, dayong@uow.edu.au

Quan Bai
University of Wollongong, quan@uow.edu.au

Minjie Zhang
University of Wollongong, minjie@uow.edu.au

Zhen Ye
Hefei University of Technology, yezhen1952@yahoo.com.cn

Recommended Citation

Ye, Dayong; Bai, Quan; Zhang, Minjie; and Ye, Zhen: P2P distributed intrusion detections by using mobile agents 2008, 259-265.
https://ro.uow.edu.au/infopapers/1403

P2P distributed intrusion detections by using mobile agents

Abstract

Traditional hierarchical intrusion detection systems have a central manager which attracts hackers to attack and might overload when there are too many client requests. To overcome these drawbacks, some researchers suggested to apply Peer-to-Peer approaches in intrusion detection. Most current Peer-to-Peer intrusion detection systems only allow hosts to collect related information from "neighbours" (one hop hosts). The limitation of information sources may lead a system to make inaccurate decisions. In this paper, we propose a Mobile Agent Based Peer-to-Peer Distributed Intrusion Detection Framework. Agents are included in this framework to achieve intrusion detections. In addition, a mobile agent migration strategy is applied in the framework to allow agents not only to collect information from direct-linked "neighbours" but also other hosts in the network. Benefitted from agent and Peer-to-Peer techniques, our framework can decrease the overhead of each host in the network, reduce the security risk, and achieve more accurate detections.

Keywords

era2015, P2P, Distributed, Intrusion, Detections, Suing, Mobile, Agents

Disciplines

Physical Sciences and Mathematics

Publication Details

Bai, Q., Zhang, M., Ye, D. & Ye, Z. 2008, "P2P distributed intrusion detections by using mobile agents", in R. Lee (eds), IEEE/ACIS International Conference on Computer and Information Science, IEEE, Los Alamitos, California, USA, pp. 259-265.

P2P Distributed Intrusion Detections by Using Mobile Agents

Dayong Ye
School of Computer Science
and Software Engineering
University of Wollongong
NSW 2522 Australia
dy721@uow.edu.au

Quan Bai
School of Computer Science
and Software Engineering
University of Wollongong
NSW 2522 Australia
quan@uow.edu.au

Minjie Zhang
School of Computer Science
and Software Engineering
University of Wollongong
NSW 2522 Australia
minjie@uow.edu.au

Zhen Ye
School of Computer and Information
Hefei University of Technology
Hefei Anhui China 230009
yezhen1952@yahoo.com.cn

Abstract

Traditional hierarchical intrusion detection systems have a central manager which attracts hackers to attack and might overload when there are too many client requests. To overcome these drawbacks, some researchers suggested to apply Peer-to-Peer approaches in intrusion detection. Most current Peer-to-Peer intrusion detection systems only allow hosts to collect related information from "neighbours" (one hop hosts). The limitation of information sources may lead a system to make inaccurate decisions. In this paper, we propose a Mobile Agent Based Peer-to-Peer Distributed Intrusion Detection Framework. Agents are included in this framework to achieve intrusion detections. In addition, a mobile agent migration strategy is applied in the framework to allow agents not only to collect information from direct-linked "neighbours" but also other hosts in the network. Benefitted from agent and Peer-to-Peer techniques, our framework can decrease the overhead of each host in the network, reduce the security risk, and achieve more accurate detections.

1. Introduction

Security issues, such as network intrusion and virus infection, are becoming more and more serious with the growth of computer and network applications. Intrusion is "a set of actions which attempt to compromise the confidentiality, integrity or availability of a resource" [2]. In order to prevent information from malicious attackers, Intrusion Detection System (IDS) is used to detect various intrusions in network environment. Traditional IDSs [1] [3] [4] [8] [10] normally have centralised and hierarchical architectures. In a traditional IDS, audit records data and network monitoring packets are collected from each host and analysed by a top central analyser. Then, the central analyser makes an estimation on whether there is an intrusion, and decides how to take actions against intrusions.

Traditional IDSs have two obvious drawbacks. Firstly, the centralised and hierarchical architecture of traditional IDSs may lead to a single point failure. Within a hierarchical architecture, the failure of the central analyser (e.g. the central analyser is cracked by an attacker) would cause the whole system to be destructed. Secondly, the central analyser of a traditional IDS is easy to become the bottleneck of the whole system when there are many simultaneous client requests.

To overcome the above two limitations of traditional IDSs, some researchers suggest to use Peer-to-Peer (P2P) IDSs instead of hierarchical IDSs. In a P2P IDS, each host can send detection request to other hosts of the system to check whether there are suspicious activities, and estimate whether the network is intruded. Without a centralised controller, single point failure can be avoided in P2P IDSs effectively. However, most current P2P IDSs only allow hosts of a system to obtain detection information from limited sources (e.g. direct-linked neighbours). This limitation may lead a system to make inaccurate decisions. For example, when a multi-hosts attack occurs, current P2P IDSs may not recognise it due to the limitation of information. Multi-hosts attack means an attacker simultaneously tries to attack multiple hosts of a network, such as *doorknob* and *network browsing* [6]. The main feature of multi-hosts attack is that the activity level of the attacker on each host may not be sufficiently high enough to raise an alarm to the whole network. However, if a distributed intrusion detection system can collect and analyse correlated information from multiple hosts, it might recognise this attack. The motivation of this research is to develop a distributed intrusion detection framework by taking advantages of mobile agent technique and P2P architecture so as to overcome some limitations of current P2P IDSs.

In this paper, a P2P Distributed Intrusion Detection Framework by Using Mobile Agents (MADIDF) is proposed. Mobile agents are intelligent agents that can migrate among hosts. They can execute tasks autonomously in dynamic environments. In MADIDF, mobile agents are used to travel among hosts and detect intrusion of a system. A host can send mobile agents to other hosts in the network to gather relevant information from multiple hosts (the exact number of hosts is dependent on the specific multi-hosts attack) and recognise the aforementioned multi-hosts attacks. In addition, P2P architecture can avoid single point failure and decrease overhead of each host.

The remaining of this paper is organised as follows. Section 2 introduces some related works of this research. In Section 3, the architecture of MADIDF is proposed and each component in the framework is introduced in detail. In Section 4, a novel mobile agent migration strategy is proposed to allow agents traveling on the P2P network effectively. Experiment and analysis are presented in Section 5. Finally, the paper is concluded and the further research is outlined in Section 6.

2. Related Work

In [1], a distributed intrusion detection framework based on autonomous and mobile agents was presented. It makes use of administrator agent to create analyser agents and send them toward the stations to be analysed. There is a crisis agent to create a new administrator agent if the administrator falls in breakdown. In a large network, the administrator needs to create many analyser agents and send them out. This is a heavy burden for the administrator, and the administrator will become the bottleneck of the whole system. Hence, the scalability might be an issue.

DIDMA [3] is a distributed intrusion detection system using mobile agents. It is aimed at building a distributed IDS which places static agents at every host and the network along with a centralised Mobile Agent Dispatcher and IDS console. Although it has better scalability and is platform independent, it still faces a security problem. That is if a hacker cracks mobile agent dispatcher or IDS console, the whole system will fail.

The Multi-agent based Intrusion Detection Architecture [10] is a hierarchical architecture too. It consists of four types of agents, including basic agent, coordination agent, global coordination agent, and interface agent. If basic agent encounters a complex task it is unable to handle, a coordination agent is created dynamically. The coordination agent communicates with other basic agents and directs them to perform certain functions cooperatively. When the coordination agent encounters a complex task it is unable to handle, it will give a report to the global coordination. In this system, the global coordination might be a weak point for intrusion.

In [5], a P2P intrusion detection system based on mobile agents is introduced. This IDS gives up traditional hierarchical architecture. Hosts in a LAN monitor each other. They periodically send mobile agents to their neighbours to detect intrusions. When anomalous behaviours are detected, the observer neighbour initiates a voting process. It sends a mobile agent with a voting sheet to other neighbours of the compromised host for cooperative decision. This system can totally avoid single point failure problem. However, each host in the network has to store critical information of its neighbours which is a burden for each host, and periodic detection may result in network overload and have negative impact on the system overall performance.

The systems proposed in [7] and [9] are the same. The system is similar with that described in [5]. However, the host in the network only dispatches a mobile agent to its neighbours when a suspicious incident is observed at that host instead of periodically sending mobile agent to its neighbours. Although this system overcomes some drawbacks of that proposed in [5], it still has a few limitations. Because the host in the network only asks its neighbours for collaborative decision, it might not detect some specific attacks, which simultaneously attack multiple hosts in a network, like *doorknob* and *network browsing*.

As introduced in the previous paragraphs, most current IDSs have some limitations and drawbacks especially on the aspects of single point failure and detection accuracy. MADIDF, which is introduced in this paper, adopt a P2P detection architecture, which can make it avoid single point failure. Furthermore, Intrusion detections in MADIDF are not only relied on direct-linked neighbours of a particular host, but also other hosts in the network. In this way, the original host can obtain more information to achieve a more accurate decision.

3. Framework architecture

MADIDF is composed of six types of agents which are Monitor Agent, Analysis Agent, Executive Agent, Manager Agent, Retrieval Agent, and Result Agent. The former four agents are static agents that inquiline on hosts, while the latter two are mobile agents that can travel among hosts. Consideration of the security and flexibility of the system, each host in the network has to be equipped with the four static agents when they join in a network. This framework is independent of specific network architecture. The framework of MADIDF is demonstrated in Figure 1. The following sessions describe each component of the framework in detail.

3.1 Monitor Agent

Monitor Agent is like a host monitor which fixes at a host. It is a main component to find intrusions in MADIDF. The responsibility of Monitor Agent is collecting and preprocessing information of both system audit records and network traffic for further analysis, such as system file operation and network connection.

3.2 Analysis Agent

Analysis Agent integrates and analyses the information received from Monitor Agent. In MADIDF, each host in the network has a local knowledge base which stores some critical information, such as attack signatures, intrusion patterns, system file size, and so on. If it can confirm an intrusion or attack, it will send a notification to Executive Agent to quarantine damaged file or cut off network connection. If Analysis Agent suspects that a multi-hosts attack occurs, it will request Manager Agent for help and store the suspicious activity.

3.3 Executive Agent

Executive Agent is responsible for executing tasks depending on the notification of Analysis Agent. These tasks include restoring corrupted files, preventing network connection, and so on.

3.4 Manager Agent

As introduced in Subsection 3.1, the Monitor Agent of a host is the agent that collects information from the host. However, to detect multi-hosts attacks, it is not sufficient to only collect information from a single host. Hence, another three kinds of agents (i.e. Manager Agent, Retrieval Agent and Result Agent) are included in MADIDF to collect related information from multiple hosts of a network.

The Manager Agent is the agent that manages retrieval processes. It takes charge of Retrieval Agent and Result Agent, including dispatching, retracting and communicating with these two agents. A Manager Agent also maintains a Neighbour List of the host which the Manager Agent resides on (see Definition 1 and Definition 2).

Definition 1: Neighbour List NL_i is a list which contains IP addresses of the direct-linked neighbours of host H_i.

Definition 2: Neighbour Size NS_i is an integer that represents the number of neighbours (direct-linked hosts) of host H_i.

Obviously, the Manager Agent of a host can calculate NS_i of the host by using a function $Size()$ as the following Equation 1.

$$NS_i = Size(NL_i) \tag{1}$$

When a host connects/disconnects with another host, the Manager Agent of the host modifies the Neighbour List by adding/removing related information to/from the list. In addition, each Manager Agent has a Retrieval Agent Recorder which is used to store Retrieval Agent Identifiers (see Definition 3 and Definition 4).

Definition 3: *Retrieval Agent Identifier (RAID)*is used to distinguish different Retrieval Agents. RAID is also generated by the Manager Agent. We define it as the format "HostName0001". "HostName" means the name of the host which dispatches the Retrieval Agent, while "0001" means the serial number of the Retrieval Agent, for example, the first group of Retrieval Agents which perform the same task is "0001", the second group is "0002", and so on.

Definition 4: *Retrieval Agent Recorder (RAR)* is used to store Retrieval Agent Identifiers in order to avoid Retrieval Agent traveling the hosts which it or other Retrieval Agents with the same RAID have already visited. Each Manager Agent has a RAR.

If a Manager Agent originates the mobile agent traveling detection, this Manager Agent is called as an *Initiator*. When an Initiator receives a request from an Analysis Agent for deciding a multi-hosts attack, it will dispatch Retrieval Agents to inform other hosts to check whether they have the similar records from the same suspicious remote host. Then, each Manager Agent of the hosts, which have been visited by those Retrieval Agents, will send a Result Agent back to the Initiator. The Initiator will correlate the information and confirm whether this suspicious activity is a multi-hosts attack. If so, the Initiator will broadcast this information to other hosts in the network and notify the local Executive Agent to take actions.

3.5 Retrieval Agent

Definition 5: *Time to Life (TTL)*, generated by an Initiator, is used to demonstrate the number of rest hosts the Retrieval Agent needs to visit.

Retrieval Agent moves to other hosts and lets their Analysis Agents check whether there are the similar records from the same suspicious remote host. There are four main types of information that Retrieval Agent needs to maintain,

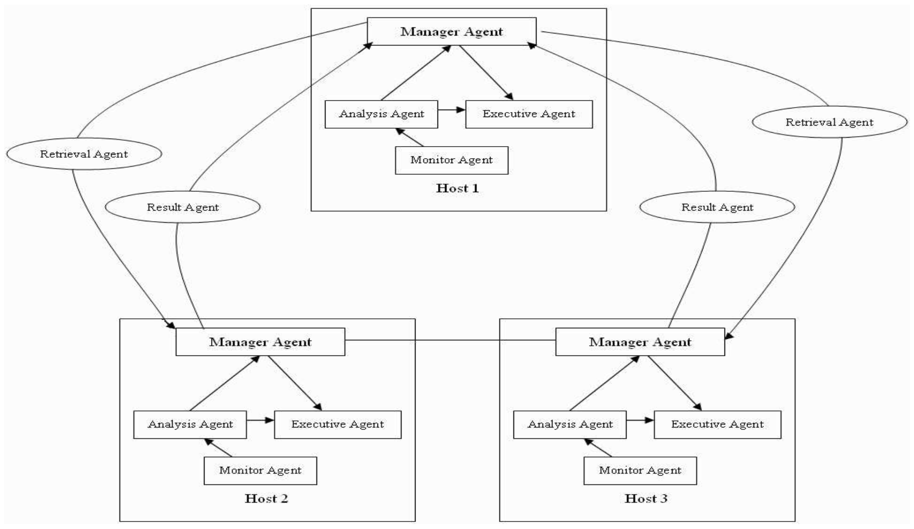

Figure 1. The components in MADIDF

which are source IP address from where the original host dispatches this Retrieval Agent, characters of the incident, Retrieval Agent Identifier (RAID), and Time to Life (TTL). The Retrieval Agent will be discarded when the value of TTL reaches zero or there is no more host to be traveled.

3.6 Result Agent

Result Agent with a result record will be sent back by each Manager Agent, which has been visited by the Retrieval Agent, to the Initiator. Then, the Initiator tallies all the result records to make a final decision.

3.7 Agent working process

The working process of agents in MADIDF is described by a sequence diagram in UML notations shown in Figure 2. In MADIDF, Monitor Agents are fixed at hosts of the network and monitor the local activities of hosts. When a suspicious activity happens and is detected by a Monitor Agent, it collects and preprocesses relevant information, and reports the information to Analysis Agent. The Analysis Agent analyses the information and decides whether there is an intrusion or attack based on the local knowledge base. If so, it informs the Executive Agent to take actions against the intrusion or attack. However, if the Analysis Agent suspects that a multi-hosts attack is occurring, it asks the upper level Manager Agent for help. When the Manager Agent receives a request from Analysis Agent, this Manager Agent becomes the Initiator and then analyses the request and dispatches Retrieval Agents to gather information for determining whether some suspicious activities in different hosts could be combined to form a multi-hosts attack. Then those hosts, visited by Retrieval Agent, will send Result Agents with necessary information back to the Initiator. The Initiator will make a final decision based on the information it received. If there is a definite multi-hosts attack, the Initiator will broadcast this detection result to other hosts in the network, and inform the local Executive Agent to take actions.

4. Mobile agent migration strategy in MADIDF

In order to collect information from multiple hosts, a good migration strategy is essential. In this section, a novel mobile agent migration strategy for the Retrieval Agent in MADIDF is proposed and introduced in detail. This migration strategy can let Retrieval Agent visit not only its neighbour hosts but more hosts in the network to collect relevant information.

The retrieval migration strategy includes two parts, Retrieval Agent generation and Retrieval Agent dispatch. These two parts are described in detail in subsections IV-A and IV-B, respectively.

4.1 Retrieval Agent generation

When the Analysis Agent suspects there might be a multi-hosts attack, it sends a request to the Manager Agent (Initiator) for help. Then, the Initiator analyses this request, decides how many hosts (say N) the Retrieval Agent should visit which depends on the specific attack type, and creates a RAID for Retrieval Agents. Therefore, the Initiator compares the number of neighbours it has with the number of

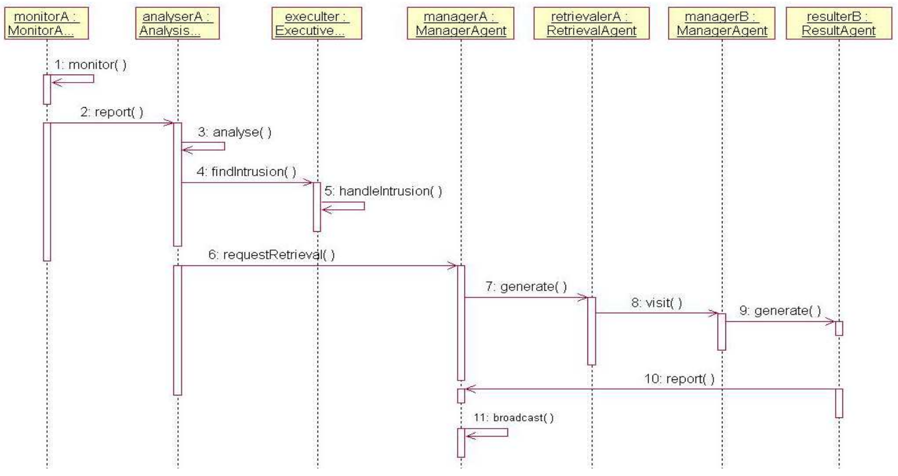

Figure 2. Retrieval Process

hosts the Retrieval Agent needs to visit. The comparison results will fall into the following two cases.

Case One: If the number of neighbours is more than the number of hosts the Retrieval Agent needs to travel, namely $NS_I \geq N$ (NS_I means the number of neighbours the Initiator has), the Initiator generates N Retrieval Agents with the same RAID and $TTL = 1$ for all of them.

Case Two: If the number of neighbours is less than the number of hosts the Retrieval Agent needs to travel, namely $NS_I < N$, the Initiator creates NS_I Retrieval Agents with the same RAID and $TTL = N - NS_I$ for all of them. The pseudocode of Retrieval Agent generation is shown as below.

```
/*NH[i][j] is the IP address of the jth neighbour of host $H_i$; RAR[i][j] is the jth record on the host $H_i$.*/
    Initialise N, NH[i][j], Size(NL[i]), and RAID;

//Case One
    if (Size(NL[I])≥N)
        TTL=1;

//Case Two
    if (Size(NL[I])<N)
        TTL=N-Size(NL[I]);
```

4.2 Retrieval Agent dispatch

After the Initiator decides the number of Retrieval Agents, it will dispatch them with a strategy based on two different ceases (refer to Subsection IV-A).

In Case One, the Initiator just randomly selects N hosts from its NS_I neighbours, and sends N Retrieval Agents to the N hosts, respectively.

In Case two, the dispatch procedure comparing to Case One is more complicated. First, the Initiator has to send NS_I Retrieval Agents to its NS_I neighbours. Then, each of these neighbours randomly selects a host from their neighbours, and retransmits the Retrieval Agent to the next host. In order to avoid Retrieval Agent repeatedly traveling the hosts which it or other Retrieval Agents with the same RAID has/have visited, the Manager Agent on each host has a RAR (refer to Definition 4). If there does not exist a same RAID in the RAR, the Manager Agent just communicates with this Retrieval Agent, performs tasks, minuses 1 from the Retrieval Agent's TTL value, and retransmits this Retrieval Agent to next host. Otherwise, the Manager Agent will send the Retrieval Agent back to the previous host which just dispatched the Retrieval Agent, but does not minus the TTL value of this Retrieval Agent. Then, the previous host chooses another neighbour host to send this Retrieval Agent to. When the value of TTL reaches zero or there is no more host to visit, the process will stop. The process for retrieval agent dispatch in Case One and Case Two is described by the following pseudocodes.

```
//Case one
    if (Size(NL[I])≥N)
        //randomly selects N neighbours from Size(NL[I]),
        //and stores their IP addresses in array temp[i]
        for i=0 to N-1
            RAR[temp[i]][++j]=RAID;

//Case two
    if (Size(NL[I])<N)
        for t=0 to Size(NL[I])-1
            for j=0 to TTL-1
    //Initialise Path[i][j]
    //Path[i][j] means the jth traveling host IP
    //address of the ith Retrieval Agent.
        Path[t][j]=0;
for t=0 to Size(NL[I])-1
    RAR[NH[I][t]][++j]=RAID;
    Path[t][0]=NH[I][t];
TTL=TTL-1;
int counter;
for m=0 to TTL-1
    for t=0 to Size(NL[I])-1
        counter=0;
        for k=0 to Size(NL[Path[t][m]])-1
            if (RAID is in RAR[NH[Path[t][m]][k]])
                counter=counter+1;
        if (counter==Size(NL[Path[t][m]]))
        //There is no more host to travel
            continue;
        do
            randomly generate x
            from 0 to Size(NL[Path[t][m]])-1;
        while(RAID is in RAR[NH[Path[t][m]][x]])
        Path[t][m+1]=NH[Path[t][m]][x];
        RAR[NH[Path[t][m]][x]][++j]=RAID;
```

5. Experiment

In this research, experiments that analyse detection precision, network latency, and network load of MADIDF were conducted. In these experiments, we compared MADIDF with the P2P IDS presented in [7] (i.e. MASHD). MASHD is a P2P intrusion detection model. It also uses mobile agents to detect intrusions in a network. In MASHD, when a corruptive event happens on a host, the host will dispatch a mobile agent to its direct-linked neighbours and request assistances from neighbours to make a collaborative decision. The main drawback of MASHD is that it does not have an efficient migration strategy for agent travelling and only allows agents to travel on direct-linked neighbours of a host. This might lead to inaccurate decisions.

5.1 Experiment description

In order to test the general performance of MASHD and MADIDF, the experiment is performed in four different network scales which include ten hosts, twenty hosts, fifty hosts, and one hundred hosts. The topology of network is generated randomly, and the host which suspects a multi-hosts attack occurring and other hosts which are attacked by the multi-hosts attack are also randomly selected from all hosts in the network. Some terms, which will be used in the experiment, are defined as follows:

Detection Precision: the ratio of detected attack incidents to all attack incidents. In this paper, we only focus on multi-hosts attack.

Network Latency: the range from the time that attack happens to the time that all hosts take actions.

Network Load: the overhead of total communications and agent migration during detection process.

In most situations, detection precision is always the most important aspect for IDSs. Network latency and load are also significant for IDS. Low latency means IDS can take actions against an intrusion as early as possible, and low load can save network resources and improve performance of IDS.

Suppose there is a scenario: a specific host suspects a multi-hosts attack. In MASHD, this specific host sends a mobile agent to traverse its neighbours, then the neighbours dispatch Result Agents to the specific host, finally, the specific host dispatches a Response Agent to all hosts in the network; in MADIDF, this specific host dispatches mobile agents to some hosts (not only neighbours), then these hosts send Result Agents back to the specific host, finally, the specific host broadcasts attack information to all hosts in the network.

In order to achieve precise result, the experiment is executed one hundred times in all of the above four different network scales, and every time the network topology and attacked hosts are re-generated randomly.

5.2 Experiment result and analysis

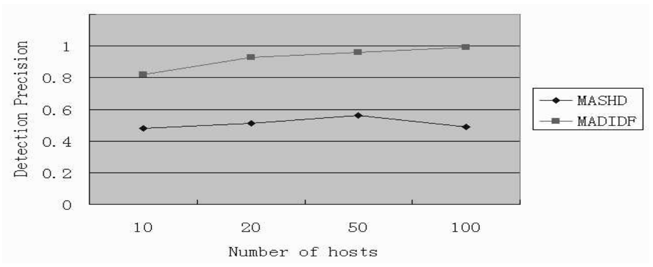

Figure 3. Detection Precision for different number of hosts

Figure 3 shows the detection precisions of MASHD and MADIDF. From Figure 3, it can be seen that the precision of MADIDF is much higher than that of MASHD. This is because that initiators in MASHD just obtain information from its one hop neighbours, while initiators in MADIDF obtain information not only from neighbours but from more hosts. Therefore, hosts in MADIDF can obtain more related information from other hosts and make more accurate decisions.

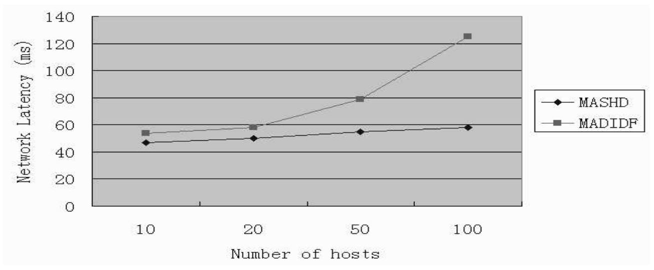

Figure 4. Network Latency for different number of hosts

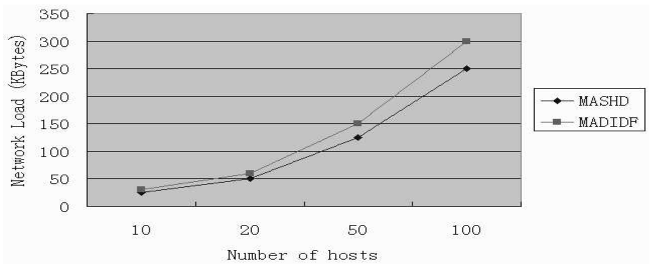

Figure 5. Network Load for different number of hosts

The network latency and network load of MADIDF and MASHD are compared in Figure 4 and Figure 5, respectively. From these two figures, it can be seen that MADIDF has higher network latency and network load than MASHD. This is because initiators in MADIDF detect more hosts in the network which will spend more time and consume more network resources. However, it is also noticed that when the number of hosts is less than fifty, the network latency and network load of MADIDF and MASHD are very close.

According to the above description, it is evident that the detection precision of MADIDF is much better than MASHD. Although network latency and network load of MADIDF are higher than those of MASHD, the gap is very small, especially in medium or small scale network. Therefore, MADIDF uses time and resource to trade higher detection precisions. Since detection precision is the most important factor to evaluate whether a IDS is secure, we can conclude that the overall performance of MADIDF is better than MASHD.

6. Conclusion and future work

In this paper, a framework which uses agent techniques and P2P concept to detect distributed intrusions was proposed. An agent migration strategy that can enable hosts of a network to make more efficient and accurate detections was introduced. Compared to [5], [7], and [9], MADIDF can gather information not only from neighbours of the compromised host but from more other hosts in the network that can lead to more accurate final decision. The future work of this research is to improve the performance of MADIDF, including optimizing the number of Retrieval Agents and avoiding duplicate detection, especially in large scale network and test it in real applications.

References

[1] D. Boughaci, H. Drias, A. Bendib, Y. Bouznit, and B. Benhamou. Distributed intrusion detection framework based on autonomous and mobile agents. In *Proceedings of the International Conference on Dependability of Computer Systems*, pages 248–255, May 2006.

[2] Intrusion-Detection. http://en.wikipedia.org/wiki/intrusion-detection.

[3] P. Kannadiga and M. Zulkernine. Didma a distributed intrusion detection system using mobile agents. In *Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, and First ACIS International Workshop on Self-Assembling Wireless Networks*, pages 238–245, May 2005.

[4] C. Li, Q. Song, and C. Zhang. Ma-ids architecture for distributed intrusion detection using mobile agents. In *Proceedings of the 2nd International Conference on Information Technology for Application (ICITA 2004)*, pages 451–455, 2004.

[5] G. Ramachandran and D. Hart. A p2p intrusion detection system based on mobile agents. In *Proceedings of the 42nd annual Southeast regional conference*, pages 185–190, Huntsville, Alabama, 2004.

[6] S. R. Snapp. Dids (distributed intrusion detection system) - motivation, architecture, and an early prototype. In *Proceedings of the 14th National Computer Security Conference*, pages 167–176, Washington D.C., October 1991.

[7] X. Wang, J. Zheng, K. Xiao, X. Xue, and C. Toh. A mobile agent-based p2p model for autonomous security hole discovery. In *Proceedings of the Fifth International Conference on Computer and Information Technology*, pages 723–727, Sep. 2005.

[8] J. Wu, C. Wang, J. Wang, and S. Chen. Dynamic hierarchical distributed intrusion detection system based on multiagent system. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence and International Agent Technology Workshops*, pages 89–93, Dec. 2006.

[9] K. Xiao, J. Zheng, X. Wang, and X. Xue. A novel peer-to-peer intrusion detection system using mobile agents in manets. In *Proceedings of the Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies*, pages 441–445, Dec. 2005.

[10] R. Zhang, D. Qian, C. Ba, W. Wu, and X. Guo. Multi-agent based intrusion detection architecture. In *Proceedings on the 2001 International Conference on Computer Networks and Mobile Computing*, pages 494–501, Oct. 2001.