Hierarchal Model to Prevent DoS Attack in Mobile Agents

Mayank Aggarwal
Faculty of Engineering & Technology
Gurukul Kangri University
Haridwar (Uttarakhand)

Nipur
Kanya Gurukul Mahavidyalaya
Gurukul Kangri University
Haridwar (Uttarakhand)

Pallavi
Amity University
Noida (U.P.)

ABSTRACT
Mobile Agents are soft-wares migrating from one node to another to fulfill the task of its owner. Mobile agents are not properly utilized because of security concerns. One such concern is 'Denial of Service', in it the malicious host might deny resources required by the agent and kill the agent, thus the result computed so far is lost and this might happen every time the agent visits any malicious host. This paper is an extension of our work [12] in it we extend the concept to hierarchical model by grouping similar hosts together and finally detects the malicious host to prevent Denial of Service Attack. Simulation is done using CPN tools, Colored Petri Nets (CPNs) is a language for the modeling and validation of systems in which concurrency, synchronization and communication play a major role [3].

Keywords
Hierarchal model; mobile agent; denial of service attack; colored petrinets; group in charge.

1. INTRODUCTION
The advantage for using mobile agent technology is that interaction cost for the agent-owner is remarkably reduced since after leaving its owner the agent migrates from one host to the next autonomously. Though a lot of research has been done for security of mobile agents and host security [8,15], it is still a major concern. One such concern is 'Denial of Service' attack by malicious host, in such an attack; the malicious host might prevent an agent from migrating to another host or might even delete the agent. As a consequence, all the results agent has collected so far are lost. This might repeat every time the agent pass through that malicious host while the owner has no knowledge to detect the malicious host[13]. This paper presented a detection mechanism for posteriori identification of such malicious hosts. In general, the term Denial Of Service is used for attacks in which the focus is on exhausting resources with the effect that other entities cannot be served anymore[13]. In this paper we extends the model, we presented in [11] and other model discussed in [12,18], we introduced the concept of the grouping of hosts together to make hierarchal model, finally of which proved that owner can detect the malicious host. The paper is divided in VIII Sections. Section II defined the model having three main components Trust Server, Group In charge and Guard.. Section III gives the steps in the model Section IV deals with Simulation Section V gives the assumptions taken to simulate the model. Section VI discusses the CPN implementation of the model and shows instances of the model at different stages. Many researchers have simulated mobile agents in CPN tools before also [14]. Some related work on agent security is discussed in Section VII. Last Section i.e. Section VIII concludes the paper.

2. MODEL
The proposed model had following three main components:

2.1 Trust server

2.2 Group In charge

2.3 Guard

2.1 Trust Server
Trust server is the topmost layer of the model. It is responsible for mobile agent authentication and commitment. It receives the agent from the creator. We have assumed that trust server uses a trusted hardware to ensure security.

2.2 Group In charge
The group in charge on receiving the agent decrypts the agent with its private key. Check for the authenticity of the agent and also do the mutual authentication. The group is made of more than one mobile agent platform doing the same type of service. The in charge depending on the load on its members transfers the agent to one of its member for execution.

2.3 Guard
Guard receives the host id on successful execution of the agent. If the host is malicious then its id is not given to the guard.

3. STEPS IN THE MODEL
- Agent is received by Trust Server.
- Trust server passes it to different group In charge
- Group In charge register the agents on their corresponding hosts. Hosts executes the agents if successful passes the host id to Guard if not then no id is passed to Guard. Guard on complete execution checks for complete itinerary for all host ids, if anyone is missing it is detected as malicious.

4. SIMULATION
Simulation of the above model is done in CPN tools, to implement the model several assumptions have been made which are discussed further. Finally the result shows that malicious host can be detected in Hierarchal model also.

5. ASSUMPTIONS
- Trust server uses a trusted hardware
- Agent is static i.e. its itinerary is pre-decided
- Group members are of same type

- Group In charge does not send the agent as per the load of its members, it transfer the agent to all its designated hosts in a group. Guard is implemented as a Fusion set. Some members are intentionally made malicious

6. CPN MODEL

The model is simulated using CPN tools. In CPN model we had 7 modules namely model, trust server, G1, G2, G3, G4 and G5. The agent is created with a unique id, source id and ids for host it has to visit. Trust server transfers the agent to different group incharge. Group incharge is implemented for regitering the agent to the respective hosts , on execution at a particular host its id is tranfered to Guard. Guard is implemented as a Fusion set so that all the Guards can have same information at any instance. Whenver the agent visits a predecided malicious host , its acknowledgment is lost. On complete execution a place called result in G5 contains the list of mallicious hosts(if any).

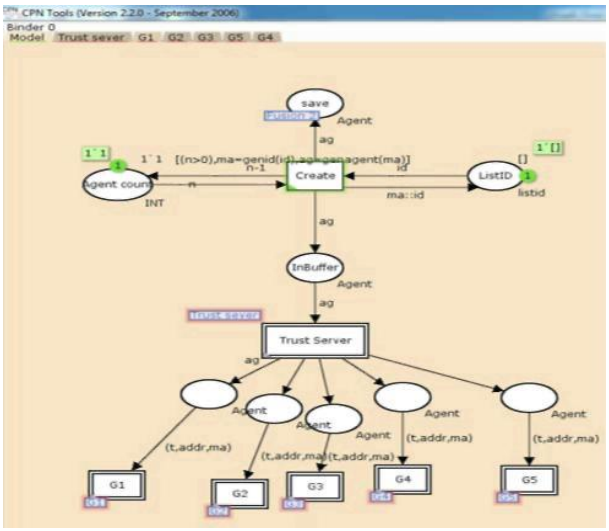

Figure 1. Model implementation in CPN

Figure 1 shows the CPN Model , transition called “Create” is used to create the agent , control on transition is programmed as such that it creates only one agent at a time.The created agent’s copy is saved on place called “Save” and other copy is given to trust server for further action. Trust server transfer the agent to group incharges.

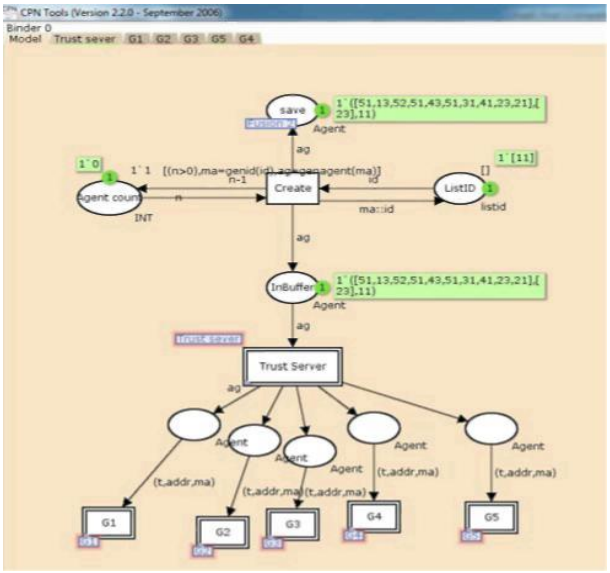

Figure 2. Creation of Agent

Figure 2 shows agent is created having the destinations hosts id, source id and agent id. The created agent is saved on place “Save” and simultaneously transferred to place “Inbuffer” from Inbuffer it will be transferred to Trust Server. The agent shows the path 51,13,52,51,43,51,31,41,23,21. The destination ids are of two digits first digit represents the Group Incharge and second digit represents the host id within the group. We have taken five groups and three hosts within each group. The generated path has repletion of same destination ids, which are ignored.

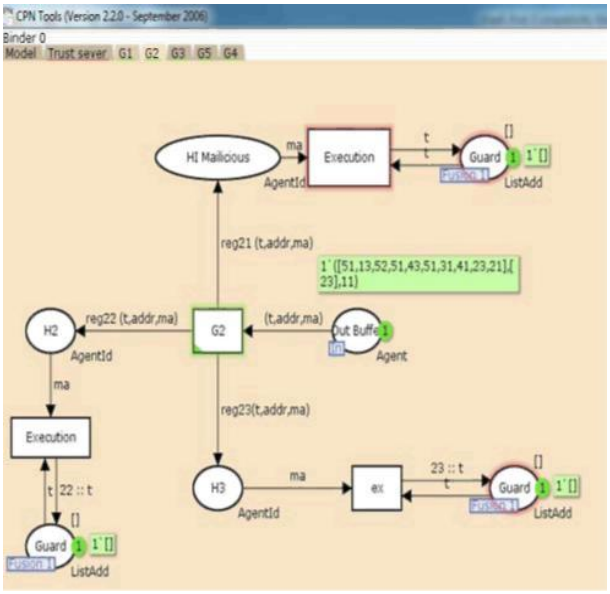

Figure 3. Trust server sends agent to Group Incharge G2

Trust server sends the agent to respective group incharge as per the destination path. The group incharge in turn sends the agent to corresponding host. If the host is not malicous then on execution the host id is transferred to place “Guard” if host is malicous then no acknowledgemnet is given to Guard.

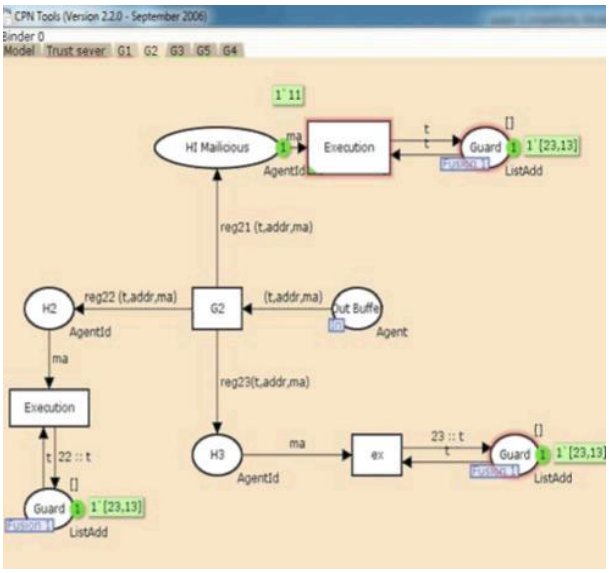

Figure 4. Guard having acknowledgement

Figure 4 shows that agent is executed by two hosts 23 and 13 as Guard has the entry 1`[23,13] which means that guard has received acknowledgment from host 23 and 13 after execution. Host H1 of group G2 is malicious so on execution no acknowledgement will be added to Guard.

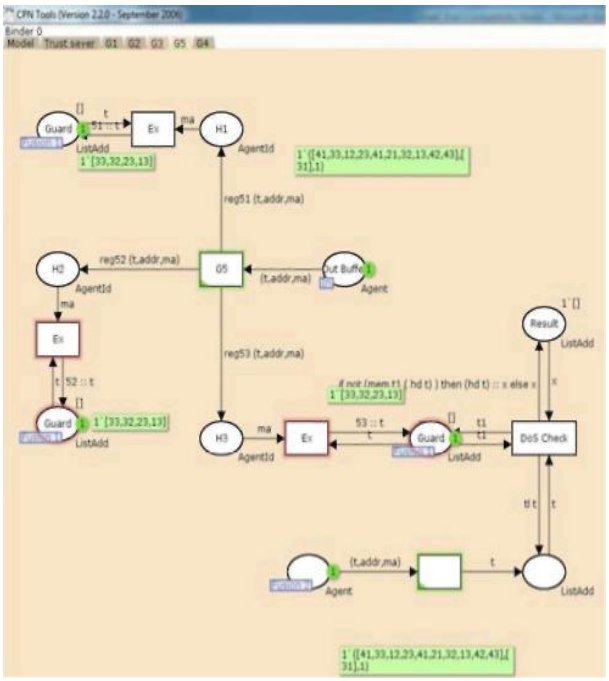

Figure 5. Agent executed by malicious Host

Figure 5 shows that agent is executed malicious host H1 of group G2 and there is no change in Guard, the entries in the Guard is same as Figure 4.Execution by H1 is confirmed by the removal of entry 1`11 in green color from H1 as in figure 4.

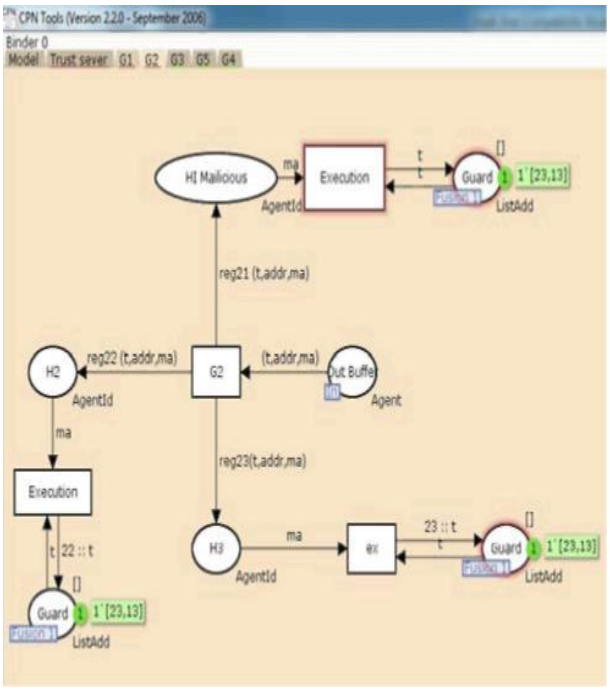

Figure 6. Instance of G5

Figure 6 shows an instance of G5, where agent has travelled in Group G1, G2 and G3, the Guard shows acknowledgment from host ids 33, 32, 23 and 13.Agent is still to be executed by G5. In figure 7 , agent has been executed by all the hosts and Guard shows acknowledgment from host ids 41, 33, 32, 23 and 13 whereas the place Result has host ids 43, 42, 21 and 12. This shows that on final execution we can detect the malicious host by the presence of their ids on place Result.

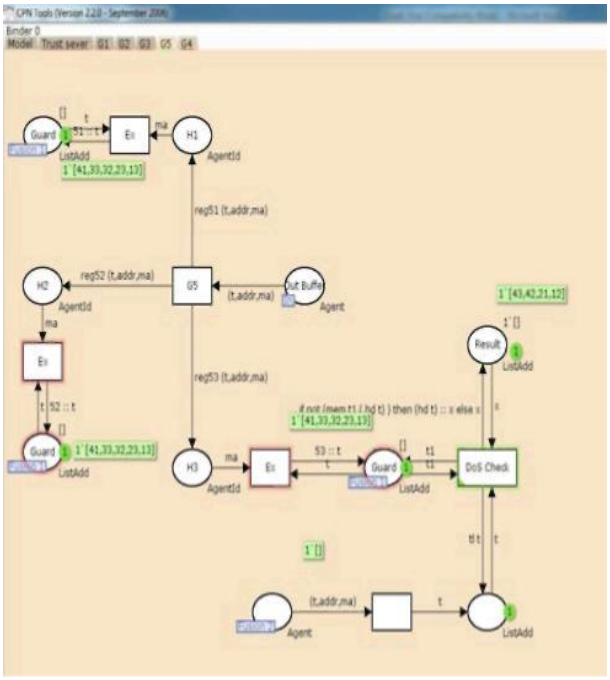

Figure 7. Malicious host at result

7. RELATED WORK

Many of the problems concerning the security of mobile agent systems, both protecting the host from malicious agents and protecting agents from malicious hosts, have been discussed in the literature. Denial of service attack is tackled by [ ],

Counter measures for mobile agent security are well discussed in [19]. Execution tracing [9] is a technique for detecting unauthorized modifications of an agent through recording the agent's behavior during its execution on each host. In [10] Lofti and Samuel introduce the concept of mobile agent protection by clones. In [17] Sander and Tschudin introduce the concept of computing with encrypted functions and thus protecting the integrity and the privacy of the agent's computations. Path history based accessed control is discussed in [2].

Corradi present in [1] methods for protecting the agent's integrity —both making use of a Trusted Third Party and without it. In [6], Kim presented an adaptive migration strategy that can be used to avoid mobile agents from blocking or crashing.

This is achieved by a route reordering algorithm and a backward recovery algorithm.

In [4,5] Westhoffet describe methods for the protection of the agent's route against hosts spying out route information. One technique for ensuring that a mobile agent arrives safely at its destination is through the use of replication and voting [7]. The problem of detecting the black hole—a stationary process destroying visiting agents—in an anonymous ring is addressed in [16] visiting agents—in an anonymous ring is addressed in [16].

8. CONCLUSION AND FUTUTRE WORK

The simulation above shows that that the model can detect for the malicious host whenever 'Denial of service' attack occurs and thus in the next journey that malicious host can be skipped to prevent 'Denial of service' attack.

The model discussed above is well suited for Static mobile agent, but there may be certain case like the acknowledgement is lost due to network, or the receiver does not send the acknowledgement deliberately; such cases need to be considered. Also we have considered only for Static Mobile Agent, it is not considered for Dynamic mobile agent. Other limitation is we have created single agent at a time it has not dealt multi agent system.

9. REFERENCES

[1] Antonio Corradi, Marco Cremonini, Rebecca Montanari, and Cesare Stefanelli. Mobile agents integrity for electronic commerce applications.Information Systems, 24(6), 1999.

[2] Cao, Chun andLu, Jian 'Path-history-based access control for mobile agents', International Journal of Parallel, Emergent and Distributed Systems, vol 21: 3, pp 215 — 225 , 2006.

[3] CPN Tools website:www.daimi.au.dk/CPNtools

[4] Dirk Westhoff, Markus Schneider, Claus Unger, and Firoz Kaderali. Methods for protecting a mobile agent's route. In Information Security, Second International Workshop (ISW'99), number 1729 in LNCS. Springer Verlag, 1999.

[5] Dirk Westhoff, Markus Schneider, Claus Unger, and Firoz Kaderali. Protecting a mobile agent's route against collusions. In Selected Areas in Cryptography, 6th Annual International Workshop (SAC'99), number 1758 in LNCS. Springer Verlag, 2000.

[6] Dong Chun Lee and Jeom Goo Kim. Adaptive migration strategy for mobile agents on internet. In Technologies for E-Services (TES 2001), Second International Workshop, Proceedings, number 2193 in LNCS. Springer Verlag, 2001.

[7] Fred B. Schneider. Towards fault-tolerant and secure agentry. In Distributed Algorithms, 11th International Workshop (WDAG'97), Proceedings, number 1320 in LNCS. Springer Verlag, 1997.

[8] Garrigues, C., et al. Promoting the development of secure mobile agent applications. J. Syst. Software (2009), doi:10.1016/ j.jss.2009.11.001

[9] Giovanni Vigna. Cryptographic traces for mobile agents. In G. Vigna, editor, Mobile Agents and Security, number 1419 in LNCS. Springer Verlag, 1998.

[10] Lotfi Benachenhou, Samuel Pierre, " Protection of a mobile agent with a reference clone," Elsevier , Computer Communications , vol 29, pp. 268-278, 2006.

[11] M.Aggarwal,Nupur,Pallavi, " Protecting Dynamic Mobile Agent against Denial of Service Attacks",AIP,Conference Proceedings,1324 (316),pp 316-318,2010.

[12] M.Aggarwal, Nipur , Pallavi , Simulation of Dynamic Moble Agent Model to pevent denial od service attack, international journal of computer applications. Vol 20 (1), 2011

[13] M. Schenider, B.Cubaleska "A method of protecting mobile agents against denial of service attacks" , Springer-Verlag Berlin Heidelberg, LNAI 2446, pp. 297–311, 2002.

[14] N.Desai,K.Garg,M.Mishra, Modelling Hierrarchical Mobile Agent Security Prorotocol Using CP Nets, Springer-Verlag Berlin Heidelberg, LNCS 4873, pp 637-649,2007.

[15] Price, Sean M. 'Host-Based Security Challenges and Controls: A Survey of Contemporary Research', Information Security Journal: A Global Perspective, vol 17: 4, pp 170 — 178, 2008.

[16] Stefan Dobrev, Paola Flocchini, Guiseppe Prencipe, and Nicola Santoro. Mobile search for a black hole in an anonymous ring. In Distributed Computing (DISC 2001), 15th International Conference, Proceedings, number 2180 in LNCS. Springer Verlag, 2001.

[17] Tomas Sander and Christian F. Tschudin. Protecting mobile agents against malicious hosts. In G. Vigna, editor, Mobile Agents and Security, number 1419 in LNCS. Springer Verlag, 1998.

[18] Venkatesan S, et al. "Advanced mobile agent security models for code integrity and malicious availability check.", J Network Comput Appl, doi:10.1016/j.jnca.2010.03.010 ,2010.

[19] W.A.Jansen, "Countermeasures for mobile agent security," Elsevier, Computer Communications, vol. 23 , pp. 1667-1676 , 2000