CPN Modelling And Performance Analysis Of CBHSA

Swati Singhal
PHD, Computer Science
Gurukul Kangri University
Haridwar
Aggarwalswati37@gmail.com

Heman Pathak
Associate Professor
Kanya Gurukul Campus
Dehradun
hemanp@rediffmail.com

Abstract— Security is a major issue associated with MAs and Hosts. MAs themselves may need to be protected from the hosts they visit and vice versa. For mobile multi agents, a new Cryptography Based Hierarchical Security Architecture (CBHSA) has already been proposed in our previous work. CBHSA provides four different kinds of algorithms to secure agents during migration which combines various existing security mechanisms such as encryption and decryption, signed agreement etc. This paper gives the description of Colored Petri Net (CPN) modelling of CBHSA and analyses the performance of CBHSA against some identified parameters. Different graphs have been developed for min, max and average values of different parameters. Simulation results show that CBHSA gives expected result and secure MAs and hosts from attacks.

Keywords*- Mobile Agents, Multi Agent System, Cryptography, Security, CPN Tool.*

*****

I. INTRODUCTION

Security mechanisms are necessary to safeguard the host's resources from the MAs executing on them. Similarly, MAs themselves may need to be protected from the hosts they visit. In this paper, CPN modelling of CBHSA for mobile multi agents has been described. CBHSA combines two different mechanisms to provide security. The first mechanism uses cryptography based approaches to allow secure migration of MAs while second mechanism uses reputation based trust management to protect hosts and MAs from attacks. CBHSA presents a new way to compute reputation value of both host and MAs based either on past experience or experiences of other trusted and known entities and third party. It combines various existing security mechanisms such as Intrusion Detection System, behaviour report analysis and signed agreement. CBHSA has been modelled using the timed CPN. Simulation results show that CBHSA can secure MAs and hosts from attacks but TT and ND increase as malicious rate of MA and host increases. Incorporating security features add some overheads but for low malicious rates it is not significant.

II. SECURITY REQUIREMENTS

In order to secure the host and agents from attack of each other various security measures have been identified by researchers working in the concerned areas [5][6]. Following section summarizes the MA security requirements.

A. Agent Authentication and Authorization

The origin and integrity of MAs should be verified, and agent access to host resources should be subject to an authorization check.

B. Integrity and Confidentiality

Integrity [7] and confidentiality of information in the host systems must be preserved by proper access control. An agent may carry information that needs protection with respect to, external parties that are not involved in the agent's operation. Protection against external parties has two components, one is protection against eavesdropping and modification when an agent migrates from one host to another and second is the protection of the agent when resident on a host.

C. Trust

Agents need to be capable of assessing the trustworthiness of received information [8] (e.g. by using a reputation system [9]).

D. Autonomy and migration

An agent should have control over its internal state and migration. Greater degrees of autonomy and more sophisticated migration capabilities require higher levels of security as a result of the increased risks arising from agent code manipulation.

E. Anonymity

While knowledge of the identity of an agent may be important for certain applications and services, it may not be needed by others.

F. Delegation

It must be possible for an agent to be granted rights to carry out certain tasks on behalf of another entity. The security for such a delegation act could, for example, be supported by the use of public key and attribute certificates.

III. CBHSA ARCHITECTURE

CBHSA is a framework that combines various existing security approaches to protect agents in multi agent system. CBHSA is inspired by the already existing security techniques including digital signature, encryption, signed agreement etc. proposed in [1][2]. Cryptography is a mechanism to secure data. Privacy/confidentiality(C), Authentication (A), Integrity (I), Non-repudiation (R) and Key exchange (K) are five primary functions of cryptography today. These functions can be achieved through various methods starting from physical securing to the use of mathematical algorithms for data encryption and decryption. Different kinds of keys (private and public or shared) are used for these encryption and decryption. CBHSA uses a hierarchical network environment which works at three layers (GSP, LSP & PSP). It uses centralized approach at one level and distributed at other. Network divides the open network like internet into regions and then assigns the responsibility to one of the centralized component (router) within each region to implement features to provide security for agents executing in its region. Router is an active component in CBHSA. A MA wishes to visit a host within a network, first arrives at the router of the network and then passed to the designated host. Network uses a layered architecture (3-level). The server at the lowest layer is Personal Service Provider (PSP), at the middle level Local Service Provider (LSP) and at the highest level there is Global Service Provider (GSP). Role of GSP, LSP and PSP has been described in [3].

Agent Execution, Agent Local Migration, Agent Global Migration, Agent to Agent Communication algorithms have been used in CBHSA for secure migration of MA and communication among MAs in the network, where each algorithm has two phase encryption and decryption.

IV. PETRI NETS (PNS)

Petri Net or Place Transition Net is a well-known formalism for modelling concurrency [10]. PN is a directed, connected, bipartite graph in which each host is either a place or a transition. Tokens occupy places. When there is at least one token in every place connected to a transition, the transition is enabled. Any enabled transition may fire, removing one token from every input place, and depositing one token in each output place. PNs have been used extensively in the analysis of networks and concurrent systems.

PN structure can be represented as a directed bipartite graph [11]. In a PN graph, places are represented by circles and transitions by bars or boxes. Places and transitions are connected with directed arcs. Assignment of tokens to the places of a PN structure is called its marking and represents the state of the modelled system at each time instance.

V. COLORED PETRI NETS (CPN)

CPN is a language for modelling and validation of concurrent and distributed systems and other systems in which concurrency, synchronisation, and communication plays a major role. CPN is a discrete-event modelling language combining PN with the functional programming language Standard Mark-up Language (ML). PN provide the foundation of the graphical notation and the basic primitives for modelling concurrency, communication, and synchronization. Standard ML provides the primitives for the definition of data types, describing data manipulation, and for creating compact and parameters able models [12][13]. CPN models facilitate simulation, state space analysis, behavioural visualisation, and simulation-based performance analysis. CPN differs from PNs in one significant respect; here tokens are not simply blank markers, but have data associated with them. A token's color is a schema, or type specification. Places are then sets of tuple, called multi-sets.

CPN model of a system is an executable model representing the states of the system and the events (transitions) that can cause the system to change state. CPN language makes it possible to organize a model as a set of modules, and it includes a time concept for representing the time taken to execute events in the modelled system. CPN is an industrial-strength computer tool for constructing and analyzing CPN models. Using CPN, it is possible to investigate the behaviour of the modelled system using simulation, to verify properties by means of state space methods and model checking, and to conduct simulation-based performance analysis. User interaction with CPN is based on direct manipulation of the graphical representation of the CPN model using interaction techniques, such as tool palettes and marking menus. A license for CPN can be obtained free of charge, also for commercial use. Typical application domains of CPNs are communication protocols [14], data networks [15], distributed algorithms [16], and embedded systems [17][18]. CPN are, also applicable more generally for modelling systems where concurrency and communication are key characteristics. When simulating CPNs, it is often useful to be able to examine the markings and occurring binding elements, to periodically extract information from the markings and binding elements, and then to use the information for different purposes, such as:

Stopping a simulation when a particular place is empty
Counting the number of times a transition occurs
Updating a file when a transition occurs with a variable bound to a specific value
Calculating the average number of tokens on a place

A monitor is a mechanism in CPN Tools that is used to observe, inspect, control, or modify a simulation of CPN [20][19]. Many different monitors can be defined for a given net. Monitors can inspect both the markings of places and the occurring binding elements during a simulation, and they can take appropriate actions based on the observations.

CPN is a tuple CPN = (∑, P, T, A, N, C, G, E, I):
∑ is a finite set of color types.
P is a finite set of labelled places of type ∑.
T is a finite set of labelled transitions.
A is a finite set of arcs such that: P ∩ T = P ∩ A = T ∩ A = Ø.
N is a node function. It is defined from A into P × T ∪ T × P.
C is a color function. It is defined from P into ∑.
G is a guard function, defined from T to expressions.
E is an arc expression function, defined from A into expressions.
I is an initialization function, defined from P into expressions.

VI. PROPERTIES OF CPN

Some of the properties which make CPN [21] a valuable language for the design, specification and analysis of many different types of systems are-

CPNs have a graphical representation.
CPNs have a well-defined semantics.
CPNs are very general.
CPNs have very few, but powerful, primitives.
CPNs have an explicit description of both states and actions.
CPNs have a semantics which builds upon true concurrency.
CPNs offer hierarchical descriptions.
CPNs integrate the description of control with data manipulation.
CPNs can be extended with a time concept.
CPNs are stable towards minor changes of the modelled system.
CPNs have a large number of formal analysis methods.
CPNs have computer tool.

VII. CPN MODELLING OF CBHSA

In order to evaluate the working and performance of CBHSA, it has been modelled by using CPN. Since the system model of the network remains same as discussed in [3][4]. Only the additional components related pages and their descriptions are given in this paper. To the modelling of CBHSA certain assumptions have been made. In order to model the cryptography based secure migration of MAs in LAN and GN, no mathematical details have been given only time has been added for encryption and decryption. It has also been assumed that GRT is implemented by one of the host in the GN and accessible to all hosts of the network and accessing time is constant. Any host can access or can make an entry in the table, no security checks has been used to update GRT. A malicious host is assumed to be recovered and make trustworthy by network recovery mechanism in finite time. MAs waiting to be executed on malicious hosts are blocked until it is recovered and become trustworthy.

VIII. COMPONENTS DESCRIPTION OF CBHSA

A hierarchical CPN has been used to model the CBHSA. The model uses some fusion places and substitution transitions for better representation of different components and their relations in CBHSA. Following section explains the design and working of each level of the hierarchy as shown in Figure 1.

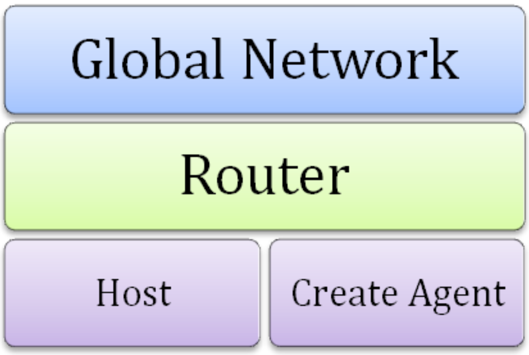

Figure 1. Page Hierarchy of CPN Model of CBHSA

A. Network Page Router

This network page models the functioning at router as shown in Figure 2. Components installed at routers are responsible for receiving/sending MAs and to enforce various security features of CBHSA. Since some of the places and their working have already been discussed in [4], this section explains the role and working of additional places and transitions only.

1) Working of Network Page Router

Place *ReceivedPacket* receives a packet from other part of the network. If target address of the packet is current network, then a cryptographic decryption DMAC is applied on the packet and transition Register is fired, which makes an entry in *LogTable*, and place the agent at place *InBuffer*. All MAs received, as well as created or executed by the hosts within the network are submitted at place *InBuffer*.

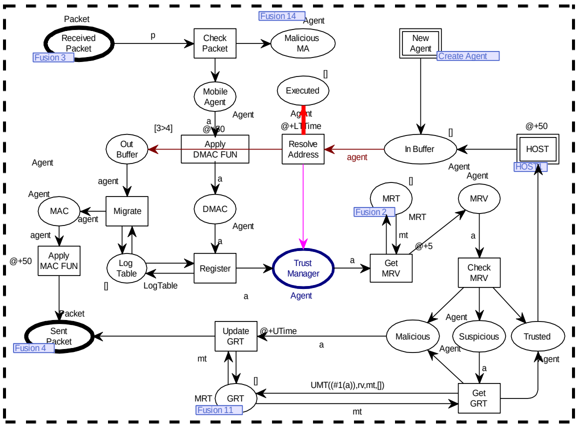

Figure 2. CPN Model of Page Router of CBHSA

A token at place *InBuffer* fires the transition *ResolveAddress*, which perform following actions –

1. If target address list is empty, MA has completed its itinerary and placed at place *Executed*.
2. If next host to be visited by the MA is in the same network then agent is passed at place *TrustManager*.
3. A token at place *TrustManager* fires the transition *GetMRV*, which collects the RV from local *MRT* (if any) and update RV of MA. MA with updated RV is placed at place *MRV*.
4. A token at place *MRV* fires the transition *CheckMRV*, which computes the trustworthiness of MA based on its RV and passes it at place *Trusted*, *Suspicious* or *Malicious*.
5. A token at Trusted means MA is trusted and passed to sub-page *HOST*.
6. A token at place *Malicious* fires the transition *UpdateGT*, which updates the *GRT* and starts the recovery.
7. A token at place *Suspicious* fires the transition *GetGRV*, which concerns the GRT to get MA's RV.
8. If there is an entry in GRT for MA, it is declared as malicious and placed at place *Malicious*. If no information is available in GRT, an entry is made in GRT but MA is updated to trusted and passed at place *Trusted*.

B. *Network Page Host*

This network page models the execution of MA at host and IDS as shown in Figure 3. Its components are responsible for successful execution of MA. In CBHSA behaviour of host and executing MAs are recorded during execution of MA and used to update their RV. To model this concept, random RVs are generated and modified in each steps of MA execution and used by transition *UpdateRT* to update RV of host and MA in their respective RT after successful execution of MA. This network page also models the IDS of CBHSA installed at router. IDS has been modelled by the transition IDS. This transition periodically creates intruders to observe the behaviour of randomly selected host. To model the behaviour report of host by intruders, random RV is generated by transition *CHKbehaviour* and host RV is updated in HRT.

1) *Working of Network Page HOST*

1. A token at place *Trusted* fires transition *CheckHost*, which concerns *HRT* to check if the target host is trusted or not.
2. If host is trusted MA is passed to place *Executing* else to the place *Waiting* where MA waits until target host becomes trusted again.
3. Meanwhile IDS periodically observes the behavior or hosts and updates their RV. A malicious host is assumed to be recovered and make trusted again by network's recovery mechanism.
4. Token at place *Executing*, fires the transition *Execute*, which model the execution of MA at the target host. It has been assumed that all MAs execute in five steps at host.
5. After execution, MA arrives at place *Execute* and in turn fires the transition *UpdateRT*.
6. MA is then placed at place *InBuffer* to continue its execution.

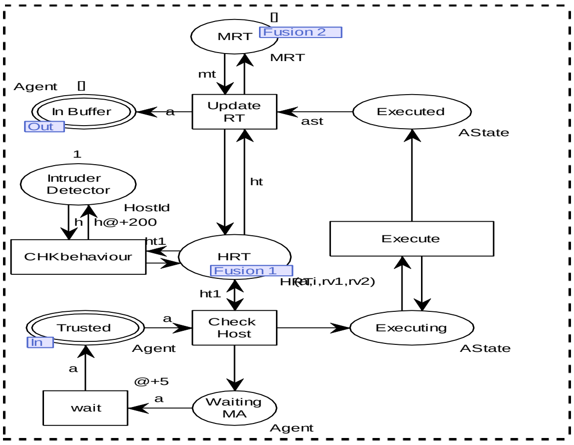

Figure 3. CPN Model of Page Host of CBHSA

IX. PERFORMANCE EVALUATION OF CBHSA USING MODEL

Before going to discuss performance analysis of CBHSA, Experimental set up of the model has been made and parameters for performance analysis have been identified.

A. *Experimental Set Up*

Performance analysis of CBHSA has been done on the basis of simulation results obtained from CPN model of CBHSA. Since timed CPN has been used to model CBHSA, time is required to be assign to some of the transitions for evaluation of performance. Before simulation starts, several parameters are required to be supposed while a few are generated at random or calculated at some stage in simulation. Some Random time has been assigned to different transitions.

The MA's itinerary contains 50 randomly selected hosts during its execution. Table 1 shows the time assign to some of the transitions of CPN model of CBHSA.

Table 1: Assignment of Different Time Units to different parameters (CBHSA)

Time Variable Declaration	Value Declaration
Encryption & Decryption Time, MA Local Migration Count	50 time units
Local Table access count	10 time units
Global Table access count	20 time units
MA Global Migration Count	100time units

B. *Parameters for Performance Analysis*

Before using the model to collect results, it needs to be setup for analysis and parameters also need to be identified for which model is to be used. Parameters identified for analysis are defined and discussed here.

1) Trip Time (TT)

When security algorithm is applied, Trip time of MA is:

$$TT=CT+ (MT+ET)*n+ ENT*p+ DCT*q+ C$$

Here p is the No. of MA encryption and q is the No. of decryption. C is a constant that model other factors that may delay MA's execution.

2) Network Overhead (ND)

ND is function of the following-

$$ND =fun (LMC*a, LTO*b, GMC*c, GTO*d)$$

Here LMC is the local MA migration count, GMC is the global MA migration count, LTO is the count of local table accessing, and GTO is the count of global table accessing. Here a, b, c and d are the weights based on size of packet/message and type of links.

X. PERFORMANCE ANALYSIS OF CBHSA

This section of the paper observes and analyses the performance of CBHSA when not all components are trusted. In order to observe the parameters TT and ND, 100 MAs have been launched. Itinerary size for each MAs are fixed (i.e. 50), it includes both local and global hosts. Experiments are repeated 1000 times and minimum, maximum and average cases are reported. Various such cases are listed below-

A. *Case 1: Trip Time Vs Malicious MA Rate*

This experiment shows the effect of malicious MAs on MA's TT.

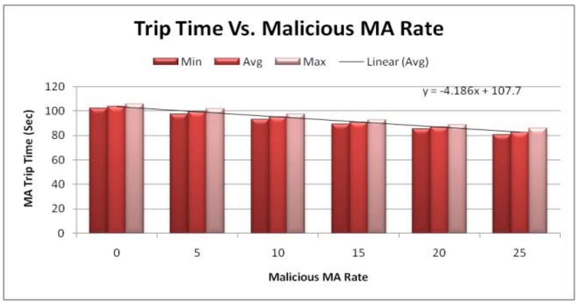

Figure 4. Trip Time Vs Increasing Malicious MA Rate

Once a MA is found malicious, it is blocked and not allowed to continue its execution. Since malicious MAs are not able to complete their itinerary, TT for this case does not give a result that can give a trend to interpret but it verifies that CBHSA is able to identify the malicious MAs and secure the hosts from their attack. Figure 4 shows the graph between min, max and average TT vs. malicious MA rate.

B. *Case 2: ND Vs Malicious MA Rate*

Figure 5 shows the graph between ND vs. malicious MA rate. Since malicious MAs terminate premature. ND decreases as malicious MA rate increases. No specific trend has been observed.

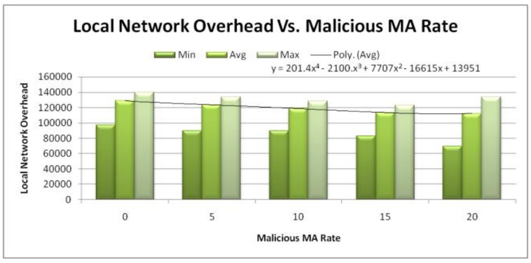

Figure 5. Local ND Vs. Increasing Malicious MA Rate

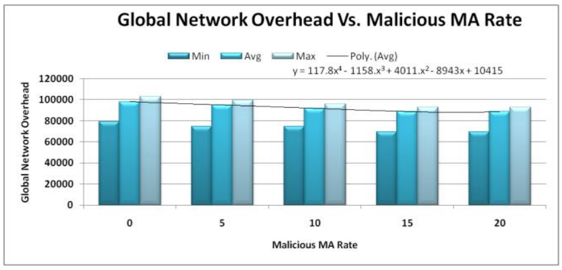

Figure 6. Global ND Vs. Increasing Malicious MA Rate

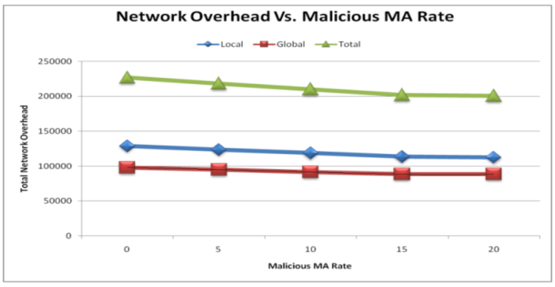

Figure 7. Total ND Vs. Increasing Malicious MA Rate

C. *Case 3: Malicious MA Vs Host Itinerary Count*

In CBHSA when a MA is found suspicious first time it is allowed to execute but an entry for suspicious MA is made in GRT. If MA is found suspicious next time it is declared malicious and blocked. An experiment has been conducted to find after how many hosts visit an intentionally introduced malicious MA is identified malicious and blocked. Figure 8 shows the graph between malicious MAs vs. host itinerary count when introduced malicious MAs detected malicious.

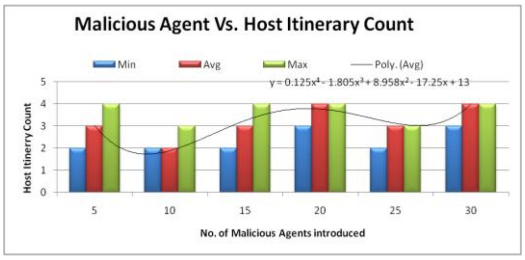

Figure 8. Malicious Agent Vs. Increasing Host Itinerary Count

It is clear from the graph that CBHSA is able to detect and block the malicious MAs with in maximum four execution steps.

D. *Case 4: Trip Time Vs Malicious Host Rate*

In CBHSA if the target host of MA is found malicious, MA is blocked and not able to continue its execution. It is equivalent to premature termination of MA. Figure 9 shows the graph between TT and malicious host rate for min, max and average cases. As malicious host rate increases more and more MA will be blocked and overall TT decreases due to blocking.

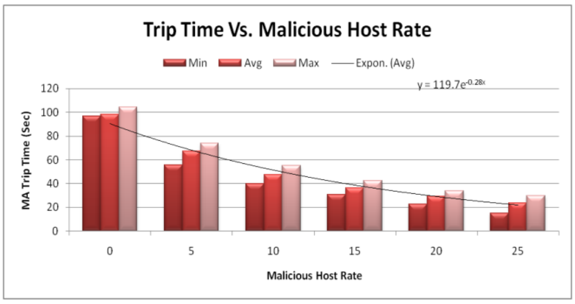

Figure 9. MA TT Vs. Increasing Malicious Host Rate

E. *Case 5: Network Overhead Vs Malicious Host Rate*

Figure 10 and Figure 11 show the graph between ND vs. malicious Host rate for local and global movements. It is clear from the graph that ND decreases as malicious host rate increase. Increased malicious host rate blocks more MAs and ND decreases due to blocking.

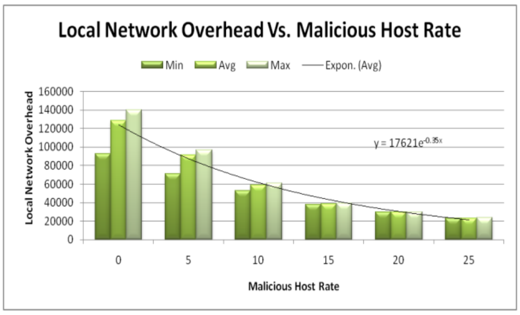

Figure 10. Local ND Vs. Increasing Malicious Host Rate

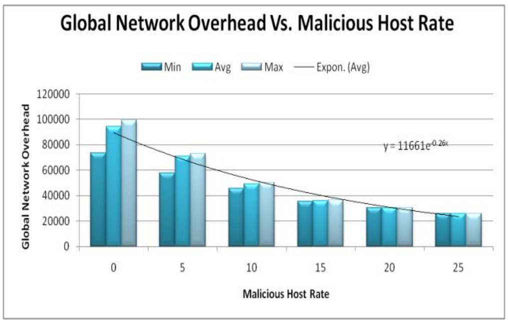

Figure 11. Global ND Vs. Increasing Malicious Host Rate

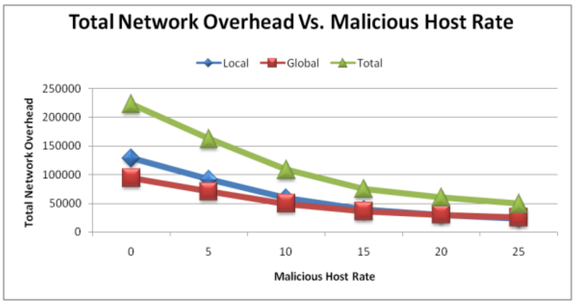

Figure 12. ND Vs. Increasing Malicious Host Rate

F. *Case 6: Number of Blocked MA Vs Malicious Host Rate*

Figure 13 shows the graph between no. of blocked MAs found vs. malicious host rate. It is clear from the graph that as more and more hosts are behaving maliciously more no. of agents will get blocked.

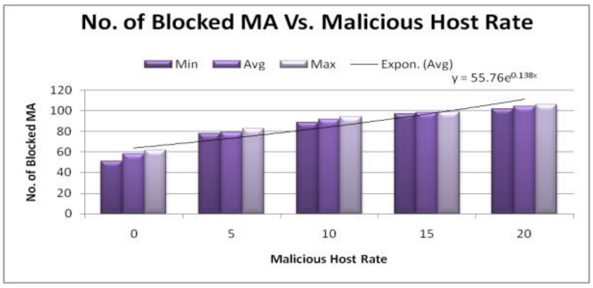

Figure 13. Blocked Agent Count Vs. Increasing Malicious Host Rate

XI. COMPARATIVE PERFORMANCE ANALYSIS OF CBHSA

In order to observe the effect of implementing CBHSA on TT and ND, it is compared with the model which does not implement any security algorithm i.e. without-CBHSA.

A. Case 1: Trip Time Vs Host Itinerary Size

Since CBHSA performs different steps for inter or intra region migration, different host addresses (local and Global) may affect the performance (TT and ND) of the system based on itinerary of MA.

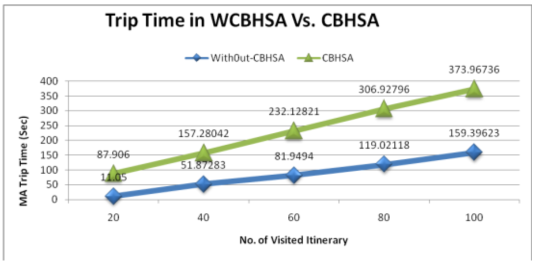

Figure 14. Comparison of TT between Without-CBHSA and CBHSA

Figure 14 below shows the comparative graph for without-CBHSA and CBHSA between TT and No. of visited itinerary for 100% intra-region migration. All the hosts and MAs are trusted. It is clear from the graph that TT increases with size of itinerary. TT for CBHSA is little higher than without-CBHSA because incorporating the security features delay the execution of MAs.

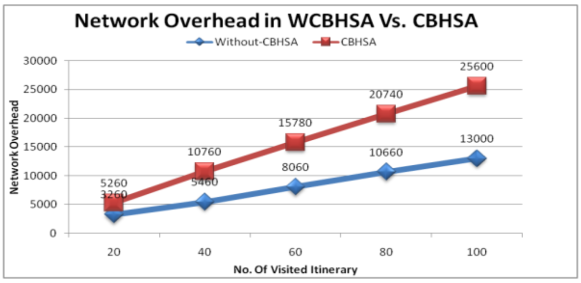

Figure 15. Comparison of ND between Without-CBHSA and CBHSA

B. Case 2: Network Overhead Vs Host Itinerary Size

Figure 15 below shows the comparative graph for without-CBHSA and CBHSA between ND vs. itinerary size for 100% intra-region migration when both hosts and MAs are trusted. For both the models ND increases as the itinerary size increases. ND in CBHSA is higher than without-CBHSA because of security measures taken in CBHSA.

XII. CONCLUSION

Security is one of the major barriers that prevent the large-scale deployment of MAS. Security concerns arise to protect the agents if the remote systems are malicious. A malicious MA may attack the hosts which enable it to execute. An agent can also attack another agent. Previous paper proposed a CBHSA framework that combines various existing security approaches to protect agents and hosts. CBHSA uses various existing security techniques including digital signature, encryption, intrusion detections, signed agreement, reputation based trust management, behaviour report analysis etc. to provide security to both MA and executing hosts. There are two types of security mechanisms in CBHSA. One is secure migration of agents in Local Area Network (LAN) and Global Network (GN) and Reputation and Trust Value computation of agents & hosts to evaluate the trustworthiness of both. To secure migration of MA in LAN and GN, Four algorithms (Agent Execution, Agent Local Migration, Agent Global Migration and Agent to Agent Communication) have been proposed, where each algorithm has two phase encryption and decryption. Use of different keys in CBHSA provides authentication, confidentiality and Integrity of MA.

CBHSA assumes that routers are trusted while hosts and MAs may be malicious. In order to detect the malicious hosts and MAs, their behaviour are observed and analysed. Based on their behaviour analysis Reputation Values (RVs) are computed. These RVs are used to evaluate the trustworthiness of hosts and MAs. If MAs are found malicious, they are blocked and reported while if a host is detected malicious its recovery starts by recovery mechanism of the network. In CBHSA, only trusted MAs are transferred to the host and host gets protected from the attack of malicious MA. Also during the execution, behavior of MA is recorded and Check point Manager saves the MA and its execution state in the LSSS periodically and MAs RVs have been computed using different components. Similarly MA is allowed only to be executed on trusted host; it gets protected from the attack of the malicious host and Hosts RVs have been computed using different components.

Host reputation value is computed by the Intrusion Detection System (IDS), PSP and executing MAs and incoming and outgoing MAs RVs is computed by PSP and last visited router only. According to RV of MAs, It is divided into three parts. Malicious (RV from 0-3), suspicious (RV from 4-6) and trusted (RV from 7 to high). To compute the RVs for the MA, observations of each entity interacted with MA must be compiled. For this reason a GRT is maintained on one of the network. This table is accessible to all the routers and assumed to fault free and trust worthy. Since accessing and updating this table is time consuming and will increase lots of network traffic, this table only maintains the list of MAs and their RVs that have been found suspicious or malicious by some watching entities. This table is concerned only when information gathered locally or from source router of MA is insufficient to make decision about the RV of the MA.CBHSA has been modelled using the timed CPN. Model is verified for its correctness and using various tools and simulations. Performance of CBHSA is then observed for identified parameters such as TT and ND. Simulation results show that CBHSA can secure MAs and hosts from attacks but TT and ND increase as malicious rate of MA and host increases. Incorporating security features adds some overheads but for low malicious rates it is not significant.

XIII. References

[1] Pathak h., “A novel flexible and reliable hybrid approach to provide security to mobile agents and the executing host”, proceedings of the international conference on electronics, information and communication systems engineering (iceice-2010), jodhpur.

[2] Pathak h., “A novel hybrid security architecture (hsa) to provide security to mobile agents and the executing host”, proceedings of the international conference on communication, computing & security pages 499-502, rourkela, 2011.

[3] Swati Singhal and Heman Pathak,” Cryptography Based Security Mechanism For Mobile-Multi-Agent Environment”, research paper published in VSRD International Journal of Computer Science & Information Technology, Vol. VII Issue XI November 2017 / 133, e-ISSN: 2231-2471, p-ISSN: 2319-2224.

[4] Heman Pathak & Swati Aggarwal,” Performance Analysis of Hierarchical Location Management Scheme to Locate Mobile Agents” paper published in International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 3, March 2016.

[5] H. Reiser and G. Vogt., “Security Requirements for Management Systems using Mobile Agents”, In S. Tohme and M. Ulema, editors, Proceedings of the Fifth IEEE Symposium on Computers & Communications, pages 160–165, Washington, DC, USA. IEEE Computer Society, 2000.

[6] N. Borselius, “Multi-agent system security for mobile communication”, PhD thesis, Royal Holloway, University of London, 2003

[7] Kristian Schelderup, Jon Ølnes, ”Mobile Agent Security – Issues and Directions”, Research paper published online.

[8] C. Castelfranchi., ”The Role of Trust and Deception in Virtual Societies,” International Journal of Electronic Commerce, 6(3):55–70, 2002.

[9] M. Schillo, P. Funk, and M. Rovatsos, ”Using Trust for Detecting Deceitful Agents in Artificial Societies”, Applied Artificial Intelligence”, 14(8):825–848, 2000.

[10] K. Jensen: High-level Petri Nets. In: A. Pagnoni, G. Rozenberg (eds.): “Applications and Theory of Petri Nets”, Informatik-Fachberichte Vol. 66, Springer-Verlag 1983, 166–180.

[11] Heman Pathak, ”Fault Tolerant Execution of Mobile Agent Systems” Thesis Submitted In Gurukula Kangri University Haridwar in 2010.

[12] J. Billington, M. Diaz, and G. Rozenberg, editors. “Application of Petri Nets to Communication Networks”, volume 1605. Springer-Verlag, 1999.

[13] Kurt Jensen and Lars Michael Kristensen and Lisa Wells, “Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems” Department of Computer Science University of Aarhus IT- Parken , Aabogade 34, DK - 8200 Aarhus N , DENMARK.

[14] J. Billington, G. E. Gallasch, and B. Han., “A Coloured Petri Net Approach to Protocol Verification”, In J. De-sel, W. Reisig, and G. Rozenberg, editors, Lectures on Concurrency and Petri Nets - Advances in Petri Nets. Proc. of 4th Advanced Course on Petri Nets, volume 3098 of Lecture Notes in Computer Science, pages 210–290. Springer-Verlag, 2004.

[15] Karnik, N. and Tripathi, A., ”A security Architecture for MAs in Ajanta,” in proceedings of the 20th International Conference on Distributed Computing Systems (ICDCS 2000), Taipei, Taiwan, pp. 402-409, April 10-13, 2000.

[16] W. Reisig, “Elements of Distributed Algorithms: Modelling and Analysis with Petri Nets”, Springer-Verlag, 1998.

[17] A. Yakovlev, L. Gomes, and L. Lavagno, “Hardware Design and Petri Nets. Springer, 2000.

[18] M. A. Adamski, A. Karatkevich, and M. Wegrzyn, editors, ”Design of Embedded Control Systems”, Springer, 2005.

[19] Anne Vinter Ratzer, Lisa Wells, Henry Michael Lassen, Mads Laursen, Jacob Frank Qvortrup, Martin Stig Stissing, Michael Westergaard, Søren Christensen, and Kurt Jensen, "CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets" in ICATPN 2003, LNCS 2679, pp. 450–462, 2003.c Springer-Verlag Berlin Heidelberg 2003.

[20] www.daimi.au.dk/CPNTools/.

[21] Kurt Jensen and Lars Michael Kristensen and Lisa Wells, “Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems” Department of Computer Science University of Aarhus IT- Parken , Aabogade 34, DK - 8200 Aarhus N , DENMARK.