A JADE Implemented Mobile Agent Based Host Platform Security

Parul Ahuja[1*], Vivek Sharma[2]

1. Scholar in Computer Science & Engg., JMIT, Kurukshetra University, Radaur, YamunaNagar-135001, Haryana, India. Ph: +919729304478.
2. Assistant Professor & Head, Department of Computer Science & Engg., JMIT, Kurukshetra University, Radaur, YamunaNagar-135001, Haryana, India.

* E-mail of the corresponding author: parulahuja89@gmail.com

Abstract:

Mobile agent paradigm relies heavily on security of both the agent as well as its host platform. Both of the entities are prone to security threats and attacks such as masquerading, denial-of-service and unauthorized access. Security fissures on the platform can result in significant losses. This paper produced a Robust Series Checkpointing Algorithm (SCpA) implemented in JADE environment, which extends our previous work, keeping in mind the security of mobile host platforms. The algorithm is Series Check-pointing in the sense that layers are placed in series one after the other, in the framework, to provide two-level guard system so that if incase, any malevolent agent somehow able to crack the security at first level and unfortunately managed to enter the platform; may be trapped at the next level and hence block the threat. The work also aimed to evaluate the performance of the agents' execution, through graphical analysis. Our previous work proposed successfully a platform security framework (PSF) to secure host platform from various security threats, but the technical algorithm realization and its implementation was deliberately ignored, which has now been completed.

Keywords: Mobile Agent, Security, Reputation Score, Threshold Value, Check-points, Algorithm.

1. Introduction

Mobile agents are the software programs capable of migrating from one host to another host in a network, at any time, at any place of their own choice [4]. An agent is typically identified with a unit of execution. A unit of execution usually carries the following information: the code (governing its behavior), the data (associated with it and necessary for its computation), and its execution state (e.g. program counter and call stack). Mobile agent systems allow migration of either the whole unit or a part (one of the above three constituents) of it. The state of the running program at one host is saved, then migrated to the new host, and then resumed the execution there allowing the program to continue from where it left off to be shifted [7, 1].

It has been suggested that mobile agent technology, amongst other technologies, can help to reduce network traffic and to overcome network latencies [3], [11]. However, an agent's ability to migrate does introduce significant security concerns. Mobile agent systems have not only incorporated security issues that have often incurred in conventional distributed systems, it also possesses some new security threats. Security threats in mobile agent systems are classified into four main categories: agent-to-platform, platform-to-agent, agent-to-agent, and others-to-agent platform. To reduce the risks of these threats, different security requirements [2], [8], [5], [4] such as confidentiality, integrity, availability, accountability, anonymity, non-repudiation; the mobile agent paradigm needs to be satisfied.

This paper is an extension of our previous work in which, we proposed an architecture for the Platform Security Framework (PSF) [6]. The framework was designed to protect the agent hosting platform from various security threats. However, the authors remained silent about the practical implementation of the framework algorithm which was left off as future work. Thus, the focus of this paper is to practically implement the algorithm in the JADE (Java Agent Development Framework) environment and presenting a result analysis of the agents' execution.

The paper is structured as follows. Section 2 provides a brief overview of the PSF architecture proposed in our earlier work. Section 3 explores the work of the eminent researchers in the area of agent technology and establishment of various security mechanisms & techniques. Section 4 presents the extension of the PSF Architecture towards the creation of the practical security algorithm namely, Series Check-pointing Algorithm (SCpA); presenting its implementation scenario in the real JADE (a framework meant for agent development in java language) environment. Section 5 shows the result evaluation and performance through graphical analysis. Section 6 concludes the paper.

2. Background

This section presents PSF architecture in brief. The proposed PSF focuses on detecting the un-trusted & malevolent mobile agents requesting platform access and then preventing the agent's computing platform from vulnerable to attacks such as includes masquerading, denial of service and unauthorized access. As is evident from literature [12], that a Digitally Signed Trust Certificate (DSTC) is issued to an agent at the time of registration, therefore, in order to establish initial trust level and prove its authenticity, every mobile agent is assumed to get registered with Central Certificate Authority (CCA). A mobile agent is able to access and provide services, only after the registration process. PSF comprises of two layers namely Authentication & Authorization Layer (AAL), and Supervision & Filtration Layer (SFL) respectively. Every incoming request for platform access first goes to AAL for authentication and privilege authorization. Once AAL approves the agent, it enters in Supervision & Filtration Layer (SFL) [6]. This is shown in figure 1.

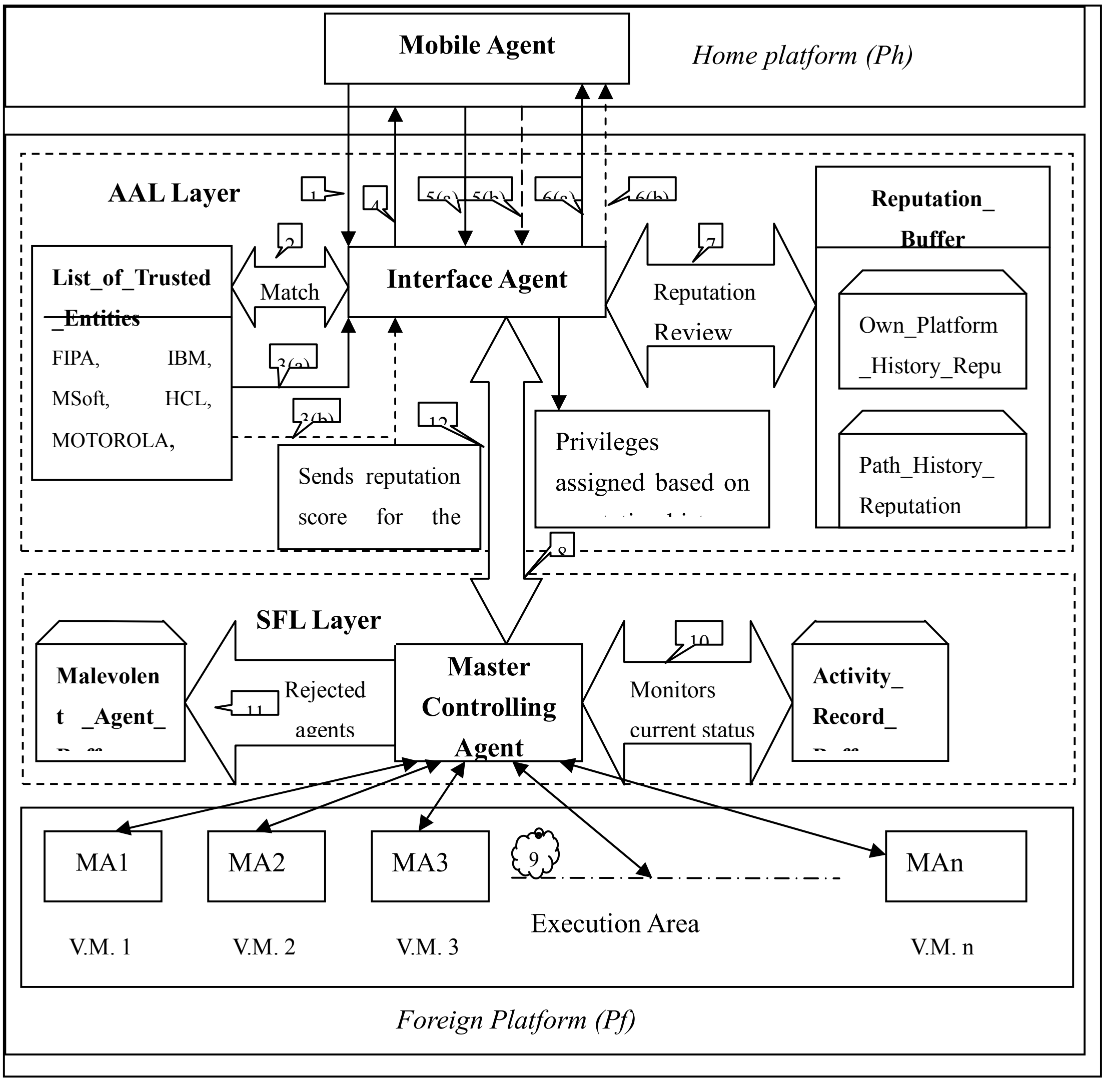


Figure 1: Architecture of the Proposed Platform Security Framework (PSF) [6]

Any *Guest Agent (GA)* can access the platform, its resources, or can perform any task, only after passing through security checks in these two layers. These layers acts as guards on the platform to prevent the security breaches, since if one layer fails, other can compensate the fault tolerance. An agent sends request from its own home platform to the foreign platform of whom it wants access. The foreign platform performs the security checks by activating the *Test Agents (IA & MCA)* in the AAL & SFL Layers.

The list-checking act and cross validation checks isolates the mechanism from literature and ensures the blocking of masquerading attack. IA either grants permission to the agent or deny the access on the basis of the results obtained from validation checks computation. AAL also assigns privileges to the accepted agents by concerning the history buffers *(OPHRB & PHRB)* maintained at IA. IA at the AAL layer then forwards the confirmed, trusted and authorized agents down to the SFL layer, where MCA assigns separate execution area to each agent and monitor their activities as a check for any misconduct or misbehaviour.

3. Related Work

This section explores the work of researchers in the fields of mobile agent and their hosting platform security and highlighted the areas of potential scope for research.

In our earlier recent research work [9], we had explored the available literature in detail and provided a brief overview of the recent researches & developments associated with the field of mobile agents, highlighted various security threats, also touching the weakest hot-spots of the field which needs to be nurtured. Many Researchers [20, 21, 22] have made fabulous developments; proposed various models, theories; introduced new technologies, techniques and frameworks, in the field of mobile agents. The authors in [13] proposed a path based security technique for mobile agents, but the fast growth in the technology suggested the extensions to support a finer granularity of trust levels. The authors in [14] proposed a mobile agent technology for the management of networks and distributed systems as an answer to the scalability problems of the centralized paradigm and sufficiently addressed the issues such as security mechanism and fault tolerance, but was surrounded by various issues such as extensive testing and practical evaluation of MAP in real-world monitoring applications by research engineers in order to identify potential deficiencies and derive methods for improving the MAP’s performance.

Similarly, the authors in [15] introduced the mobile agent technology based on quantitative hierarchical network security situational assessment model designed for large-scale network and evaluated the whole network security situation for future prediction. But the technical realization of the quantitative model for further prediction had not been discussed which degrades the quality aspects of the model. Literature [16] has elaborated the performance of Mobile agent systems compared with other cryptosystems over various parameters using Antecedence Graph Approach.

Literature review [11, 23, 10, 24] brought up the fact that although many attempts have been made to provide security in MASs (Mobile Agent Systems) communication and establishing trust among the agents, many rigid technologies developed to support security; but as the wheel of the technology spins every time, so the area always needs further refined researches in every approach we take.

4. Proposed Work Implementation And Simulation

We have implemented our security framework in JADE environment. JADE [18, 19] is a software Framework that is FIPA compliant and is fully implemented in java language. It simplifies the implementation of multi-agent systems through a middle-ware that complies with the FIPA [17] specifications and through a set of graphical tools that supports the debugging and deployment phases. The agent platform can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required. JADE is completely implemented in Java language and the minimal system requirement is the version 1.4 of JAVA (the run time environment or the JDK) and the latest version of JADE is JADE 4.1.1.

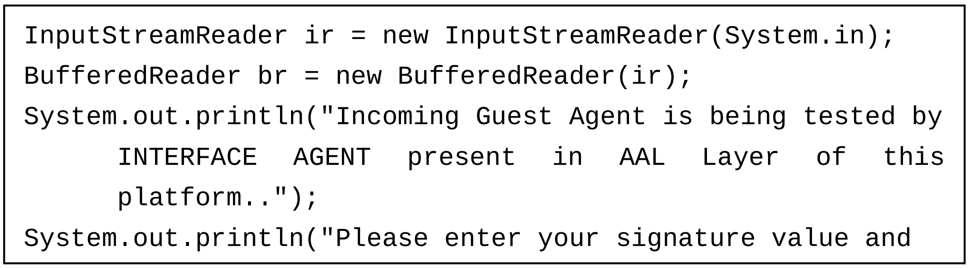
4.1 Simulating the Agent Platform

We have developed a stand alone prototype of real implementation, instead of actual implementation. Step-wise understanding the implementation of PSF-SCpA in JADE environment through partial code segments is as follows:

4.1.1 The incoming Guest Agent requests for access and simultaneously shows DSTC or other certificates obtained from CCA.

4.1.2 This request is entertained by the IA in the AAL Layer. The IA in AAL asked the GA to enter its signature value. (figure 2).

4.1.3



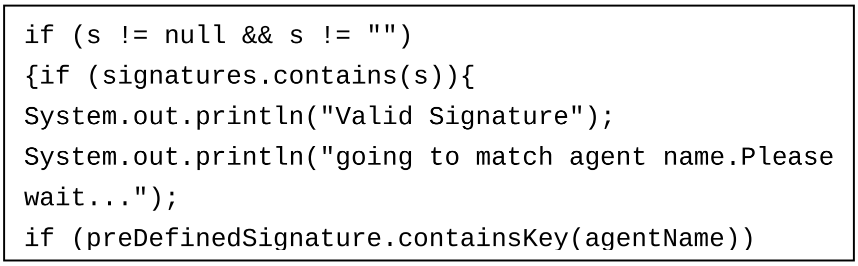
```
InputStreamReader ir = new InputStreamReader(System.in);
BufferedReader br = new BufferedReader(ir);
System.out.println("Incoming Guest Agent is being tested by
      INTERFACE  AGENT  present  in  AAL  Layer  of  this
      platform..");
System.out.println("Please enter your signature value and
```

Figure 2: Agent Authentication

4.1.4 IA then match the GA's identity against the LTE maintained in its Buffers.(figure 3).

4.1.5 On getting optimistic result after the signature matching test, the agent is considered as 'trusted' ; And 'untrusted', otherwise. (figure 3).

4.1.6 The Pf sends validation-check message to the Ph, of which the GA claims to be belonged. (Figure 3).



```
if (s != null && s != "")
{if (signatures.contains(s)){
System.out.println("Valid Signature");
System.out.println("going to match agent name.Please
wait...");
if (preDefinedSignature.containsKey(agentName))
```

Figure 3: Agent Authorization

4.1.6 If the GA prooves out to be a 'valid agent', then Pf grants it access permission and performs 'handshaking'; And dismissed permission & deny for access, otherwise. figure 4 shows the code for denied access.

```
if (preDefinedSignature.containsKey(agentName))
{System.out.println("signatures are valid but doesn't match the
      Incoming Agent's Identity. Masquerade agent trying to access
      platform. Access denied..!!");
System.out.println("Please try again");
}else{System.out.println("Invalid Signature. Masquerading attack
      attempted by unauthentic agent is blocked by IA. Access
      denied..!!");
```

Figure 4: Permission Dismissed & Access Denied

4.1.7 The GA is assigned privileges alongwith the threshold value by IA, and then enters in the SFL Layer. (figure 5).

```
System.out.println("Agent has been initialized and assigned the
      handshaking permissions ");
System.out.println("Following are the details of incoming Agent:");
System.out.println("Incoming Agent name:"+ tt.getAgentName());
System.out.println("Permissions assigned:" +accessCode
      .get(tt.getAccessCode()));
```

Figure 5: Entry In SFL Layer After Privilege Assignment

4.1.8 The MCA in the SFL Layer assigns separate execution area to each agent and monitors the current activity status of each agent.

4.1.9 All the buffers and tables are updated after the session termination.

4.1.10 On successful completion and termiation of the session, the reputation score of the GA is incremented by 1; And decremented by -1, otherwise.

4.1.11 The platform finally prepares IA and MCA in both layers for the next session and reset all the parameters.

5. Result Evaluation And Performance Analysis

The proposed PSF-SCpA outperforms in the security scan mechanism and proves efficient than other existing security techniques. The mechanism follows *'shades-of-grey policy'* to achieve perfect security of the host platform. According to the policy, the mechanism doesn't handshake any agent immediately after knowing it trusted, nor it concludes any agent malevolent and terminates its session by knowing it un-trusted. It further investigates and searches for the whole spectrum to grant privileges to the agent, i.e. it carries validation checks by cross-checking with the GA's home platform, to get the accurate identity status of incoming guest agent. In this way, an agent is not prone to be denied for access by mistakenly grading it bad, and nor it is provided access to the services for which it is not entitled. The *'cross validation-check'* activity of PSF isolates it from other techniques and proves better. Furthermore, the *'reputation score'* mentioned in the mechanism is actually a performance parameter. The performance of any agent can be depicted from the RS value. After the computation of RS at SFL Layer, this information is returned back to the AAL Layer so that IA could decide the level of privileges to be assigned to GA's in the next session.

The code may be replicated for any frequency of agents according to the application requirement. To analyze the efficiency of the security framework, we need a large number of agents starving for access, to be bombarded on the platform. The above setup is run for gaining insight into the performance against some of the earlier developed security mechanisms and technologies, such as AG based Non-Checkpointing approach (NCpA) [16] and Parallel Check-pointing algorithm (PCpA) [16], and our own proposed platform security framework (PSF) i.e. a robust two layered Series Check-Pointing Algorithm (SCpA).

5.1 Graphical Results Using Performance Metrics

The agents' execution is simulated in JADE environment using API of JGraphT library developed in Java programming language and the comparative analysis is made between the proposed PSF-SCpA (Architectural & Algorithmic approach) framework and other existing techniques, for securing mobile agent hosting platform against security threats. The evaluated results are compared through graphical analysis.

The performance & efficiency of the proposed PSF is mainly determined by the following three metrics: the identifying time, the waiting time and the total application message overhead. We vary the number of mobile agents from 10 to 100 to see the corresponding changes of these performance metrics. The graphical comparison is as follows:

5.1.1 Average Identifying Time:

It is the time which is required to determine that, which and how many mobile agents should go through security-checks placed in series in the framework.

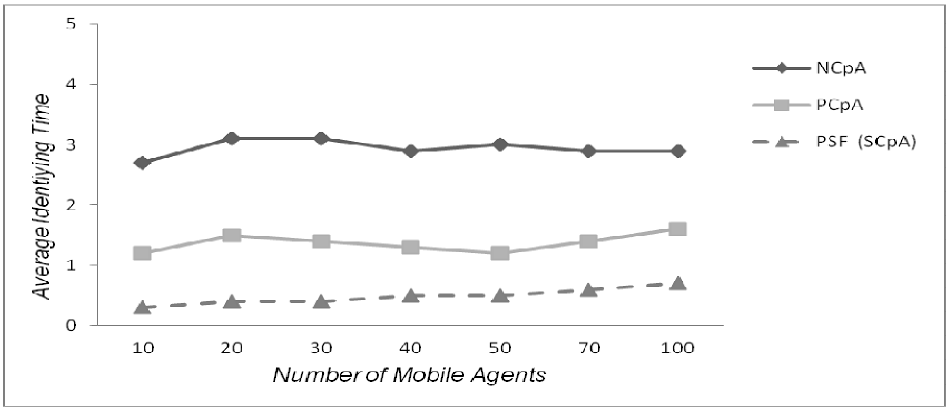


Figure 6: Average Identifying Time Comparison

The average identifying time is an important performance metric, especially for certain time critical applications. Fig. 6 illustrates the average identifying time comparison for NCpA (means when no security is implied), PCpA and PSF (SCpA). We observe that though, the time is increasing with increase in the number of mobile agents, but still our proposed PSF takes less time to identify malevolent MA's (GA's) as compared to other previously existing techniques. This is because, with in the two layers of the PSF, are the two test agents or base agents (IA & MCA) which remains active all the time and continuously monitors, controls, manages and maintain all records on the platform. All the buffers are readily & timely updated and hence the latest information in the framework flows within a few seconds to IA & MCA so that they can act accordingly and dealt well with GA's.

5.1.2 Average Waiting Time:

The average time which a MA (GA) waits in response from PSF's guards involved in maintaining and securing the platform. The waiting time is proportional to MA-to-Platform, Platform-to-MA and MA-to-MA message latency. *Latency* is the time span from the sending of message from one agent to the receipt of message from other agent. Rise in the latency time may leads to hikes in the waiting time y-axis.

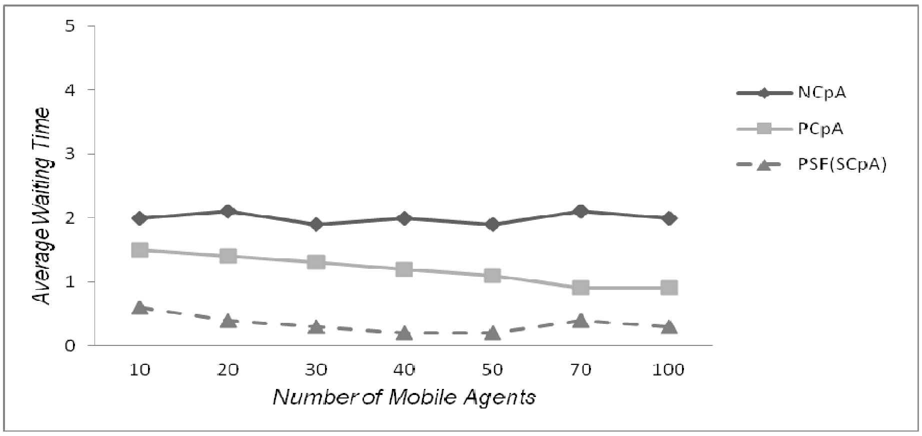


Figure 7: Average Waiting Time Comparison

The figure 7 shows that average waiting time decreases with the constant increase in the frequency of incoming mobile agents. The effect is similar in PCpA but PSF's slope is even much lower than others. The waiting time is less in our PSF because the security layers are arranged in series. Any GA who arrives at the platform passes through these layers in sequence one by one. Each layer intakes and entertains one agent at a time and hence the processing time is reduced which further minimize waiting.

5.1.3 Total Application Message Overhead:

The message overhead includes the request and reply messages from MA (GA) to platform and vice-versa.

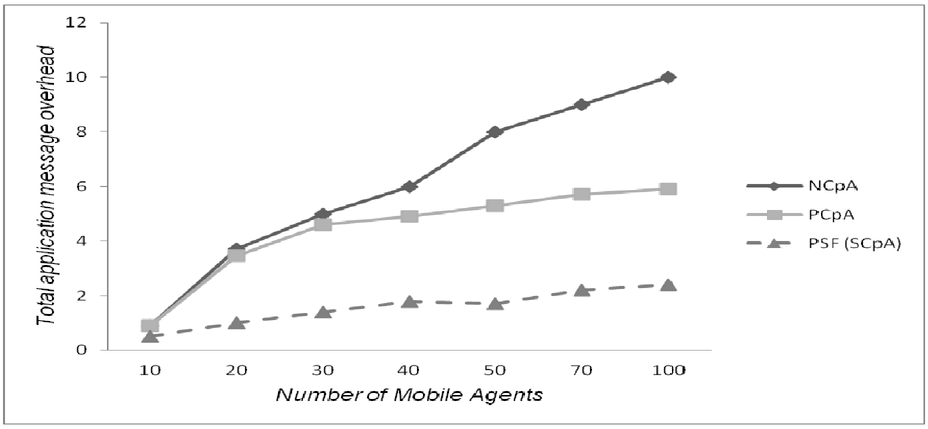


Figure 8: Total Application Message Overhead Comparison

The figure 8 above illustrates the total application message overheads as a function of number of mobile agents. From figure 8, we can see that, though the message overhead increases with the increase in number of mobile agents but comparative to other existing techniques, is reduced to greater extent after implying PSF (SCpA) for securing host platform. The reason lies in the occurrence of threshold events, which exempts the execution timely and hence reduced the unnecessary message exchanges. The worst case is that it increases slowly instead it should not, and the best-case is that the overall slope did not hiked so high and remained closer to the horizontal axis.

6. Conclusion

The two layered phenomenon provided by the PSF-SCpA is very active & flexible in securing the host platform. The 'cross validation-check' activity of PSF isolates it from other techniques and proves better. The use of 'reputation score' enables the platform to estimate the character status of an agent for future interactions. Since, the agent technology advances continuously and has made significant contributions in the area of code mobility and security, it would not be wise to ignore this fact and try to reinvent the wheel every time in every new approach we take. So, the proposed framework may be extended to include the datum of the buffers: PHRB, ARB & MAB. The SCpA prototype implementation may be extended to include the rollback of the reputation score to the AAL layer, for further security processes. By integrating solutions already tested in other domains of the framework, we can build on the top of these and provide more sophisticated approach, which may tackle the ever increasing complex security attacks.

References

[1] Martin L.Griss, Ph.D. (2001), "Software Agents as Next Generation Software Components", In Component-Based Software Engineering: Putting the Pieces Together, Addisson-Wesley publications, May 2001.

[2] Wayne Jansen and Tom Karygiannis (1999), "Mobile Agent Security", In NIST Special Publication, Vol. 800, issue-19, pp. 39, 1999.

[3] Gian Pietro Picco (2001), "Mobile agents: an introduction", In Microprocessors and Microsystems, Vol. 25, pp. 65-74, 2001, Milan, Italy.

[4] Mousa Alfalayleh, and Lijilana Brankovic (2005), "An Overview of Security Issues and Techniques in Mobile Agents", In International Federation for Information Processing (IFIP), Vol. 175, pp. 59-78, October 2005.

[5] S.M. Sarwarul Islam Rizvi, Zinat Sultana, Bo Sun, and Md. Washiqul Islam (2010), "Security of Mobile Agent in Ad hoc Network using Threshold Cryptography", In World Academy of Science, Engineering and Technology 70- 2010.

[6] Aarti Singh and Parul Ahuja (2012), "Robust Algorithm for Securing an Agent Hosting Platform", In International Journal of Advancements in Technology (IJoAT), Vol. 3, Issue 2, April 2012.

[7] Nick Jennings and Michael Wooldridge (1996), "Software Agents", IEE Review, January 1996, pp. 17-20.

[8] Priyanka Dadhich, Dr. Kamlesh Dutta, and Prof.(Dr.) M.C. Govil (2010), "Security Issues in Mobile Agents", In International Journal of Computer Applications(0975-8887), Vol. 11, Issue 4, December 2010.

[9] Parul Ahuja and Vivek Sharma (2012), "A Review on Mobile Agent Security", In International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Volume-1, Issue-2, June, 2012.

[10] Li An, Qiangfeng Jiang, Xiaoping Luo, and Zhaohui Ren (2002), "Protecting Mobile Agents Against Malicious Hosts", In CS685-002 Term Paper, Spring 2002.

[11] Niklas Borselius (2002), "Mobile agent security", In IEEE Journal of Electronics & Communication Engineering , Vol. 14, issue 5,pp. 211-218, October 2002.

[12] Aarti Singh, Dimple Juneja, and A.K. Sharma (2011), "Elliptical Curve Cryptography Based Security Engine for Multiagent Systems Operating in Semantic Cyberspace", In International Journal of Research and Review in Computer Science (IJRRCS), Vol. 2, No. 2, April 2011.

[13] G. Knoll, N. Suri, and J.M. Bradshaw (2002), "Path-based Security for Mobile Agents", Electronic Notes in Theoretical Computer Science, Vol. 58, Issue 2 , pp. 16, 2002.

[14] D. Gavalas, G.E. Tsekouras, C. Anagnostopoulos (2009), "A mobile agent platform for distributed network and systems management", In Journal of Systems and Software Vol. 82, Issue 2, pp. 355-371, 2009.

[15] C. Xiaorong, L. Su, L. Mingxuan (2012), "Research of Network Security Situational Assessment Quantization Based on Mobile Agent", Vol. 25, pp. 1701–1707, International Conference on Solid State Devices and Materials Science, April 1-2, 2012.

[16] R. Singh and M. Dave (2011), "Antecedence Graph Approach to Checkpointing for Fault Tolerance in Mobile Agent Systems", IEEE Transactions On Computers, 2011.

[17] http://www.fipa.org (1997), "Foundation for Intelligent Physical Agents, Specifications", 1997.

[18] Fabio Bellifemine, Agostino Poggi, and Giovanni Rimassa (2001), "Developing Multi-agent Systems with JADE", University of Parma, In Springer-Verlag Berlin Heidelberg, pp. 89–103, 2001.

[19] Aarti Singh, Dimple Juneja, and A.K. Sharma (2011), "Agent Development Toolkits", In International Journal of Advancements in Technology (IJoAT), Vol. 2, No. 1 (January 2011).

[20] S. M.. Moussa, G.A. Agha (2010), "Integrating Encrypted Mobile Agents with Smart Spaces in a Multi-agent Simulator for Resource Management", Journal of Software, Vol 5, No 6, 630-636, Jun 2010.

[21] A. Saxena and B. Soh (2005), "Authenticating Mobile Agent Platforms Using Signature Chaining Without Trusted Third Parties", In IEEE International Conference on e-Technology, e-Commerce, and e-Services, pp.282-285, 29 March - 1 April 2005, Hong Kong, China.

[22] M. Soriano and D. Ponce, Technical University of Catalonia (2002), "A Security and Usability Proposal for Mobile Electronic Commerce", IEEE Communications Magazine, developed as part of the project ACIMUT CICYT TIC2000-1120-C03-03. August 2002.

[23] William M. Farmer, Joshua D. Guttman, and Vipin Swarup (1996), "Security for Mobile Agents: Issues and Requirements", In Proceedings of the 19th National Information Systems Security Conference,Vol. 2, pp. 591-597. National Institute of Standards and Technology,Baltimore, Maryland, October 1996.

[24] O.A. Ojesanmi and Ajai Crowther (2010), "Security Issues in Mobile Agents", In International Journal of Agent Technologies and Systems, Vol. 2, Issue 4, pp. 39-55, October-December 2010.

Parul Ahuja has done B.Tech (CSE) in Honours and is pursuing M.Tech (CSE) Dissertation from Kurukshetra University. Her research work is in the field of Agent Technology. She has published technical and review papers in national / international journals. Her specialization among programming languages includes C, C++ and java; and among databases includes Oracle 9i. She has also keen interest in the fields of network management & security, mobile computing, internet fundamentals, data structures and design & analysis of algorithm. She has delivered seminars on various technologies such as mobile IP, cloud computing, and android technology.

Vivek Sharma has done B.Tech (CSE) & M.Tech (CSE) from Kurukshetra University and achieved gold medallist during his graduation. His publications includes the field of wi-fi protocols, mobile sensor networks in health care. His specialization is in object oriented programming languages. His research area is in mobile networks. He is a member of CSI (Computer Society of India).

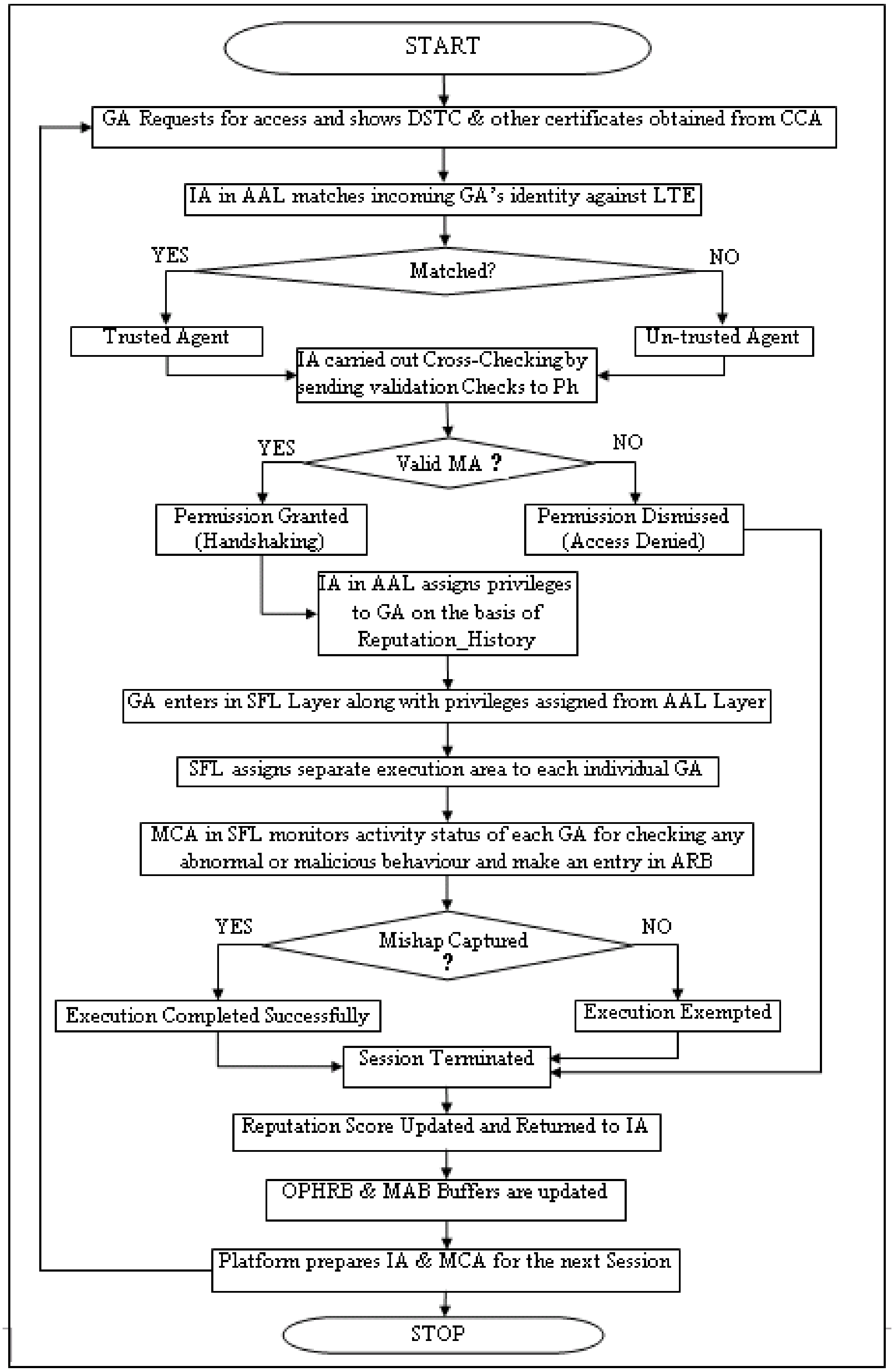

START
GA Requests for access and shows DSTC & other certificates obtained from CCA
IA in AAL matches incoming GA's identity against LTE
YES
Matched?
NO
Trusted Agent
Un-trusted Agent
IA carried out Cross-Checking by sending validation Checks to Ph
YES
Valid MA ?
NO
Permission Granted (Handshaking)
Permission Dismissed (Access Denied)
IA in AAL assigns privileges to GA on the basis of Reputation_History
GA enters in SFL Layer along with privileges assigned from AAL Layer
SFL assigns separate execution area to each individual GA
MCA in SFL monitors activity status of each GA for checking any abnormal or malicious behaviour and make an entry in ARB
YES
Mishap Captured ?
NO
Execution Completed Successfully
Execution Exempted
Session Terminated
Reputation Score Updated and Returned to IA
OPHRB & MAB Buffers are updated
Platform prepares IA & MCA for the next Session
STOP

Figure 9: Scenario Behavior Flow Diagram of the PSF-SCpA

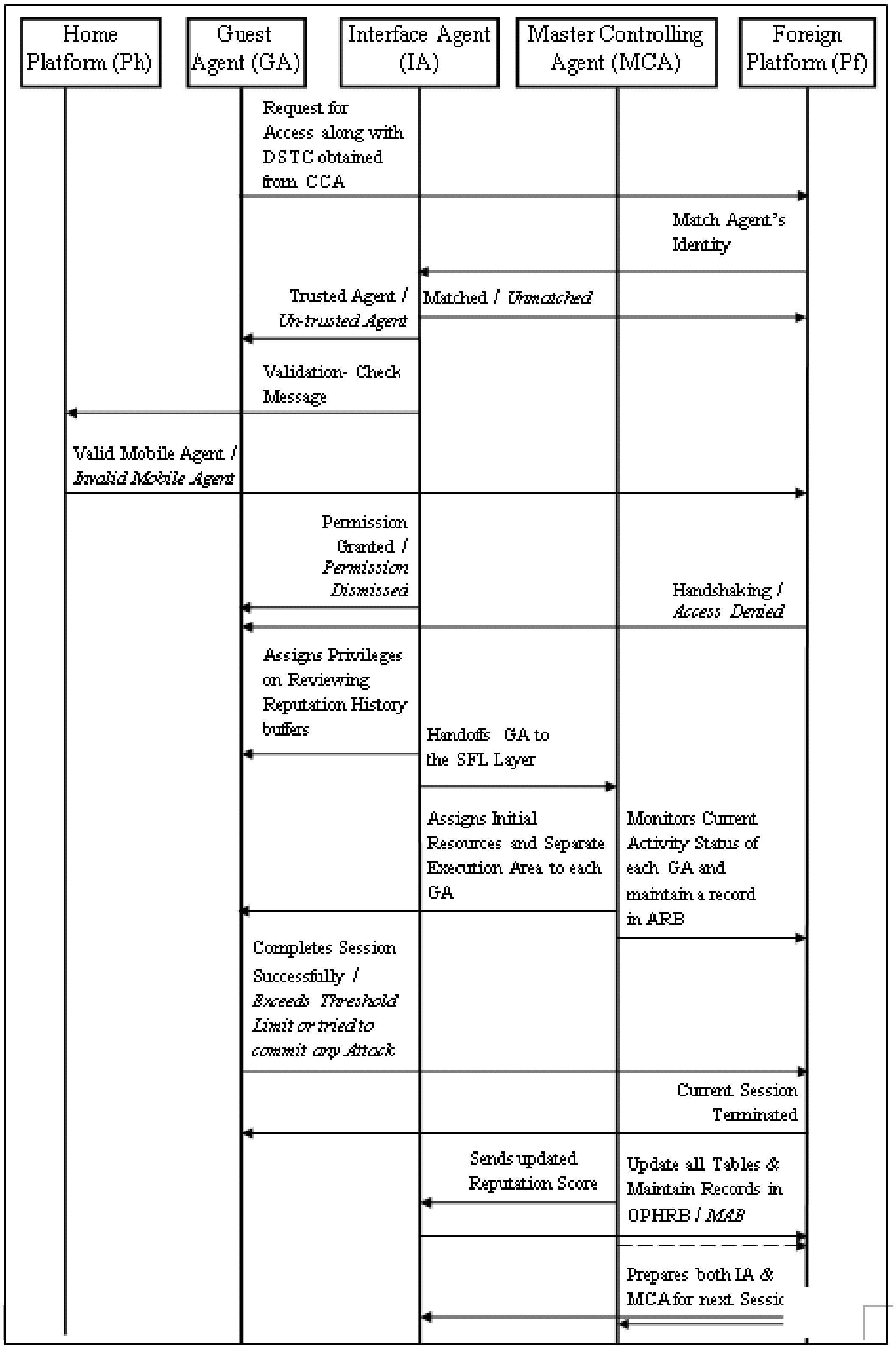

Home Platform (Ph)
Guest Agent (GA)
Interface Agent (IA)
Master Controlling Agent (MCA)
Foreign Platform (Pf)
Request for Access along with DSTC obtained from CCA
Match Agent's Identity
Trusted Agent / *Un-trusted Agent*
Matched / *Unmatched*
Validation- Check Message
Valid Mobile Agent / *Invalid Mobile Agent*
Permission Granted / *Permission Dismissed*
Handshaking / *Access Denied*
Assigns Privileges on Reviewing Reputation History buffers
Handoffs GA to the SFL Layer
Assigns Initial Resources and Separate Execution Area to each GA
Monitors Current Activity Status of each GA and maintain a record in ARB
Completes Session Successfully / *Exceeds Threshold Limit or tried to commit any Attack*
Current Session Terminated
Sends updated Reputation Score
Update all Tables & Maintain Records in OPHRB / *MAB*
Prepares both IA & MCA for next Sessic

Figure 10: Scenario Behavior Interaction Diagram of PSF-SCpA