Computers, Materials & Continua
DOI: 10.32604/cmc.2022.025582
Article

Tech Science Press

Developing Check-Point Mechanism to Protect Mobile Agent Free-Roaming Against Untrusted Hosts

Tarig Mohamed Ahmed*

Department of Information Technology, FCIT, King Abdulaziz University, Jeddah, Saudi Arabia
*Corresponding Author: Tarig Mohamed Ahmed. Email: Tmahmad@Kau.edu.sa
Received: 29 November 2021; Accepted: 21 February 2022

Abstract: Mobile Agent has many benefits over traditional distributed systems such as reducing latency, bandwidth, and costs. Mobile Agent Systems are not fully utilized due to security problems. This paper focuses on mobile agent protection against malicious hosts. A new security mechanism called Checkpoints has been proposed. Checkpoint Mechanism (CPM) aims to protect Mobile Agent against malicious hosts in case of Capturing and Integrity attacks. CPM assumes using a free-roaming mobility mechanism by Mobile agent systems. The main idea behind CPM is to generate multiple versions of Mobile Agent. The multiple version is used to recover Mobile Agent from Capturing and Integrity attacks by untrusted hosts. MA versions are kept in Recovery Host(RH). RH plays a key role in CPM by controlling and monitoring MAs' recovery processes. A prototype method has been used to prove the feasibility of CPM. The prototype was implemented by using the .Net framework and C#. full discussion for several scenarios has been done to analyze the feasibility and performance for CPM. As found from this research, CPM has a strong ability to protect Mobile Agents from Capturing and Integrity attacks completely. In addition, there is no negative impact on the overall performance of the mobile agent system.

Keywords: Mobile agent; security capturing; integrity attacks

1 Introduction

Mobile Agent systems are a promising area for computers to communicate through networks. Mobile Agent(MA) allows users to perform their tasks efficiently by using an asynchronous mode [1]. MA is an object that has the ability to move autonomously among a network node to perform tasks on behalf of users [2]. It has many benefits over traditional ways such as reducing the network bandwidth, low network latency and reducing the communication cost [3]. Through a journey, MA visits hosts that provide services based on user requirements [4].

Mobility is a key feature for MA systems that allows MAs to move in heterogeneous networks to perform tasks on behalf of the user [5]. The mobility could be one of two mechanisms such as a predefined itinerary (Static) and a free-roaming (Dynamic) [6]. The predefined itinerary mechanism is used when all visited hosts are known before MA starts its journey [7]. The free-roaming mechanism is used when MAs visit hosts are unknown and MA will move among hosts based on execution environment conditions [8]. This paper deals with the second one the free-roaming [9–11].

MA deals with many entities during its execution so, security is highly required. The MA security is classified into two main categories: platform security and MA security (As Object) [12]. The platform security aims to protect hosts (service providers) against malicious MAs such as accessing unauthorized information or acting like viruses, [13,14]. On the other hand, MA security aims to protect MA against malicious hosts that may attack the MA integrity. In addition, malicious hosts may prevent MAs to continue their journeys (Capturing) [15]. Two main approaches to deal with MA security issues: detection or prevention approaches. The detection approach is to find if there is a problem that occurred and try to solve it by using different mechanisms such as tracing MA results or using cryptography mechanisms [16,17]. Most of the proposed mechanisms are used to protect MA systems partially but not completely. This paper addresses unsolved security issues completely by current security mechanisms in case of using a free-roaming mobility mechanism, such as [18–20]:

1. MA capturing Attacks by malicious hosts.
2. MA integrity Attacks by malicious hosts.

This paper proposes a mechanism to protect MAs against two types of attacks Capturing and Integrity. In addition, it provides a fault tolerance mechanism in case of normal failure that disallows MAs to continue their journeys. The proposed mechanism is called Check-Point Mechanism (CPM). CPM is based on two concepts: Checkpoints to protect MA from capturing by malicious hosts and MA Multi-Version to protect the MA integrity during a journey. the mechanism details will present in the below sections.

The paper rest is organized as follows: Section 2 presents some proposed mechanisms related to MAs protection. Section 3 explains a method of CPM to protect MA. The method presents the mechanism framework, components, and algorithm. Section 4 shows the result of the MA system with CPM and without CPM implementation as a prototype. Section 5 discusses the results in terms of CPM feasibility and performance. The paper is concluded in Section 6 with some recommendation points as future works.

2 Related Work

This paper focuses on MA protection against malicious hosts. When it uses free-roaming as a mobility mechanism. This section presents some proposed mechanisms in the same area of security to specify the research gap as follows:

Roth [21] proposed a schema that allows mutual recording of MA itinerary with cooperating Mobile Agent Systems. While MA roaming among different platforms, it provides information about current, previous, and next platforms to follower MA to check the consistency by making authentication. In case of any violation, fitting action is done. Yee [22] developed a mechanism to ensure MA integrity during its journey. The mechanism was based on an encryption mechanism. In each place visited by MA, the result is encrypted. When MA returns home, all results are verified to make sure no integrity attack occurred. But, encryption has a bad impact on overall performance. To check MA integrity, Vigna [23] proposed to make tracing the execution history of MA. If there is any type of conflict in the execution, the host of that execution will be considered as a malicious host. Esparza [24] suggested watermarking approach to detect attacks against MA. This approach uses a digital watermark to be added in MA during its journey. When MA finished its duty, the result is verified to find the mark. If the MA code is changed, the watermark becomes defective and scrambled [25]. Tajer et al. proposed a framework to support MA agents during their journeys. The framework aims to protect MAs against Eavesdropping and Alternation attacks. The idea is based on evaluating how different types of MAs react with attacks. In this way, the system can build trust in hosts [26]. Address Forward and Data Backward (AFBD) is a mechanism to protect MA's integrity. MA sends a result of each visited host before moving to another. In this case, MA only carries the hosts' addresses. In this way, the obtained results from hosts will be completely protected. But sometimes, the results may be needed in computation processes during a journey [27]. Secure Image Mechanism (SIM) was proposed by Ahmed [28]. A copy of MA is sent to a Controller (Trust Host) before sending to an anonymous host for execution. MA back to the controller after finishing the execution to be verified. If any modifications occur, the mechanism declares the host as malicious [29]. Muñoz et al. proposed a trusted platform model to protect MAs based on hardware components. The model is implemented as a chip that is integrated into hardware. It has two activates first one, configuration justification of hosts. The second one, a host data protection. This model works as a protocol that is enforced to migrate MA [30]. Trusted security mechanism proposed by Dadhich et al. It is based on Trust decisions which are combined with security decision making. The security decision is taken based on security management [12]. Trust Management involves organizing the information necessary to pass a trust relationship. Trust specifies a relationship between two objects on a specific statement. In addition, it is described using belief levels, disbelief, and uncertainty [31].

As mentioned above, these security mechanisms were not given a solution regarding capturing and integrity as a security prevention method. In addition, the Fault-Tolerance feature is missing.

3 Method and Material

The main idea behind this paper is to propose a novel mechanism that has the ability to protect MAs against malicious hosts in two situations: first, Capturing MAs by preventing them to continue their journeys. The second, MAs Integrity by detecting and preventing at the same time. The mechanism is called Check Point Mechanism (CPM). The following sections describe CPM in terms of a framework, components, and algorithm.

CPM Description

CPM aims to protect MAs against Malicious hosts. The CPM assumes there are two types of hosts: Trusted Hosts (TH) and Untrusted Hosts (UT). MA can Visit multiple hosts either TH or UT such as $th_1, th_2, \ldots th_n$ N is a number of trusted hosts and, $ut_1, ut_2, \ldots ut_m$ M is the number of untrusted hosts. When a MA visits any type of host, a new version of the MA such as $ma_1, ma_2, \ldots, ma_k$ K is the number of total hosts that are visited by MA. The main entities of CPM are described as follows:

Mobile Agent (MA)

MA is an object that can travel autonomously among hosts to perform some duties on behalf of users. It consists of two main parts: first, data state represents information about MA, required data for processing in different hosts, and results that are obtained during the journey. The second part represents the behavior of MA that changes the MA state. The behavior is constituted based on services that are requested by the users. MA moves from one host to another by using mobility property. The proposed mechanism assumes that MA uses a free-roaming method as a mobility mechanism. Free-roaming means MAs are not equipped with itinerary tables. When MA reaches the first station, will get information about the next station based on required services. During a journey, multi versions of MAs are generated such as ma_1, ma_2, . . . , ma_k K is a number of total hosts that are visited. Fig. 1 presents MA components.

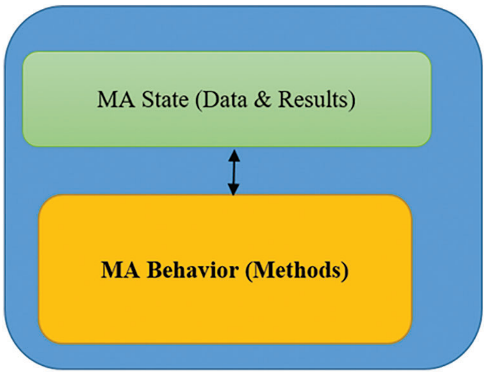


Figure 1: MA components

3.1 Mobile Agent Base (MAB)

Mobile Agent Base (MAB) represents MAs home. From this place, MAs start their journeys and after finishing their duties return to MAB. It receives requests from users and provides MAs with these requests. After completing its journey, MA returns to MAB with results. MAB extracts the results. In addition, it monitors MAs during their journey. CPM allows MAB to verify the results by using information from Recover Host (RH). RH is one of the key players in CPM. It saves image copies of MA versions during it is journey to protect it against malicious hosts. MAB consists of several components such as: MA Creator: This component aims to create MA according to the user's request. In addition, it provides MA by needs during its journey for example the first station of its journey and some stations as backup hosts that may need it. The backup hosts will be used in case the visited hosts fail to provide MA with an appropriate host as a next station according to its mission. MA Dispatcher aims to transfer MA to its first station. In case of any problem transferring MA to the next host, the Dispatcher will select one of the backup hosts that is stored inside MA. MA Receiver: This component receives MAs after completing their journeys and extracting the obtained results. MA Verifier: By using MA versions from RH, this component verifies MA integrity by making tracing. MA reporter: This component aims to prepare the result and submit it to the users. Fig. 2 is USE CASE diagram presents MAB components.

3.2 HOSTS

Hosts provide MAs with services based on their mission. There are two types of hosts: first, a trusted host (TH) that is completely secure to MAs. The second, an untrusted host (UT). The untrusted host may capture MAs or attack their integrities. All hosts in CPM are considered to belong to one of the two types. To protect MAs against UTs. Before, MA visits UT, a copy of the current version will be kept in RH. So, RH may keep multiple different versions of MA. The host composes of components to achieve its role such as Services: Services are requested by visited MAs. Hosts Database: this component provides MAs with the next hosts in their journeys according to their missions. Because in our proposed mechanism, MAs use the Free-Roaming mechanism for mobility. Recovery Unit: this unit is only available in trusted hosts. It aims to send a MA copy to RH in case the next station is an untrusted host.

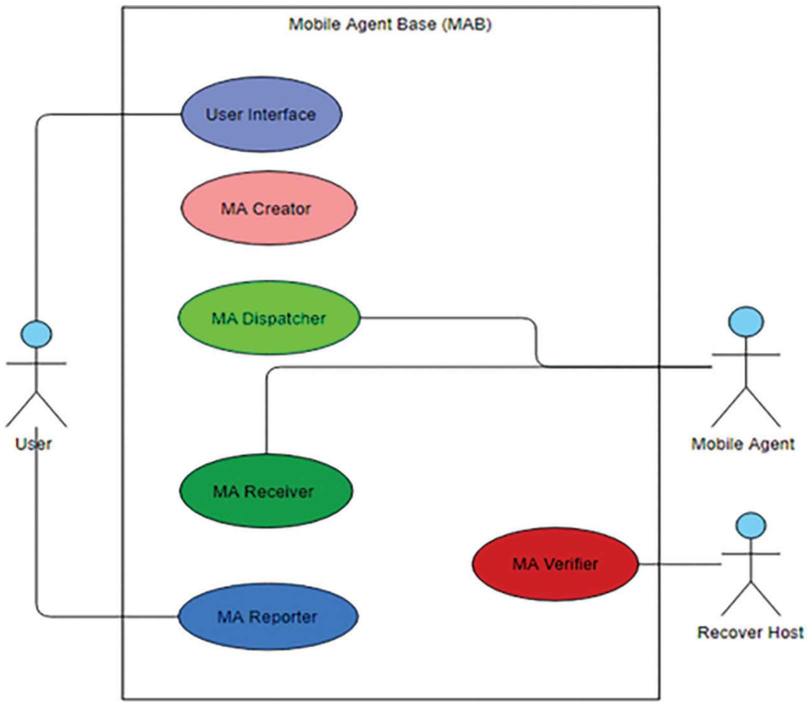


Figure 2: MAB use case

3.3 Recovery Host

Recovery Host(RH) plays a key role in CPM. It keeps multi versions of a MA to grantee protection against MAs capturing integrity attacks. The latest version could work instead of MA in case UT prevents an original MA to continue or being lost for any technical reason (fault-tolerance). The multi versions of the MA are used to verify the MA's integrity by MAB. MAs are kept in RH when a checkpoint is performed. The checkpoint is performed in two cases. First, before MA visits UT. The second, if a user sets multiple checkpoints when MA is created. For example, the user may set a number of checkpoints during a journey. By this, the user can monitor his/her MA's progress. To accomplish its duties, RH stores MAs multi-version in a stack and it uses PUSH and POP operation to store and retrieve MA versions. Fig. 3 presents the USE CASE of RH.

3.4 CPM Algorithm

This section describes the logic steps of CPM by using the entities that are described above. Before MA starting it is journey, MAB send a message to RH to create a stack for the MA versions. These versions are generated when a MA visits TH or UT. The MA version will be sent to RH in two cases: first, before MA visiting UT. Second, checkpoints are settled by MAB. Periodically, RH sends signal to MAB about a status of MAs based on receiving MA versions. Tab. 1 presents CPM algorithm that are followed by MAs during journeys:

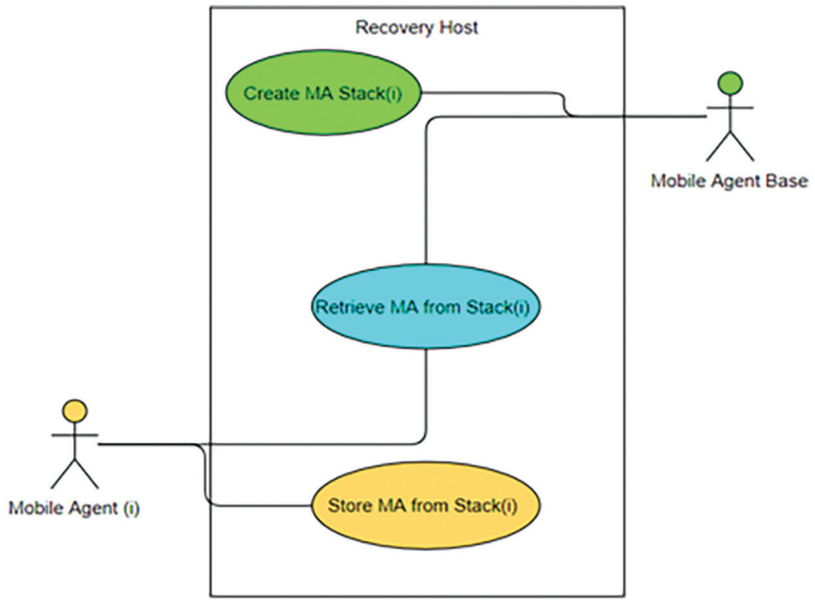


Figure 3: Recovery host use case

Table 1: CPM algorithm

#	Steps	Location	Output
1	Create MOBILE Agent(*user_tasks*)	MAB	Ma_0
2	Dispatch(ma to th1)	MAB	
3	Receive ma by th1	th_1	
4	Serve (ma)	th_1	ma_1
5	If(ma satisfied user requirements)	th_1	
	Go to MBA with results		
	else		
	{		
	Get_next_host ( )		
	If(next_host type = Untrusted or Checkpoint())		
	Send_copy_of_MA(ma_1) to RH		
	dispatch(ma_1) to next_host		
	}		
6	Receive ma_1 by th2 or uh2	th_2 or uh_2	
7	Serve (ma1)	th_2 or uh_2	ma_2

(Continued)

Table 1: Continued

#	Steps	Location	Output
8	If(ma satisfied user requirements)	th_2 or uh_2	
	Go to MBA with results		
	else		
	{		
	Get_next_host( )		
	If(next_host type = Untrut or CheckPoint())		
	Send_copy_of_MA(ma_2) to RH		
	dispatch(ma_2) to next_host		
	}		
	Repeat from 6 to 8 till man return to MAB		ma_n
K	Receive(ma_n)	MAB	
K + 1	Receive(ma_1, ma_2, . . . ma_m) from RH	MAB	
K + 2	Verify_ Integrity (m_1, m_2, . . . ma_m with ma_n)	MAB	
K + 3	Submit_results to the user		Results
	End of Algorithm		

Fig. 4 presents CPM in the content of a mobile agent system. A user submits tasks to MAB. MAB creates a MA based on the tasks. The MA starts its journey by visiting multiple hosts. Before the MA visits UH, a version copy will be kept in RH. MA copies that are stored in RH will be used to protect the MA against capturing and integrity attacks.

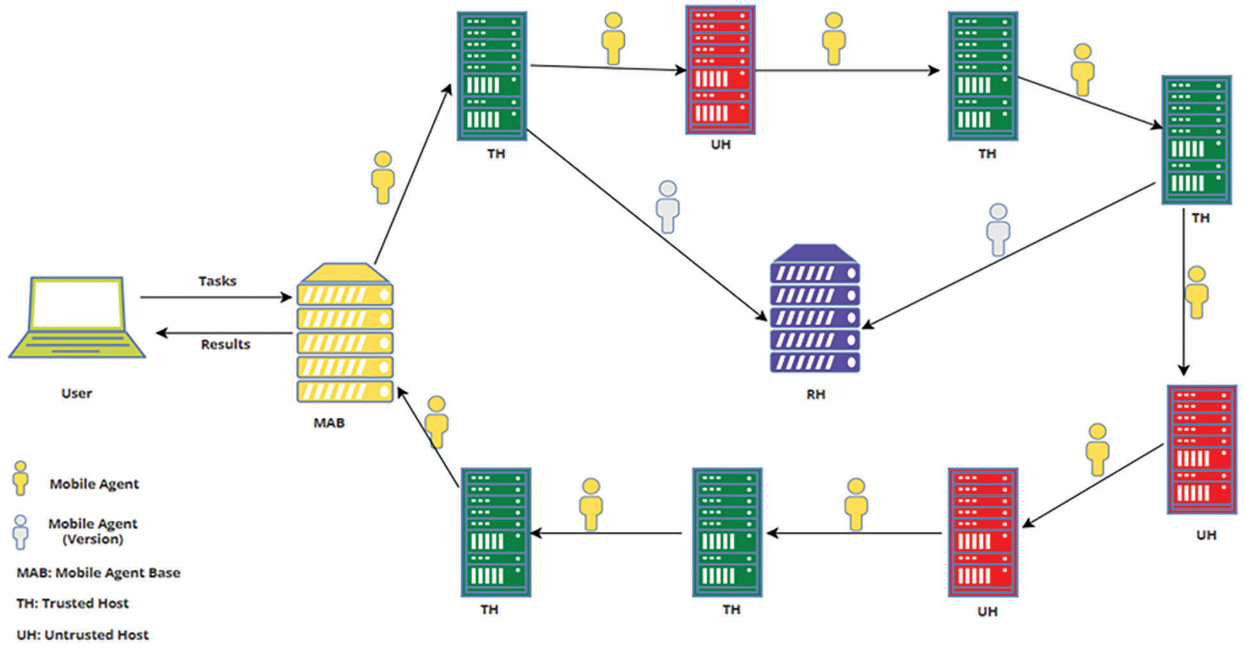


Figure 4: CPM framework

4 CPM Implementation

To measure the validity of CPM, a complete prototype of a mobile agent system has been implemented by using .Net Framework and C# language. The prototype was implemented first without CPM and after that with CPM to compare the impact of the performance. It consists of four main components:

1. MA class is used to generate MA objects. MA is serialized object that can move among hosts to perform users' tasks.
2. MBA Class is used to create MAs and provide them with the necessary information to accomplish their journeys. It has two interfaces: first to dispatch MAs and the second to receive MAs after finishing their duties. In addition, it communicates with RH to monitor MAs.
3. Host Class is used to define hosts (TH or UH). It uses to serve MAs according to MAs requirements. Hosts have two types trusted (TH) or untrusted(UH). TH sends a MA object copy to RH.
4. Recovery Host Class is used to keep MAs versions during their journeys. It coordinates with MAB

The prototype of the proposed mechanism has been implemented to prove the validity of the model. Different scenarios had been conducted by using the prototype. The scenarios were based on CPM and without CPM to measure the impact of CPM on performance.

4.1 Scenario Implementation

In this scenario, a user wants to obtain a book price from five hosts in order to buy the cheapest one. The required book maybe not be available in some hosts, so, a MA will visit an unknown number of hosts (freer-roaming mobility was used) to achieve this task. The hosts may be trusted or untrusted. The scenarios were implemented without CPM and with CPM as follows

4.1.1 First Scenario Without CPM

A MA was visited 11 hosts to get a book price. The MA target was to obtain 5 prices from different hosts. Tab. 2 presents the costs time by each host.

Table 2: Without CPM scenario

Time(ms)	Host	Host type	Service availability
5000	1	Trusted	Available
6001	2	Trusted	Not Available
5501	3	Untrusted	Available
8501	4	Trusted	Not Available
6001	5	Trusted	Not Available
5001	6	Untrusted	Not Available
6501	7	Trusted	Available
5500	8	Trusted	Available
5000	9	Untrusted	Not Available
8500	10	Trusted	Not Available
4501	11	Trusted	Available
7200	MA returned home with 5 results		

4.1.2 Second Scenario with CPM

A MA was visited 13 hosts to get a book price. The MA target was to obtain 5 prices from different hosts. Tab. 3 presents the costs time by each host. In addition, different MA versions were kept in RH to be used as recovery copies and for verification against untrusted hosts.

Table 3: With CPM scenario

Time(ms)	Host	Host type	Service availability	RH (Recovery Hosts)
6010	1	Trusted	Available	MA0
5003	2	Untrusted	Not Available	...
7020	3	Trusted	Available	...
9010	4	Trusted	Not Available	MA1
5050	5	Untrusted	Not Available	...
6001	6	Untrusted	Not Available	...
7030	7	Trusted	Available	MA2 (Checkpoint)
6089	8	Trusted	Not Available	MA3
5001	9	Untrusted	Not Available	...
6501	10	Trusted	Available	...
5500	11	Trusted	Not Available	...
8500	12	Trusted	Not Available	MA4(Checkpoint)
4501	13	Trusted	Available	...
6222	MA returned home with 5 results			...

4.1.3 Prototype Servicing Multiple MAs with CPM

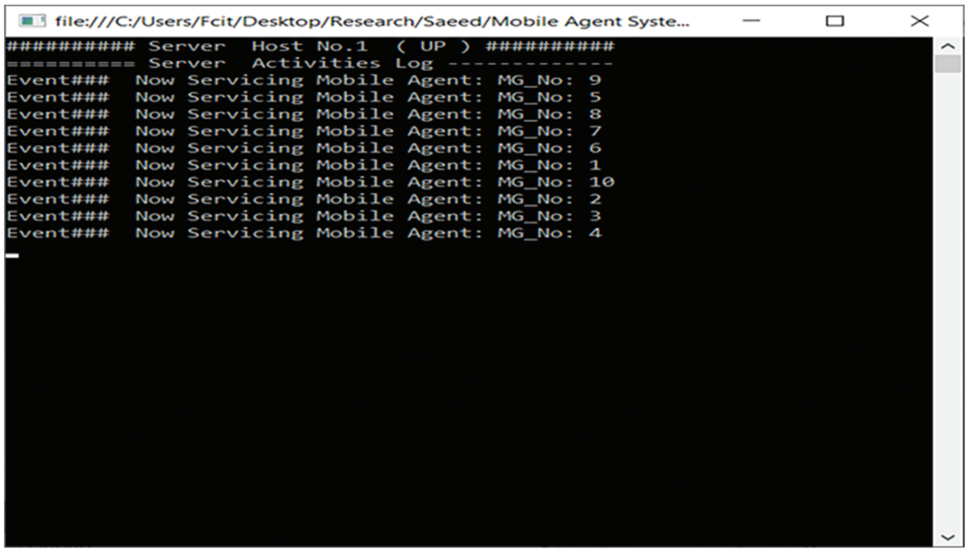
Sample screenshots are presented from the prototype implementation. Fig. 5 shows a screenshot of Host 1 that was visited by 10 MAs. Host 1 provides the MAs with the same services simultaneously.

Fig. 6 shows MAB receives multiple MAs after they finished their journeys. Each MA visited different hosts.

5 Result Discussion

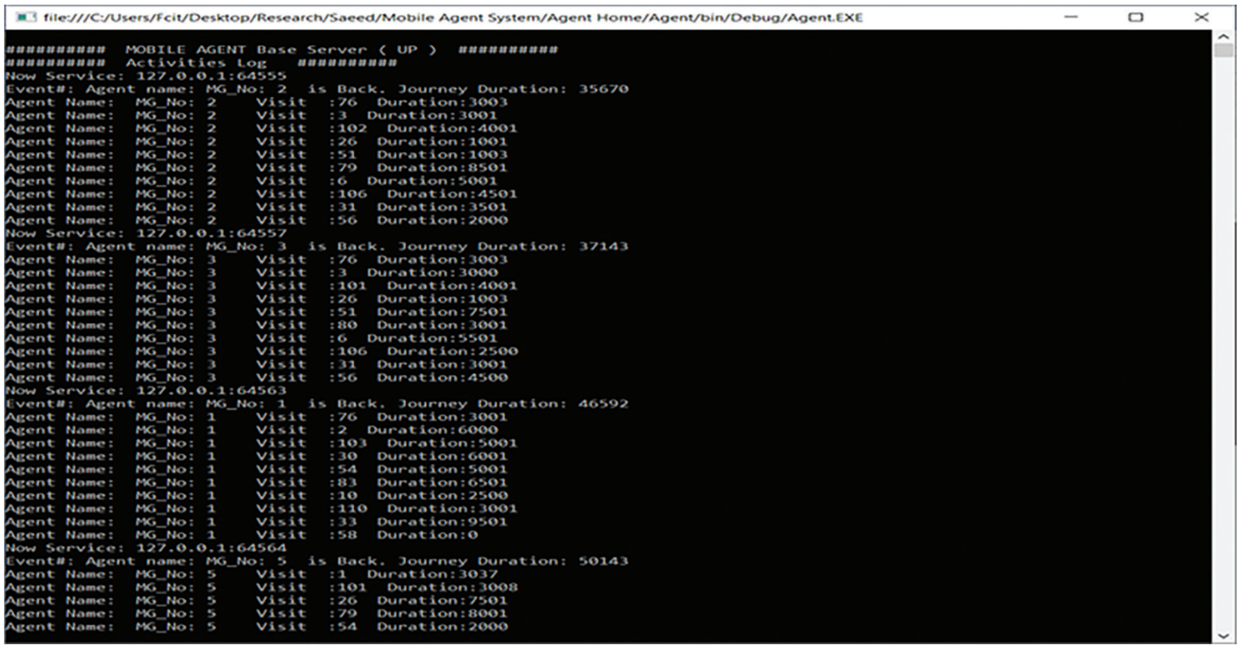
Based on several scenarios that were conducted to measure the feasibility and the performance of CPM, valuable results had been obtained. The scenarios were based on using CPM and without using CPM. First, the MA system was developed without CPM. In this scenario, the MA visited 11 hosts to ask about a book price. During the journey, the MA visited 3 untrusted hosts. So, there was a big chance that the MA may attack by those hosts either by attacking its integrity or preventing it to continue its journey. Tab. 1 presented the cost times of the journey.

The second scenario was used CPM with the same task of the first scenario. In this scenario, the MA had visited 13 hosts and 4 of them were untrusted. CPM created different versions of the MA and kept them in RH. These versions were used as recovery copies in case the MA faced attacks from the 4 untrusted hosts. In addition, some versions were created based on checkpoints assigned by the MAB. Tab. 2 explains the recovery matrix during its journey for the second scenario.



```
file:///C:/Users/Fcit/Desktop/Research/Saeed/Mobile Agent Syste...
########## Server  Host No.1   ( UP ) ##########
========== Server  Activities Log -------------
Event###  Now Servicing Mobile Agent: MG_No: 9
Event###  Now Servicing Mobile Agent: MG_No: 5
Event###  Now Servicing Mobile Agent: MG_No: 8
Event###  Now Servicing Mobile Agent: MG_No: 7
Event###  Now Servicing Mobile Agent: MG_No: 6
Event###  Now Servicing Mobile Agent: MG_No: 1
Event###  Now Servicing Mobile Agent: MG_No: 10
Event###  Now Servicing Mobile Agent: MG_No: 2
Event###  Now Servicing Mobile Agent: MG_No: 3
Event###  Now Servicing Mobile Agent: MG_No: 4
```

Figure 5: 10 MAs visit host No.1



```
file:///C:/Users/Fcit/Desktop/Research/Saeed/Mobile Agent System/Agent Home/Agent/bin/Debug/Agent.EXE
##########  MOBILE AGENT Base Server ( UP )  ##########
##########  Activities Log   ##########
Now Service: 127.0.0.1:64555
Event#: Agent name: MG_No: 2  is Back. Journey Duration: 35670
Agent Name:  MG_No: 2    Visit  :76  Duration:3003
Agent Name:  MG_No: 2    Visit  :3  Duration:3001
Agent Name:  MG_No: 2    Visit  :102  Duration:4001
Agent Name:  MG_No: 2    Visit  :26  Duration:1001
Agent Name:  MG_No: 2    Visit  :51  Duration:1003
Agent Name:  MG_No: 2    Visit  :79  Duration:8501
Agent Name:  MG_No: 2    Visit  :6  Duration:5001
Agent Name:  MG_No: 2    Visit  :106  Duration:4501
Agent Name:  MG_No: 2    Visit  :31  Duration:3501
Agent Name:  MG_No: 2    Visit  :56  Duration:2000
Now Service: 127.0.0.1:64557
Event#: Agent name: MG_No: 3  is Back. Journey Duration: 37143
Agent Name:  MG_No: 3    Visit  :76  Duration:3003
Agent Name:  MG_No: 3    Visit  :3  Duration:3000
Agent Name:  MG_No: 3    Visit  :101  Duration:4001
Agent Name:  MG_No: 3    Visit  :26  Duration:1003
Agent Name:  MG_No: 3    Visit  :51  Duration:7501
Agent Name:  MG_No: 3    Visit  :80  Duration:3001
Agent Name:  MG_No: 3    Visit  :6  Duration:5501
Agent Name:  MG_No: 3    Visit  :106  Duration:2500
Agent Name:  MG_No: 3    Visit  :31  Duration:3001
Agent Name:  MG_No: 3    Visit  :56  Duration:4500
Now Service: 127.0.0.1:64563
Event#: Agent name: MG_No: 1  is Back. Journey Duration: 46592
Agent Name:  MG_No: 1    Visit  :76  Duration:3001
Agent Name:  MG_No: 1    Visit  :2  Duration:6000
Agent Name:  MG_No: 1    Visit  :103  Duration:5001
Agent Name:  MG_No: 1    Visit  :30  Duration:6001
Agent Name:  MG_No: 1    Visit  :54  Duration:5001
Agent Name:  MG_No: 1    Visit  :83  Duration:6501
Agent Name:  MG_No: 1    Visit  :10  Duration:2500
Agent Name:  MG_No: 1    Visit  :110  Duration:3001
Agent Name:  MG_No: 1    Visit  :33  Duration:9501
Agent Name:  MG_No: 1    Visit  :58  Duration:0
Now Service: 127.0.0.1:64564
Event#: Agent name: MG_No: 5  is Back. Journey Duration: 50143
Agent Name:  MG_No: 5    Visit  :1  Duration:3037
Agent Name:  MG_No: 5    Visit  :101  Duration:3008
Agent Name:  MG_No: 5    Visit  :26  Duration:7501
Agent Name:  MG_No: 5    Visit  :79  Duration:8001
Agent Name:  MG_No: 5    Visit  :54  Duration:2000
```

Figure 6: Mobile agents return home after visiting multiple hosts

Tab. 4 presents the recovery matrix against untrusted hosts in case of two security attacks that may face MA based on the second scenario. Based on CPM, the items in the matrix will protect the MA against capturing and integrity attacks. The second column contains recovery copies for each untrusted host in case of capturing attacks. For example, if host No. 6 prevents the MA to continue its journey, the recovery copy MA1 in RH will be used instated of the original MA. The third column contains the recovery copies of MA versions for each untrusted host in case of integrity attacks. For example, if host No. 9 attacked the MA integrity, the MA versions: M0, MA1, MA2, MA3 will be used to detect the attack and make the correction in MAB. In Addition, there are more versions were generated based on checkpoints assigned before the MA starts its journey. MA2, MA4 have represented checkpoint versions.

Table 4: Recovery matrix against untrusted hosts in CPM

Host	Host type	Recovery copies of MA versions for capturing attacks	MA recovery copies
2	Untrusted	MA0	MA0
5	Untrusted	MA1	M0, MA1
6	Untrusted	MA1	M0, MA1
9	Untrusted	MA3	M0, MA1,MA2,MA3

In the third scenario, the CPM prototype can deal with multiple MAs simultaneously. Fig. 5 presents host no. 1 while it services 10 MAs. In addition, Fig. 6 presents multiple MAs who arrived home after completing their journeys.

According to the time costs that were appeared in Tab. 1 without CPM in the first scenario and Tab. 2 with CPM in the second scenario, the impact of CPM could be analyzed. Fig. 7 presents the cost times in the first and second scenarios.

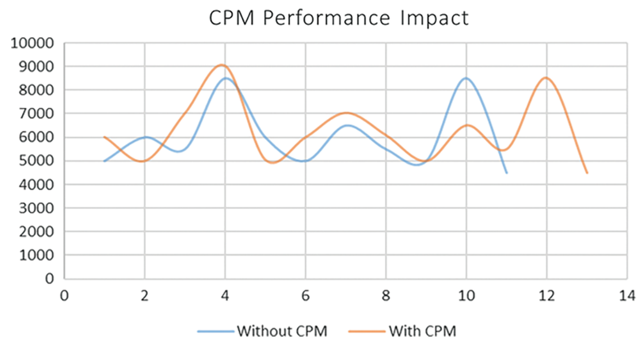


Figure 7: Cost time for CPM and without CPM

According to the cost time analysis in two scenarios, there this no significant impact on performance when using CPM. The average cost time in the case of using CPM is 6247.3 MS and without CPM the average cost time is 6000.6. The impact of CPM performance will appear in the case of untrusted hosted capturing the MA. In this situation, the MA will continue the journey by using the recovery version in RH. The cost time (P) of the complete journey could be determined by using the following equation:

$$P = \sum_{i=0}^{n+1} T_{i,i+1} \quad + \sum_{j=1}^{m} R_j$$

n: No. of hosts that are visited by MA including MAB (From and To).

T_(i−1, i) : cost of time that allows MA to move from host i−1 to i

Rj: Time that is required by MA to complete its task inside the host j.

m: number of hosts (trusted or untrusted) visited by the MA.

According to this discussion, the feasibility of CPM is approved and it is able to protect MAs against malicious hosts in two aspects: Capturing and Integrity attacks. In Addition, in terms of performance, CPM does not affect the overall performance negatively.

6 Conclusion and Future Work

Mobile Agent Technology is a promising area that allows computers to communicate with many benefits. This technology is not fully utilized because there are many security concerns. This paper has proposed CPM as a mechanism to protect MAs during their journey when they use a free-roaming mobility approach. The main idea behind CPM is to generate multiple versions of MAs to be considered as checkpoints of executions. These versions will be used to protect MAs from capturing and integrity attacks by untrusted hosts. CPM was implemented using .Net framework and C# as part of the full mobile agent system which was developed for this purpose. According to the results discussed above, CPM proved its feasibility to protect MAs against capturing and integrity attacks. In addition, there is no negatively significant difference when using CPM. Moreover, CPM can be used as a fault-tolerance mechanism in case of any normal failure which may happen to MAs. By using multiple versions that are kept in RH, MAs could be recovered.

As future plan, CPM could work with any mobile agent system which is using a free-roaming mechanism. To make this possible, CPM needs some interfaces to interact with several mobile agent systems

Acknowledgement: This project was funded by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under Grant No. D-152-611-1441. The author, therefore, gratefully acknowledge DSR technical and financial support.

Funding Statement: This project was funded by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under Grant No. D-152-611-1441. https://dsr.kau.edu.sa/Default-305-AR.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Omicini, "Towards a notion of agent coordination context," in *Process Coordination and Ubiquitous Computing*, United States: CRC Press, pp. 187–200, 2020.

[2] T. Leppänen, "Resource-oriented mobile agent and software framework for the Internet of Things," *Dissertation, Acta Universitatis Ouluensis*, C Technica, pp. 35–60, 2018.

[3] E. Fissaoui, M. Beni-hssane, A. Ouhmad and E. Makkaoui, "A survey on mobile agent itinerary planning for information fusion in wireless sensor networks," *Archives of Computational Methods in Engineering*, vol. 28, no. 3, pp. 1323–1334, 2021.

[4] D. Salil, S. Kanhere and R. Jurdak, "Multi-agent systems: A survey," *IEEE Access*, vol. 6, pp. 28573–28593, 2018.

[5] G. Fortino, F. Messina, D. Rosaci, G. Sarné and C. Savaglio, "A Trust-based team formation framework for mobile intelligence in smart factories," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6133–6142, 2020 .

[6] M. Mozaffari, B. Safarinejadian and M. Shasadeghi, “A novel mobile agent-based distributed evidential expectation maximization algorithm for uncertain sensor networks,” *Transactions of the Institute of Measurement and Control*, vol. 43, no. 7, pp. 1609–1619, 2021.

[7] T. Filgueiras, M. Rodrigues, L. de Oliveira, M. de Souza and V. Netto, “RT-JADE, A preemptive real-time scheduling middleware for mobile agents,” *Concurrency and Computation: Practice and Experience*, vol. 31, no. 13, pp. e5061, 2019.

[8] M. Kaur and S. Saxena, “A review of security techniques for mobile agents,” in *2017 Int. Conf. on Computing, Communication and Automation (ICCCA)*, India, pp. 807–812, 2017.

[9] C. Shanmuganathan and K. Boopalan, “Efficient detection of location based routing attack by routing attack mitigation in mobile agent systems,” *Annals of the Romanian Society for Cell Biology*, vol. 25, pp. 673–681, 2021.

[10] B. Schneider, “Towards fault-tolerant and secure agentry,” in *Lectures on Distributed Algorithms*, *Marios Mavronicolas and Philippas Tsigas* (Eds.), Lecture Notes in Computer Science. 1320. Springer, Berlin, pp. 1–14, 1997.

[11] A. Bryce, “A security framework for a mobile agent system,” in *Lectures on Computer Security- ESORICS*, Lecture Notes in Computer Science. 1895. Springer, Berlin, pp. 273–290, 2000.

[12] M. Karim, “Security for mobile agents and platforms: Securing the code and protecting its integrity,” *J. Inf. Technol. Softw. Eng*, vol. 8, pp. 1–3, 2018.

[13] A. Adegbite, J. Emuoyibofarhe, A. Ajala and A. Awokola, “A cybersecurity approach for evaluating mobile agents,” *Journal of Applied Security Research*, vol. 12, no. 2, pp. 253–259, 2017.

[14] E. Moussaid and M. El Azhari, “Enhance the security properties and information flow control,” *International Journal of Electronic Business*, vol. 15, no. 3, pp. 249–274, 2020.

[15] A. Acharya, H. Prasad, V. Kumar, I. Gupta and K. Singh, “Host platform security and mobile agent classification: A systematic study,” in *Computer Networks and Inventive Communication, Lecture Notes on Data Engineering and Communications Technologies,* Springer Singapore Technologies, pp. 1001–1010, 2021.

[16] F. Abdel-Fattah, F. F. Kh, F. H. Al-Tarawneh and A. M. Al-Naimat, “Dynamic intrusion detection technique for dynamic mobile ad hoc network,” *ICIC Express Letters, Part B: Applications*, vol. 10, no. 9, pp. 813–821, 2019.

[17] S. Hanaoui, Y. Berguig and J. Laassiri, “On the security communication and migration in mobile agent systems,” in *Int. Conf. on Advanced Intelligent Systems for Sustainable Development*, Morocco, Springer, Cham, pp. 302–313, 2018.

[18] P. Bagga and R. Hans, “Mobile agents system security: A systematic survey,*” ACM Computing Surveys*, vol. 50, no. 5, pp. 1–45, 2017.

[19] U. Kumar and S. Gambhir, “Device fingerprint and mobile agent based authentication technique in wireless networks,” *Int J Fut Gen Comm Netw*, vol. 11, no. 3, pp. 33–48, 2018.

[20] B. Alluhaybi, M. S. Alrahhal, A. Alzhrani and V. Thayananthan, “A survey: Agent-based software technology under the eyes of cyber security, security controls, attacks and challenges,” *(IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 10, no. 8, pp. 1–20, 2019.

[21] V. M. Roth, “Protection of co-operating agents,” in *Lectures on Secure Internet Programming*, Jan Vitek and Christian D. Jensen (Eds.), Lecture Notes in Computer Science, Springer, Berlin, pp. 275–285, 1999.

[22] B. S. YEE, “A sanctuary for mobile agents,” in *Secure Internet Programming*, Springer, Berlin, Heidelberg, pp. 261–273, 1999.

[23] G. Vigna, “Protecting mobile agents through tracing,” in *Proceedings of the 3rd ECOOP Workshop on Mobile Object Systems*, Finland, pp. 1–14, 1997.

[24] O. Esparza, M. Soriano, J. L. Muñoz and J. Forné, “A protocol for detecting malicious hosts based on limiting the execution time of mobile agents,” in *Proc. of the Eighth IEEE Symposium on Computers and Communications, ISCC 2003*, Turkey, pp. 251–256, 2003.

[25] O. Esparza, M. Soriano, J. L. Muñoz and J. Forné, “Punishing manipulation attacks in mobile agent systems,” in *IEEE Global Telecommunications Conf., GLOBECOM’04*, USA, vol. 4, pp. 2235–2239, 2004 .

[26] T. Jean, M. Adda and B. Aziz, “New computing model for securing mobile agents in IP networks,” in *2nd Int. Conf. on Internet of Things, Big Data and Security*, Portugal, SciTePress, pp. 232–238, 2017.

[27] P. Marikkannu, R. Murugesan and T. Purusothaman, “AFDB security protocol against colluded truncation attack in free roaming mobile agent environment,” in *Proc. of the Int. Conf. on Recent Trends in Information Technology (ICRTIT'11)*, IEEE, Chennai, Tamil Nadu, pp. 240–244, 2011.

[28] T. M. Ahmed. “Using secure-image mechanism to protect mobile agent against malicious hosts,” *Int. J. Comput. Electr. Auto. Control Info. Eng*, 3, vol. 11, pp. 439–444, 2009.

[29] A. Muñoz, A. Maña and D. Serrano, “Protecting agents from malicious hosts using TPM,” *Int. J. Comput. Sci*, vol. 6, no. 5, pp. 30–58, 2009.

[30] P. Dadhich, K. Dutta and M. C. Govil, “On the approach of combining trust and security for securing mobile agents: Trust enhanced security,” in *2nd Int. Conf. on Computer and Communication Technology (ICCCT-2011)*, Allahabad, pp. 379–384, 2011.

[31] J. Audun, “A logic for uncertain probabilities,” *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 3, pp. 279–311, 2001.