Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 4, Issue 3, pp: (19-32), Month: September - December 2017, Available at: www.noveltyjournals.com

Secure Enhanced Mobile Agent Security Architecture for Distributed Heterogeneous Network Environment

Omoghenemuko, Greg I.[1], Asagba, Prince O.[2], Ogheneovo, Eward E.[3]

[1]Department of Computer Science, College of Education, Warri, Delta State, Nigeria
[2,3]Department of Computer Science, University of Port Harcourt, Rivers State Nigeria

Abstract: **Mobile agent paradigmatics has become driving force for recent advances in distributed systems. The advent of mobile agency raises major security problems. The authors identified five fundamental cryptographic limitations of mobile agents: integrity; availability; confidentiality; authentication; and non-repudiation. Methodologies employed in this research that form basis for design of Secure Enhanced Mobile Agent Security Architecture (SEMASA) to advance on existing security mechanism in Marikkannu et al.(2011) are: literature review, object orientation and experimentation. Our proposed architecture employs code obfuscation, cryptography, third party trusted, digital signature, proxy signer and trip marker to achieve mobile code integrity and computation results' confidentiality. We developed Secure Employees' Attendance Mobile Agent (SEAMA) to implement our proposed security architecture using visual C-Sharp (C#) and .NET framework in Microsoft's visual studio. SEAMA was installed on one of three networked computers, and SEAMA was launched to travel from its computer with its data to compute employees' attendance work history on the two other computers having Microsoft SQL servers and malicious server administrator was programmed to carryout malicious attacks on SEAMA for experimentation, observation and ratiocination. It was discovered that our proposed architecture is immune to most attacks whereas, existing protocol of Marikkannu et al. (2011) could only detect and partially prevent some attacks but not immune to attacks like integrity and confidentiality attacks. Comparative computation shows 22.73% improvement of our scheme over existing protocol. Indubitably, our proposed security scheme can successfully detect and prevent malicious host attacks more than the existing protocol. We recommend that new security techniques could be added in our architecture to improve efficiency of the entire framework.**

Keywords: **malicious hosts; multifarious network; integrity, confidentiality and symmetric cryptography.**

1. BACKGROUND

The technology of mobile agency is precocious and very fascinating. Concept of mobile agency is unique for its capability of translocating data and code to a location on a distributed multifarious network environment. Elmarie (2004) says that the idea of mobile agent brings the computation to the data and the 'mobility' and 'autonomy' attributes make permanent connections unnecessary. The mobility of code and autonomy is historic reality and actuality hinged on Internet. Code mobility and autonomy are ultimate expressions of distributive computing in a multifarious environment. This new computer networking paradigmatics is a veritable tool for global competitiveness and it has made Internet not only popular, interesting but very welcome development. The merit, therefore, of using the mobile code technology is that, interaction cost for the mobile code owner is decreased remarkably since the mobile code after leaving its owner it transports from one host platform to another autonomously, but the blanket of insecurity hovering over mobile agents'

paradigmatics, computer, information and data transmission across multifarious network include: breach of privacy, integrity and confidentiality threats posed by malignant Web sites or host platforms on Internet. This insecurity has made Internet users' to welcome this great scientific breakthrough with mixed feelings and trepidation. Marikkannu; Adri and Purusothaman (2011) say, the mobility of code facilitates performance of tasks effectively. Mobile agents are highly efficient in usage in distributed environments' where data from various systems are required to finish a given task. This also poses threat over the system entirely since the mobile code may be prone to malicious attacks.

2. MOBILE AGENT STRUCTURE

Mobile agent structure contains: *Mobile agent; code; data;* and *execution state* (ie. *procedure+state*). Communication is salient in mobile agency. This interaction involves two main processes namely, *requests* and *responses* and these exist amongst the client (*mobile agent's owner*); *mobile agent itself* and server computer (*host platform*) in distributed multifarious network environment

Mobile Agents' Security Challenges:

Threats against mobile agents involve the protection from remote host, other mobile agents and entities outside mobile agent like entities that attack agents' transport systems. These attacks are not surmounted easily because traditional protection mechanisms were developed to overcome threats stemming from mobile agents against execution environment. Mobile agent threats during execution occur in trusted and untrustworthy environments.

3. TAXONOMY OF THREATS TO MOBILE AGENT PARADIGMATICS

Jansen (2000) identified four categories of threats to the general acceptability of mobile computing paradigmatics. They include:

- Threats imposed by mobile agent to the host;
- Threats imposed by mobile agent's Host to Mobile agent;
- Threats imposed by mobile agent to mobile agents; and
- Threats imposed by other entities to mobile agents

In designing enhanced mobile agent security architecture, we organise the possible threats from malicious hosts' platforms on mobile agents using different criteria depending on the attack. According to International Security Organisation-ISO (1988) cited in Elmarie and Elsabe (2002), the criteria by which a mobile agent is protected against a malicious host, depends on five basic requirements' of users of Internet services (namely:

- *Integrity;*
- *availability,*
- *Confidentiality,*
- *Authentication;* and
- *Non-repudiation.*

By using these basic requirements' for security, the criteria that should exist in design of mobile systems, are: integrity, confidentiality and authentication.

3.1 Integrity:

Integrity of mobile agents should be defended from manipulations by malicious hosts. This includes protection from manipulations of mobile agent's code, state, and data (Elmarie, 2004). According to Elmarie and Elsabe (2002), to protect mobile agents' from malicious hosts, integrity sub-categories should be included in the security design:

Integrity interference: mobile agent should avoid executing hosts that tends to interfere with mobile agent's execution mission. In this scenario the hosts do tamper any information, but interferes with the execution of mobile agent completely, transmitting the agent to hosts unspecified on itinerary, or executing agents' precipitately.

Information modification: The sub-criteria include several actions', namely: alteration, corrupting, manipulating, deleting, wrong execution of agent's code, and data. Second instance of information modification occurs when executing hosts interfere with interaction between agents' and alters the communications' for its own well-being.

3.2 Availability:

When mobile agent reaches host, it receives privileges and access to resources' required for its design objectives. If authorised mobile agent is denied access to objects or resources it should have legal access, then availability-refusal occurs. *Availability refusal* is deliberate action perpetrated by executing platforms, to obstruct agent. And it takes various forms' namely: *denial-of-service*, *delay-of-service* and *transmission refusal*

Denial-of-Service (DoS): Normally, this attack occurs where network crashes because it is overburdened with network traffic. Mobile agent's denial-of-service, means the requested resources' needed by agent to finish its mission are denied. It is practicable for malicious host to attack agent with much useless information that agent finds it difficult to finish its tasks and achieve goals. Attacks relating to non-reputation, where the agent platform denies that it has received an agent is also included here.

Delay-of-Service: This attack occurs where host allows mobile agent to long for service and only provides that service or access to required resources after some time. This delay can have effect on the actual purpose of mobile agent.

Transmission refusal: when malicious host disregards itinerary of mobile agent and denies agent access to next host stated in its itinerary, transmission refusal occurs.

3.3 Confidentiality:

When the assets of mobile are illegitimately accessed by host, privacy of mobile agent is not respected and agent comes under attack. *Three subclasses of confidentiality attac*ks are described, namely eavesdropping, theft, and reserve engineering.

Eavesdropping is incursion of privacy that occurs when host spies on agent and collects information about mobile agent or about intercommunications between agents. The access by remote host to mobile code and data, gives host a chance to monitor agents for other purposes than protecting itself and its own resources. Although the host may not attempt to alter the agent, it can use this information for its own benefits (Elmarie, 2004; Parul and Sharma, 2012).

The classical threat of eavesdropping when electronically communicating, is more serious in mobile agent systems because an agent platform can, not only monitor communications, but attack all unencrypted code executed agent brings to host's platform and data gathered on the platform. An agent may be exposing proprietary algorithms, trade secrets or other sensitive information (Jansen and Karygiannis, 2000). Even if the platform finds it clumsy to automatically extract the secret information, it infers the meaning from the types of services requested. Though the exact details of communications' are unknown to the platform, the communication could show that the person on whose behalf the agent acts, is planning an itinerary in future. The platform may sell this information to a suitcase manufacturer who may sends unrequested advertisements, or for worse, the platform may share this information with thieves who may target the home of the traveler (Jansen and Karygiannis, 2000).

Identity stealing or theft and eavesdroppings are related. In this subclass, the malicious hosts spy on agent, and remove information from agent. The malicious host may also steal the agent itself and use it for its own purposes, or simply kill it (Elmarie, 2004) and (Parul and Sharma, 2012).

Reverse engineering occurs when the malicious host captures mobile agent and analyzes its data and state in order of manipulates future or existing agents. While reverse engineering attacks, make host to design its own corresponding agents or update the profile of information access by agent

3.4 Authentication:

In the case of the malicious host problem, the agent correctly determines and authenticates its executing host. If agent hides identity or refuses to present its credentials, the host may hamper or jeopardize the intended objective(s) of mobile agent. *There are two subdivisions of authentication attacks*, namely *masquerading* and *cloning* (Elmarie, 2004).

Masquerading: if remote host hides as destination on mobile agent's itinerary, then, masquerading occurs. A remote host may masquerade as third party that is trusted and accept mobile agents'; to withdraw information from agents. Remote

host that masquerades can harm both visiting mobile agents' and host whose identity it has used (Elmarie and Elsabe, 2002). Masquerading agent may pose as authorised agent in an effort to gain access to services and resources to which it is not entitled. Agent may masquerade as an unauthorised agent to shift the blame for any actions for which it does not want to be held liable (Parul and Sharma, 2012).

Cloning: each agent carries its own credential to gain authorised permission to service of its executing hosts. If a host duplicate of mobile agent, it causes specific agent authentication problems.

4. PREVIOUS RESEARCH EFFORT

There are many researches that have been proposed to protect mobile agents from malicious hosts' platforms attacks. In this section, we analyse few research efforts and inferences made of them are presented. Guan *et al*. (2000) presented mobile agent security model called Police Office Model (POM), which uses hosts called police offices within defined regions. These police offices use (concept of police stations) in real world scenario. The idea of POM is to defend mobile agents' against malignant hosts' attacks, by separation of crucial components of mobile agent and allowing components to be executed at remote hosts. Although POM offers inspiring security benefits to tackle the malicious host problem, but the weakness is that the model works in closed environment and has no provision for hosts and entity trusted within a domain. Varadharajan (2000) proposed a security model that introduced security enhanced mobile agent. The security-enhanced agent has passport containing its security credentials and related security code and trusted security management component (SMC), which maintains security policy information public and private keys. The weakness of this framework is that, the mobility and autonomy attributes of mobile agent is compromised.

The FILIGRANE project is to develop security framework for mobile code for electronic commerce (Elmarie, 2004). FILIGRANE employed code obfuscation, which makes modification of the code difficult for reverse engineering). A drawback to this framework is the use of trusted hardware in the form of smart cards, which can trade mobile code on Internet.

Luo (2001) cited in Elmarie (2004) proposed a framework called Secure and Automatic Wrapper for Mobile agents (SAWMA), which uses detection approach for mobile agent's protection using three stages' techniques, namely: secret spreading, obfuscation, and Java watermarking. Before mobile agents' migration to a host, mobile agent uses wrapper to convert clear text of mobile agent cryptographically. Combining watermarking, code obfuscation and time techniques forms the basis of SAWMA and using these techniques mobile agents' become difficult to attack by malignant host platform. Sharing of secrets is by attaching time limits to the lifetime of the agent and using distributed agents, to share the secrets amongst agents. The drawback is the creation of the distributed agents for secret sharing, increase communications sessions and thus requires added computation costs.

Lee and Dick (2000) developed the Self-protecting Mobile Agent, which employs the use of distributed agents and code obfuscation and time techniques for protecting the mobile agent from malicious hosts. The disadvantages of this framework are the actual creation of the different agents and the additional communication sessions required due to additional agents. The additional agents influence computations' costs of remote hosts if executed on remote hosts. An *et al*. (2002) cited in Elmarie (2004) proposed plaintext algorithm, a method that makes code of mobile agent to be sent in plaintext, while the data and state are encrypted. The protocol relies on host to handle public key encryptions and decryptions (Jonathan *et al*., 2009). The chief drawback of the plaintext algorithm, is that it protects mobile agent against integrity and authentication attacks, but not against confidentiality attacks (Hussain *et al*., 2013).

Henry *et al*. (2001) presented the Data Encryption Standard (DES). DES algorithm is used to encrypt the secret data of mobile agent to prevent malicious hosts' attacks and attain confidentiality. The combination of these methods in applications' prevents reverse-engineering attacks on mobile agents. According to Bayan (2016), the disadvantages include the additional computation costs needed by the creator of the agent to implement code obfuscation techniques and time techniques hamper mobility and autonomy of mobile agent.

Marikkannu *et al.* (2011), proposed an Enhanced Mobile Agent Security Protocol (EMASP), which employed trip marker, digital signature for authentication and authorisation to attain confidentiality and overcomes masquerading attack on mobile agent by malicious hosts. Advantages of this protocol include: presentation of complete state of mobile agent, used

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 4, Issue 3, pp: (19-32), Month: September - December 2017, Available at: www.noveltyjournals.com

to prove modifications' done on specific hosts, mobility and autonomy of agent is maintained. Disadvantage of this protocol is that mobile code modification is possible, which violates code integrity that is a major security requirement for successful implementation of mobile agents' security. General weaknesses of existing frameworks are: mobile code tampering is possible in most cases, which violates code integrity that is a major security requirement for successful implementation of mobile agents' protection, time techniques hamper mobility and autonomy of mobile agent.

5. PROPOSED SECURITY SOLUTION

Protecting mobile agent from malicious host platforms' attacks in a multifarious network environment is a serious issue for investigation in scientific computations. Although, it is somewhat clumsy to design a foolproof model for protecting mobile agents, we are poise to design an architecture that incorporates security tactics that can comfortably protect mobile agents from malicious host platforms' attacks. Figure 1 depicts the proposed Secured Enhanced Mobile Agent Security Architecture to advance on the security mechanism in the existing work of (Marikkannu *et al.*, 2011). The components of the proposed system are:

- Agent Platform
- Trip Marker
- Proxy Signer
- Police Office
- Malicious Identification Police
- Malicious Prevention Police
- Malicious Attack Identifier
- Malicious Attack Preventer

In our proposed system architecture, we employ code obfuscation, trip marker, digital signature, third party trusted known as Certification Authority and symmetric encryption technique to improve on existing architecture. Code obfuscation, trip marker and password are used to maintain source code integrity and this makes mobile code tampering difficult for reverse engineers and malicious hosts, then third party trusted called Certification Authority for authentication and authorisation; deters eavesdroppers or masqueraders (identity theft), and cryptographic techniques ensure confidentiality of mobile agent's computation data and computation results.

Existing System Architecture:

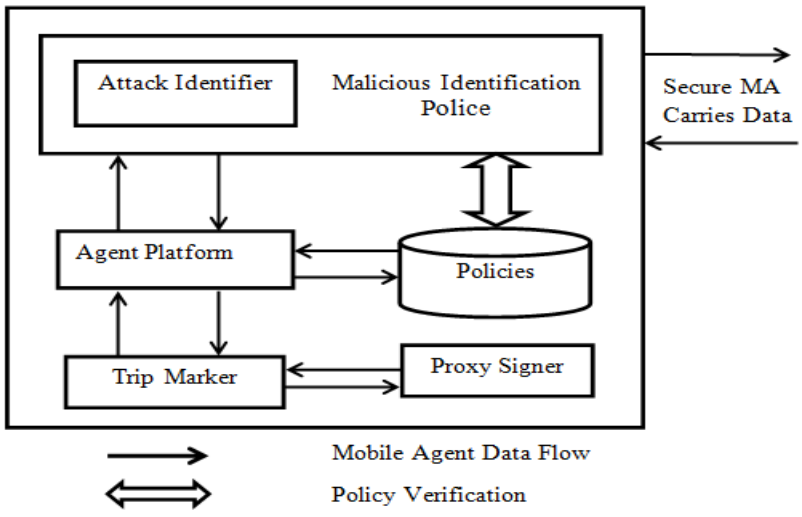

Figure 1: The existing system architecture (Source: Marikkannu *et al.*, 2011)

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 4, Issue 3, pp: (19-32), Month: September - December 2017, Available at: www.noveltyjournals.com

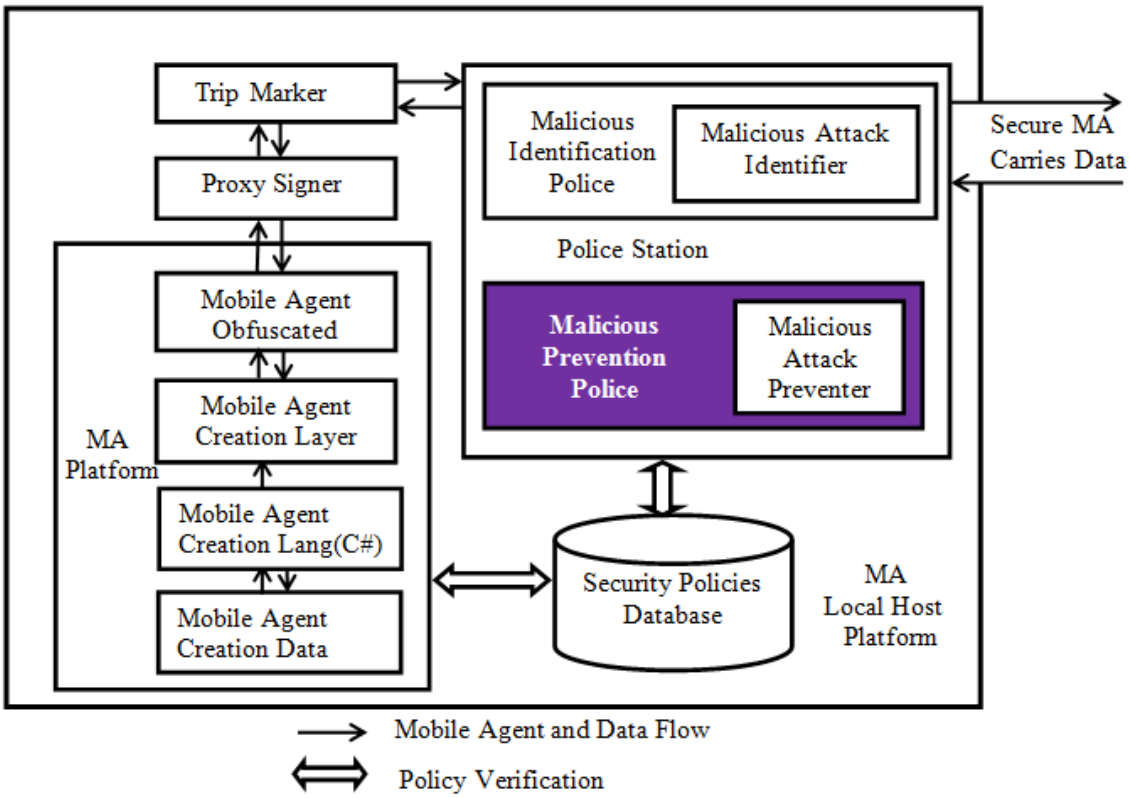

Figure 2: Proposed secure enhanced mobile agent security architecture

6. MOBILE AGENT (MA) SECURITY COMPONENTS

MA Local Host Platform: This is creator and owner of mobile agent.

Mobile agent (MA): Mobile agent is *Mobile Code*, with features of, autonomy, intelligence sociality, learning and mobility

Trip Marker: The trip marker appends mobile agent. Trip marker checks and sets expiry time for mobile agent and discriminating counter for agent. This helps to overcome external replay attacks.

Proxy Signer: Signs Mobile agents on itinerary. This is used for authentication service of the system. Proxy signer uses sequence to secure signing process.

Police Station: This is the layer or module where all mobile agent's security apparatus reside

Malicious Attack Identifier/Preventer: The malicious 'Detector'/'Preventer' helps to 'detect'/'prevent' any sort of attack on mobile agent during while on itinerary.

Malicious Identification and Prevention Police: This contains an inbuilt attack 'detector'/'preventer', which helps to detect/prevent any sort of attack on mobile while on itinerary. Moreover, it uses various policies to check if mobile agent is authorised to communicate with other agents or not. It accesses database for security Policies for mobile agents' itinerary authentication.

Security Policies Database: The policy data are maintained in the security Policy database. From policies database, decision whether mobile agent should obfuscated before transmission or not is reached. It also helps to decide whether data for computation should be encrypted or not.

Mobile Agent (MA) Creation Layer: This is mobile code creation layer- an environment where Mobile agent is developed using programming languages: like C# and Java

Mobile Agent Creation Language: C# Programming Language and .Net in Microsoft's visual studio, are employed in development of mobile agent specified in the proposed system architecture.

Mobile Agent Creation Data: These are initial input data which are part of mobile agent source code.

Table 1: Secure Mobile Agent Format of Proposed System Architecture

AID (8)	State (8)	Digital Signature or Password (16)
Expiry(8)	Counter(8)	Authorization Node (16)
Obfuscated Agent Code (Variable Length)		
Agent Code (Variable Length)		
Agent Data (Variable Length)		

The mobile agent format of the proposed system architecture contains:

1. Agent Identifier (AID)
2. State
3. Digital Signature or Password
4. Expiry
5. Counter
6. Authorization Node
7. Agent code
8. Obfuscated Agent Code
9. Data

Agent Identifier: Agent identifier is uniquely differentiate mobile agent from other agents. It is 8- bit field, which is occupied by a distinct number in the system.

State: The state field denotes agent's state. It is 8-bit field.

Digital Signature or Password: The digital signature or Password is added to the mobile agent by the proxy signer. The proxy signer use any means of signature or password. This part is for authentication and authorisation services. This field is 16-bit sized.

Expiry Timer: The expiry time is placed by trip marker which sets a time stamp which contains the maximum time limit for which the mobile agent must be alive. It is an 8 bit field.

Counter: The counter contains numbers' of itineraries mobile agent could make to its maximum. Once this filed becomes zero, the mobile agent sails to its destination or gets destroyed. The counter is 8-bit field.

Authorisation Node: The authorisation node creates the Malicious Identification Police. The node has details of the privileges' of the mobile agent, which indicates resources' to which mobile agent could gain.

Agent Code: Agent code is the actual code of mobile agent. It is the source code which is programmed to move from host to host. This field size is variable and depends on the system's design.

Obfuscated Agent Code: This is code that has been transformed to an obscure or obfuscated form complex to understand by reverse engineering activities.

Agent Data: This field is variable and it contains data the agent should convey to perform computation.

7. EXISTING DATA ENCRYPTION AND DECRYPTION ALGORITHMS PROTECTING MOBILE AGENT FROM MALICIOUS HOSTS' ATTACKS

7.1 Existing Data Encryption Algorithm Protecting Mobile Agent from Malicious Hosts' Attacks:

1. READ host_agent
2. APPEND expiry_time AND loop_counter
3. APPEND digital_signature
4. APPEND authorization_node

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 4, Issue 3, pp: (19-32), Month: September - December 2017, Available at: www.noveltyjournals.com

5. DECIDE transmit (host_data)
6. ENCRYPT AND TRANSMIT (host_data)

7.2 Existing Data Decryption Algorithm Protecting Mobile Agent from Malicious Hosts' Attacks:

1. READ remote_agent
2. DECRYPT remote_agent
3. VERIFY authorization_node
4. SCAN remote_agent
5. IF malicious
6. DESTROY remote_agent
7. ELSE
8. VERIFY digital_signature
9. IF valid
10. VERIFY expiry_timer AND loop_counter
11. DECREMENT loop_counter
12. ELSE
13. DESTROY (remote_agent)
14. ENDIF
15. ENDIF

8. PROPOSED MOBILE AGENT ITINERARY AND SECURITY PROCESS

Immediately mobile agent is initiated and computation data is initialized, the trip marker adds expiry time and counter to the whole packet. Alongside, the proxy signer adds complex password over the mobile agent, at the time of creation contains an identity and a state. When the mobile agent and trip data and password migrate to remote host, the Malicious Identification Police (MIP) adds authorization details in order to provide privileges to mobile agent to access the resources on remote hosts' platforms. Once these steps are complete, the mobile agent is allowed to travel through the Internet. When the mobile agent enters the host, the Attack Identifier verifies whether the host is malicious or not, then if malicious, the Attack Preventer, which is part of the MIP blocks the host off from having access to mobile agent's source code and computation's results from previous hosts visited. The trip marker checks the expiry time for the time stamp value and the counter values. The counter value is decremented for each itinerary for the mobile agent. Each time the mobile agent wishes to gain a resource, authorization node is examined to verify if the mobile agent has privilege to access the particular resource.

8.1 Proposed Algorithm for Mobile Agent Security from Malicious Hosts' Attacks:

01 Mobile_agent Owner

02 Create Mobile_agent with Security Policies

03 Set Expiry Time = 8 for the Created Mobile_agent

04 Obfuscate (or Sand Box) Mobile_agent Code

05 Encrypt Mobile_agent Data for Computation

06 Proxy Signer Verifies Mobile_agent Code is Obfuscated

07 Proxy Signer Verifies Mobile_agent Data for Computation are Encrypted

08 Mobile_agent Malicious Identification/Prevention Police (MIPP) containing

09 Attack Identifier and Preventer further Confirms Mobile_agent is Secure to

10 Embark on Itinerary AND to Detect/Prevent Malicious Host Attacks

11 **IF** Mobile_agent Code is Obfuscated/Computation Data Encrypted **THEN 12**

12 Trip_Marker Marks Mobile_agent to embark on Itinerary

13 Certification Authority Issues Digital Signature or Password to Mobile_agent

14 for Authentication and Authorization to be TRANSMITTED to Remote hosts

15 **IF** Mobile_agent Owner enters Incorrect Digital Signature or password **THEN 01**

16 Mobile_agent Continues on Itinerary and Arrives Host

17 **IF** Host Requests to see Mobile_agent Computation Data **THEN 18**

18 Mobile_agent Requests Host **A** to Enters Host's Private Key for Author.

19 **IF** Private Key supplied by Host is Incorrect **THEN 20**

20 Mobile_ agent Refuses to Decrypt Computation Data

21 MA Malicious Identification/Prevention Police (MIPP) flag a Message:

22 Masquerader or Malicious Host Suspected

23 **IF** Private Key supplied by Host **A** is Correct **THEN 24**

24 Host Decrypts Mobile_agent to Begin Computation

25 Mobile_agent Encrypts Computation's Result at the end of Computation

26 Mobile_agent Visits another Host-Remote Host **B**

27 Mobile_agent Visits another Host-Remote Host **C** …to Host **N**

28 Returns Home with Secure Computation's results

28 END IF

29 END IF

30 END IF

31 END IF

32 END IF

8.2 Proposed Mobile Agent Code Obfuscation Algorithm:

1. Mobile Agent Developed in C#, .Net containing MSIL Disassembler called Dotfuscator implemented in Microsoft Visual Basic IDE.
2. Programmer Clicks Mobile Code Security Button in .Net implemented in Microsoft Visual Studio IDE for type of Mobile Source Code Security needed
3. Programmer Clicks Code Obfuscation Option
4. Code Obfuscation demand is transferred to Dotfuscator contained in .Net implemented in Microsoft Visual Studio IDE
5. Dotfuscator Button Displays Available Code Obfuscation Options
6. Programmers Chooses Name Obfuscation Option
7. Name Obfuscation is implemented on Mobile Source Code Using Dotfuscator
8. Securely Obfuscated Mobile Agent is produced

8.3 Secure Mobile Agents Computation Data and Results Encryption and Decryption Algorithm:

We employed Tschindin and Sander (1998) symmetric encryption algorithm to ensure computation data and results privacy in an untrusted host or environment. It also explains a method by which a mobile agent remotely signs a document without disclosing the user's private key. **K** has an algorithm to compute a function f. **J** has an input x and is willing to compute f(x) for him, but **K** wants **J** to learn nothing substantial about f. Moreover, **J** should not need to interact with **K** during the computation of f(x). We can differentiate between a function and the program that implement it. Functions can be encrypted such that their transformation can again be implemented as programs. The resulting program will consist of plaintext instructions that a processor understands. But the processor will not understand the program's function. A protocol for non-interactive computing with encrypted functions is given below.

1. If f is a Function
2. Transform f into E(f)
3. Let P(f) denote the program which implements the function f
4. **K** encrypts f.
5. **K** creates a program P(E(f)) which implements E(f).
6. **K** sends P(E(f)) to **J**.
7. **J** executes P(E(f)) at x.
8. **J** sends P(E(f))(x) to **K**
9. **K** decrypt P(E(f))(x) and obtain f(x).

9. EXPERIMENTATION

The Secure Employees' Attendance Mobile Agent (SEAMA) developed from our proposed Secure Enhanced Mobile Agent Security Architecture (SEMASA), was installed on one of three networked computers, and the SEAMA was launched to travel from its computer with its computation data to compute employees' attendance work history on the other two computers having Microsoft SQL servers. The Microsoft SQL servers (hosts) computers, in this research, represented SEAMA's computation's environment. A malicious Microsoft SQL server administrator was programmed to unleash different types of attacks on SEAMA. The attacks include: trying to spy SEAMA source code in other to study code structure and code functions, reconfigure SEAMA's mission; attempt to spy or utter computation's data and result by trying to enter mobile agent's private key to decrypt encrypted result data. Twenty five attempts were, however, made for experimentation, observation and inference but at every attempt recorded from 1 to 25 attempts, the malicious server administrator could not succeed in any attempt because the secure employees' attendance mobile agent's source code was obfuscated and computations' data and results were symmetrically encrypted by a symmetric key cryptographic algorithm designed within SEAMA to dynamically encrypt computations' data and results. And the generated work history was able to traverse the Internet and successfully arrived destination without its computation's results being hacked and spied by any other host platform or other malicious mobile agents.

9.1 Experimental Results:

Data in Table 2 were extracted from Marikkannu *et al.* (2011) research in the line graph of Figure 3. Data in Table 2 are important here because they will compare with our research data since both research efforts are geared towards enhancing mobile agent security from malicious hosts' attacks. EMASP refers to the Enhanced Mobile Agent Security Protocol, PSP refers to the Proxy Signature Protocol and SPMA refers to the Self Protected Mobile Agents. A comparison between the three schemes is shown in Figure 3. The line graph in Figure 4 shows the Marikkannu *et al.* (2011) and our scheme for observation and inference, and it was found that the Secured Enhanced Mobile Agent Security Architecture (SEMASA) was immune to most types of attacks compared to the Enhanced Mobile Agent Security Protocol (EMASP), the Proxy Signature Protocol (PSP) and the Self Protected Mobile Agent (SPMA) scheme.

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 4, Issue 3, pp: (19-32), Month: September - December 2017, Available at: www.noveltyjournals.com

9.1.1 Result of Existing System:

Table 2 shows number of mobile agents introduced versus number of malicious agents attacks.

Table 2: Number of Mobile Agents Introduced Versus Number of Malicious Agents Attacks

No. of Mobile Malicious Agents Introduced	No. of Mobile Malicious Agent Attacks	No. of Malicious Agents Failure		
S/N		SPMA	PSP	EMASP
0	0	0	0	0
1	1	0	0	1
2	2	0	0	2
3	3	0	0	3
4	4	0	0	3
5	5	0	0	4
6	6	0	0	5
7	7	0	0	5
8	8	0	0	6
9	9	0	1	7
10	10	0	2	7
11	11	0	3	7
12	12	0	4	8
13	13	0	5	8
14	14	0	7	9
15	15	0	8	9
16	16	0	9	9
17	17	0	10	10
18	18	0	11	10
19	19	0	12	11
20	20	0	13	12
21	21	2	14	12
22	22	2	15	13
23	23	2	16	14
24	24	2	17	15

(Source: Marikkannu, et al., 2011) published data

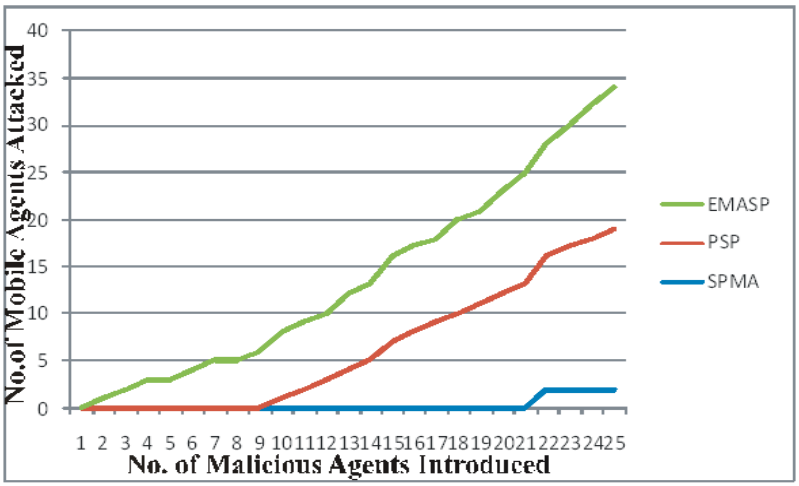

Figure 3: A comparative line graph of EMASP, PSP and SPMA showing malicious agent failure.

(Source: Marikkannu, et al., 2011) published data

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 4, Issue 3, pp: (19-32), Month: September - December 2017, Available at: www.noveltyjournals.com

9.1.2 Result of Proposed Security System:

Table 3 shows number of mobile agents introduced versus number of malicious agents failure.

Table 3: Number of Mobile Agents Introduced Versus Number of Malicious Agents Failure

No. of Mobile Agents Attacks	No. of Malicious Agents Failure in Our Security Architecture
S/N	SEMASA
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15
16	16
17	17
18	18
19	19
20	20
21	21
22	22
23	23
24	24

Table 4 shows differentials in rate of malicious agents' attacks failure in Marikkannu *et al.* (2011) security protocol and our security architecture.

Table 4: Differentials in Rate of Malicious Agents' Attacks Failure in Marikkannu *et al.* (2011) Security Protocol and Our Security Architecture

No. of Mobile Agents Attacks	No. of Malicious Agents Failure in Published Data of Marikkannu *et al.* (2011) Security Protocol			No. of Malicious Agents Failure in Our Security Architecture	Differentials in No. of Malicious Hosts Attacks Failure
S/N	SPMA	PSP	EMASP	SEMASA	
0	**0**	**0**	**0**	**0**	**0**
1	**0**	**0**	**1**	**1**	**0**
2	**0**	**0**	**2**	**2**	**0**
3	**0**	**0**	**3**	**3**	**0**
4	**0**	**0**	**3**	**4**	**1**
5	**0**	**0**	**4**	**5**	**1**
6	**0**	**0**	**5**	**6**	**1**
7	**0**	**0**	**5**	**7**	**2**

8	0	0	6	8	2
9	0	1	7	9	2
10	0	2	7	10	3
11	0	3	7	11	4
12	0	4	8	12	4
13	0	5	8	13	5
14	0	7	9	14	5
15	0	8	9	15	6
16	0	9	9	16	7
17	0	10	10	17	7
18	0	11	10	18	8
19	0	12	11	19	8
20	0	13	12	20	8
21	3	14	12	21	9
22	3	15	13	22	9
23	3	16	14	23	9
24	3	17	15	24	9

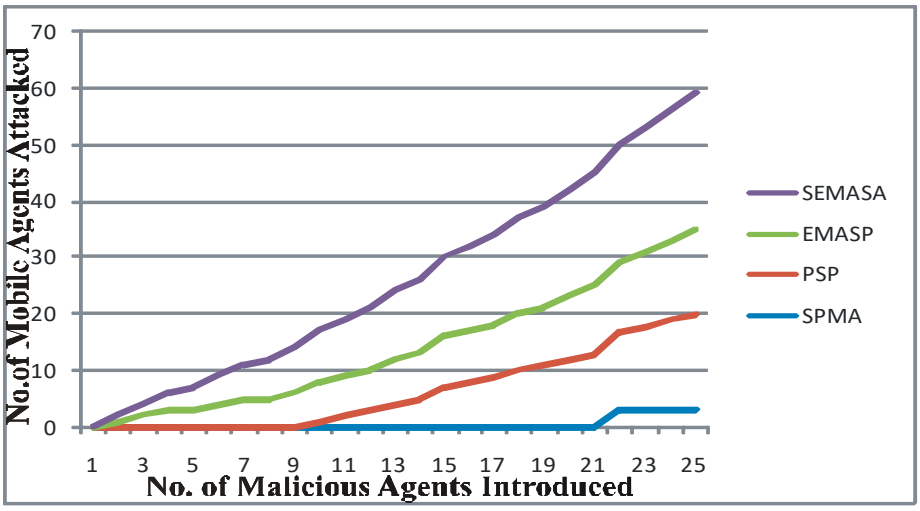

Figure 4: A comparative line graph of EMASP, PSP, SPMA and SEMASA showing malicious agents failure.

10. DISCUSSION

To measure the extent to which our security architecture has enhanced mobile agents security against malicious hosts attacks, we compared our data with the data of existing research work of Marikkannu *et al.* (2011) who also attempted to enhance the protection of mobile agents from malicious host platforms. The experimental data of Marikkannu *et al.* (2011) are presented in Table 3 and represented in the line graph in Figure 3. Whereas, the experimental data of our proposed security architecture, are presented in Table 4 and represented in the line graph in Figure 4.

And it was discovered that our proposed architecture was immune to most forms of attacks whereas, the existing security protocol of Marikkannu *et al.* (2011) could only detect and partially prevent some attacks but not immune to most forms of attacks like integrity and confidentiality attacks, which shows a progress of our scheme over existing scheme for mobile agent security from malicious hosts' attacks. Our proposed security architecture can detect and prevent malicious hosts' attacks, and then ensure code integrity and computational data and results confidentiality, which are the security requirements for implementation of successful mobile agent paradigmatics. The improvement of our scheme over existing protocol is found by the number of malicious agent failures divided by the differentials of malicious agent failure multiplied by 100%. The computation is 25/110 * 100% = 22.73%. Therefore, in clear terms, our secured enhanced mobile agent security architecture has made significant improvement on the existing security protocol of Marikkannu *et al.*(2011) by 22.73%.

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 4, Issue 3, pp: (19-32), Month: September - December 2017, Available at: www.noveltyjournals.com

11. CONCLUSION AND FUTURE WORK

Although, it is somewhat clumsy to design a foolproof model for protecting mobile code, if the security tactics incorporated in our architecture design to defend mobile agent from malicious hosts, are adopted in mobile code computing, mobile agents will definitely be protected from malicious hosts' attacks and this may bring general acceptability of mobile agents' for information gathering and dissemination, and electronic commerce involving disconnected operations. This research can be expanded in future by optimising the stages involved in the architectural security mechanism to save time to encrypt and decrypt mobile agent and its computation's data. Again, new countermeasures could be included in our security architecture to improve the efficiency of the entire system.

REFERENCES

[1] Bayan, V. (2016): Cyber Reversing. [Retrieved From: http://www.scribd.com/doc/298149737/07-Cyber-Reversing]

[2] Elmarie, B. (2004): Framework for Protecting Mobile Agent Against Malicious Host. PhD Thesis submitted to Department of Computer Science, University of South Africa

[3] Elmarie, B. And Elsabe, C.(2002): Classification of Malicious Hosts Threats in Mobile Agent Computing. Proceedings' of Annual Research Conference' of South African Institute of Computer Scientists and Information Technologists on Enablement Through Technology.[Retrieved From: http://dl.acm.org/citation.cfm?id=581526&dl=ACM&coll=DL&CFID=580514408&CFTOKEN=95723381]

[4] Guan, X., Yang, Y; and You, J.(2000): POM- Mobile Agents Security Model Against Malicious Hosts. In Proceedings' of fourth International Conference on High Performance Computing in Asis-Pacific Region.

[5] Henry, C.; Raymond, L.; Tharam, D. and Elizabeth, C.(2001): E-Commerce: Fundamentals and Applications'. [Retrieved From: http://www.scribd.com/doc/204898229/ E-Commerce-Fundamentals-and-Applications]

[6] Hussain, N; Wageeh, B. And Colin, B. (2013): Review of Medical Image Watermark Requirements' for Teleradiology: Journal of Digital Imaging, Volume 26, Issue 2. [http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3597963/]

[7] Jansen, W. and Karygiannis, T.(2000): Mobile Agent Security. National Institute of Standards' and Technology (NIST) Special Publication.

[8] Jansen, W.A.(2000): Countermeasures for Mobile Agent Security. [Retrieved From: hhttp://www.sciencedirect.com/science/article/pii/S014036640000253X]

[9] Jonathan, K.; Rafail, O. and Moti, Y. (2009): Efficient and Secure Authenticated Key Exchange Using Weak Passwords. Journal of ACM. [Retrieved From: https://www.researchgate.net/publication/237338491_Self_Protecting_Mobile_Agents_Obfuscation_Report_Final_report]

[10] Lee, B. and Dick, M. (2000): Self Protecting Mobile Agents.[Retrieved From: http://www.tolerantsystems.org/ProjectSummaries/Self_Protecting_ Mobile_Agents.html]

[11] Marikkannu, P.; Adri, J.J.; and Purusothaman, T.(2011), *An Enhanced Mobile Agent Security Protocol*. European Journal of Scientific Research, ISSN: 1450-216X,Volume 51,No.3,pp.321-331. [http://www.microsoft.com/mspress/books/3873.aspx].

[12] Parul, A. and Sharma, V.(2012), *A Review on Mobile Agent Security*. International Journal of Recent Technology and Engineering (IJRTE). ISSN: 2277-3878,Volume1, Issue 2.

[13] Tschudin, C., and Sander, T. (1998): Protecting Mobile Agents Against Malicious Hosts', Vigna G. (Ed.): Mobile Agents and Security, Springer-Verlag, pp. 44-60.

[14] Veradharadjan, V.(2000): Security Enhanced Mobile Agents. Proceedings' of 7th ACM IEEE Conference on Computer and Communications Security, November 1-4, 2000, Athen's, Greece, Pages 200 - 209