Volume: 13 Issue: 04 Pages: 5024-5028(2022) ISSN: 0975-0290

Protecting Mobile Agent against Man-In-The-Middle Attack: The Dummy Agent Model

Mohammed Suliman
Faculty of Computer Studies, Arab Open University, Saudi Arabia.
Email: msuliman@arabou.edu.sa
Bandar Alluhaybi
Faculty of Computer Studies, Arab Open University, Saudi Arabia.
Email: b.alluhaybi@arabou.edu.sa

ABSTRACT

With the fast growth of the Internet in the world, playing an important part in our daily activities and with different uses being personal or in business. The idea of the internet is based on transferring data between computers. Agent-Based Software Technology (ABST) is one of the most popular technique to exchange messages and perform the task between devices over the network. The Software Agent (SA) become vulnerable to attacks by Man-in-the-Middle (MitM) during the journey in the network. To secure the software agents against MitM attacks we propose a Dummy Agent Model (DAM) by confusing the attacker and making it difficult to differentiate the real agent with dummy agents. Dummy agents selection model is used to select dummy agents from the historical agent database for every real agent making it more difficult to attacker to identify the real agent among dummy agents. The Dummy Agent Model is proposed as a better approach to active attacks such as DoS, Collision and Alternation attacks.

Keywords – **Attack, Destination Machine, Man in the Middle, Software Agent, Source Machine.**

Date of Submission: Nov 07, 2021 Date of Acceptance: Dec 14, 2021

I. INTRODUCTION

1.1 Software Agent and its work.

A software Agent (SA) is an autonomous process that behaves like a network user program on a computer network [1]. Agent-Based Software technology (ABST) is found to be one of the significant technology used to manage and execute jobs over the network (internet). In contrast with other technologies such as remote procedure call (RPC)[2], Message Passing (MP)[3], ABST found to be better in terms of executing user programs for a high number of users and high size data, with less access latency and tuning time levels and better throughput in execution in distributed systems[1], [4]. Multiagent work more similar to multithreaded technology in the network [5]. One of the issues in the network when transmitting big data especially when it is multimedia data (audio, images and videos) can be addressed by using agents, where the SA can be shifted from Source Machine(SM) to Destination Machine (DM) to perform tasks in DM. The SA process the big data in DM and send only the results to SM as required, which significantly reduces the network overhead [6]. This is also illustrated in Figure 1.

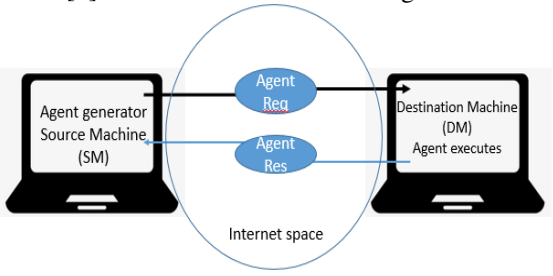

Figure 1. ABST view

Further, a software agent carries four different type of information with it, which is travel information that includes SM and DM details, code of the agent, data used by agent code, state of agent which includes results executed task and other task execution related information. Figure 2 describe the SA parts. Security of information passed between devices is always a concern in e-learning using IoT [7].

Figure 2. Software agent parts

1.2 Problem Statement

Software Agents are autonomous and are created, located in the network. These agents can move freely from machine to machine in a network. For such agents, the most important features to hold are Mobility, that agents can move from one SM to DM, doing all their job and return to home(SM); agent huskiness and portability that agents can work in a different environment with different resources and data; and agent transparency and accountability to the SM giving all the process details and results to the agent owner machine (SM) [6], [8], [9]. Software Agents are powerful with these features, but it is also vulnerable to attack in the network during their journey, to the DM, by the DM.

Volume: 13 Issue: 04 Pages: 5024-5028(2022) ISSN: 0975-0290

Some of these beforehand attack by the DM are studied, such as DoS attacks [1], Collusion attacks [10], [11], Alteration attacks [12]. Some false agents can also disturb the agent data as well as can steal the agent's data. Furthermore, the man-in-the-middle attack (MitM) is another threat to the SA as they roam from machine to machine in the network as illustrated in Figure 3, which need to address to secure the agent. Agent security needs to address without affecting the agent performance.

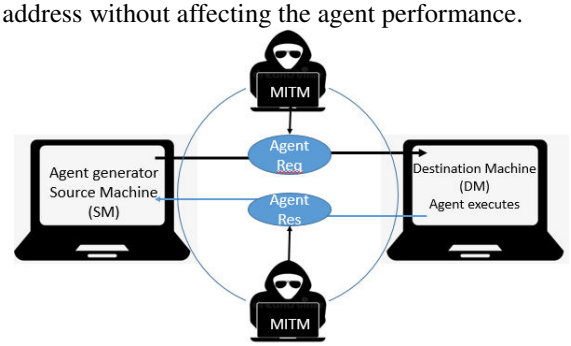

Figure 3. MitM attack

The most common MitM attack described in Table 1 involves third party (the attacker) who sits between the transceivers (SM, DM) and gain access to the communication channel to access the information and manipulate the data if intended. The transceivers are unaware any type of network attacks. There are different MitM attacks for different purposes. Further these can be categorized as SSL/TSL attacks, Spoofing-based attacks, Border Gateway Protocol (BGP) attacks and False Base Station (FBS) attacks [13]. Table 1 describes these attacks and the threats related to these attacks.

Table 1. Types of MitM attacks

MitM type of attack	Description	Threat
SSL/TSL attack	The attacker joins the communication between SM and DM, and creates separate communication channels between them and relay the messages.	Attacker can record all the messages and modify the data of the agents.
Spoofing-based attack	The attacker intercept the messages and controls such that the SM and DM has no knowledge of it.	The attacker can access the confidential data, hijack the agents and modify the data and hack the DNS system.
Border Gateway Protocol (BGP) attack	The attacker hijack IP and transfer data as authorized SM and also can manipulate the traffic	The attacker act as authorized machine and controls, manipulate the traffic leading to loss of internet parts like router.
False Base Station (FBS) attack	The attacker create a fake Base Transceiver Station (BTS) with different protocols and use the fake BTS to connect to mobile devices and block active carriers.	Fake BTS can collect the SM authentication information, cipher suits and act differently for gaining and harming the traffic

II. RELATED WORK

Many different approaches have been proposed to secure the agents which are discussed here

(B. Alluhaybi et al, 2020) proposed Dummy task selection (DTS) and improved-DTS approach, where there is a random creation of dummy tasks from the historical database of tasks. The real task agent flows from SM to DM along with the dummy task, make it unclear to the DM about the original task, thereby avoiding any malicious attack on the real task. The improved DTS is smarter in the generation of dummy tasks by creating dummy tasks with the same execution probability, a similar type of job and the same deadline to complete the task as the real task. Thereby showing more security to real tasks from the advanced attacks.

(S Srivastav and G.C. Nandi, 2014) proposed fragmentation based encryption (FBE) approach, where the mobile agent is self-protected. FBE uses encryption and decryption. The processed data at DM is encrypted and the data is ordered such that only the agent has a sense of correct order. Agent at DM uses randomize key to access the data in the correct order when returning the result to SM.

(F. Linna and L. Jun 2011) proposed security protocol against colluded truncation attack. This approach depends on recruiting a third party to migrate the agent. The entire result of the completed task at DM is encapsulated and encrypted and sent forward to the next machine, where the agent migrates to check by comparing the arrived encapsulated result and the task information to find any errors and hence detecting the attacks if any.

(L. Badger et al, 2001) proposed obfuscation technique for self-protection of the mobile agent. The obfuscation code (OC) method is to do some obfuscation transformation in the code of the mobile agent so that the attacker machine does not understand the agent code and hence preserve the agent code from the attacker DM. This guarantees that the agent code is confusing for DM and secured in mobile agent hence the agent task is secure.

(B. S. Yee, 1999) proposed a technique using the result partial encapsulation (RPE) approach. The idea is to generate a list of encryption keys (secrete key) for each DM and store it in a mobile agent. The mobile agent is instructed to use a specific key at the specific DM to perform the task and the result is encapsulated and confirms no change in the result by examining the result at

Volume: 13 Issue: 04 Pages: 5024-5028(2022) ISSN: 0975-0290

the SM, hence confirming no attack took place to the result.

III. THE PROPOSED MODEL

In this section we are presenting the Theoretical model of our proposed solution. This section is divided as firstly we present the attack model, secondly we show the dummy agent generation module and finally we present the proposed theoretical model to prevent the mobile agent from MitM attack.

In the proposed model the SM host the Agent manger that generates real agents and it is supported by a module to generate dummy agents which are related with the real agent called dummy agent generation (DAG) module. The goal of dummy agents to is to confuse the attacker, thereby protecting the real agent. Figure 4 describe the proposed model with the dummy agent generation module and the agent database where the past successfully executed agents are stored. The DAG is explained in detail in subsection B.

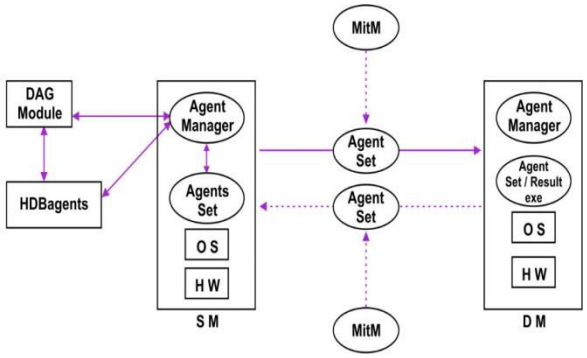

Figure 4. The proposed model

3.1 The Attack Model

Here we assume that the attacker MitM conduct the attack to gain or modify the information such as data, code, travleInfo or state of the agent and may apply perform different types attack as described in the Table 1.

The attacker target the agents in the network to gain access to the agent parts and modify the data or code or travelInfo or the state of the agent. Attacker can perform different spoofing-based attacks and SSL/TSL attacks to tamper the agents [13].

The agents can be further a normal task agent (NT) and a Time sensitive task agent (TST). A NT agent has no time limit to complete its task, whereas the TST agent has dedicated time or a time deadline to complete its task and return back result if any to home(SM)[1],[17]. TST agents will become nugatory if it does not complete the job in the given task deadline. The attacker can attack on both type of agents, looking to tamper the agent information or may be stop/delay the TST agent making the agent nugatory

3.2 Dummy Agent Generation Module

The role of Dummy Agent Generation (DAG) module is to generate dummy agents that are related with the real agent that need to be transferred to perform its task. The dummy agents are generated in such a way that the attacker cannot differentiate between dummy and real agent. To generate the dummy agents, the SM uses the DAG module and the historical agent database (HDB_{agents}) and selects appropriate dummy agents from the HDB_{agents} to join the real agent.

The HDB_{agents} include all agents that has been successfully completed their task at different DMs in past. Further the HDB_{agents} categorizes the historical agents into two categories. The first category of agent are NT agents and the second category of agents are TST agents. With this categorization, appropriate dummy agents can be selected for the real agent based on the type of real agent (such as TST real agent or NT real agent). The TST agent also records the deadline time of every agent in the database with the respective agent. Hence we define the agents in the HDB_{agents} as

$$A\text{-}DM^{i} \in \text{NT agents} \qquad i=1,2,.....n$$

$$A\text{-}DM^{i}\text{-}dl \in \text{TST agents} \qquad i=1,2,.....n;\ dl = \textit{task deadline}$$

Further, for every agent its frequency of execution at i^{th} DM can be added as

$$A_{frq}\text{-}DM^{i} \in \text{NT agents} \qquad frq = 1,2,.....,n$$

$$A_{frq}\text{-}DM^{i}\text{-}dl \in \text{TST agents} \qquad frq = 1,2,.....,n$$

And hence the HDB_{agents} is defined as

$$HDB_{agents} = (A_{frq}\text{-}DM^{i}) \cup (A_{frq}\text{-}DM^{i}\text{-}dl) \quad i=1,2,.....n \qquad (1)$$

Figure 5. illustrate the HDB_{agents} at SM.

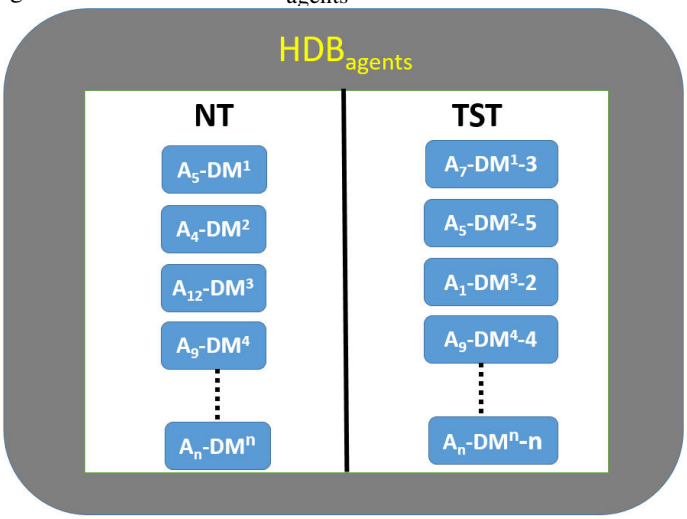

Figure 5. DB of historical agents

The NT category of HDB_{agents} in figure 5 has agents such as A_5-DM^1 that represents an agent that has successfully executed five times previously at DM^1. Similarly agent A_{12}-DM^4 represents an agent that has successfully executed twelve times previously at DM^4.

The TST category of HDB_{agents} has A_7-DM^1-3 that represents an agent that has successfully executed seven

Volume: 13 Issue: 04 Pages: 5024-5028(2022) ISSN: 0975-0290

times previously at DM^1 with task deadline of three seconds. Similarly agent A_1-DM^3-2 represents an agent that has successfully executed one time previously at DM^4 with task deadline of two seconds.

3.3 The Solution Approach

From figure 4. The proposed model, the SM uses the DAG module to have a set of randomly selected, finite number of dummy agents that suits the real agent which is ready to send to respective DM along with the real agent. The finite number of dummy agents is defined as a set of fixed number of agents selected from HDBagents, requested by the SM to combine with the real agent. Hence we can define it as

$$\{Rndagent\} \in HDBagents$$

Let $Ar\text{-}DM^i$ / $Ar\text{-}DM^i$ -dl is a real agent and Rnd_{agents} is set of randomly selected agents for a given real agent $Ar\text{-}DM^i$ from the HDB_{agents} .The Rnd_{agents} selection algorithm is given in Algorithm 1. The SM, before sending the real agent to the DM^i, makes request to DAG module to generate a finite set of dummy agents for the DM^i. The SM passes the real agent DM address (in our case it is DM number) and the required (n) number of dummy agents which are previously executed at the same DM to the DAG module. We assume that the variable n value always satisfy the equation 2. This is achieved by informing the agent manager the size of the NT and TST in HDB_{agents} before requesting for dummy agents.

$$n < size(NT(HDB_{agents})) \parallel n < size(TST(HDB_{agents})) \quad (2)$$

The DAG module prepares a random set of dummy agents based on the given input, by executing the algorithm 1 and returns the set of dummy agents { Rnd_{agents}} to the agent manager in SM.

Algorithm 1:

Input: $Ar\text{-}DM^i$/$Ar\text{-}DM^i$-dl, n; //n – number of dummy agents

Output: $\{Rnd_{agents}\}$ // A set of random dummy agents

1. $\{Rnd_{agents}\}$=null;
2. If Ar-DMi then
   a. Select NT category in HDBagents
   b. for i=1 to n
      i. Randomly select $A_{frq}\text{-}DM^i$ | $DM^i \in DM^i$
      ii. Add $A_{frq}\text{-}DM^i$ to $\{Rnd_{agents}\}$

   end for

   otherwise
   a. Select TST category in HDBagents
   b. for i=1 to n
      i. Randomly select $A_{frq}\text{-}DM^i$ -dl | $DM^i \in DM^i$
      ii. Add $A_{frq}\text{-}DM^i$ -dl to $\{Rnd_{agents}\}$

   end for
3. return $\{Rnd_{agents}\}$

The SM uses set of randomly selected dummy agents called { Rndagents} to join the real agent as a set of agents that is ready to send to the DMi. Hence we can define the complete set of agents that travel to the DMi as

$$\{CS_{agents}\} = \{Rnd_{agents} \cup Ar\text{-}DM\}^i \quad (3)$$

The SM then send the complete set of {CSagents}i to the DMi for the execution of the Ar-DMi agent at the DMi. The MitM attacker looking for the agents in the network may get confused with the number of agents travelling to DMi and may hack one of dummy agents, thereby the real agent is secure and reaches the DMi. The level of security here is based on the value of n (number of dummy agents selected) or the size of the {Rndagents}. As the value of n increases the level of security increases thereby confusing the attacker with high number of dummy agents. Figure 6 describe the level of security based on the size of {Rndagents}

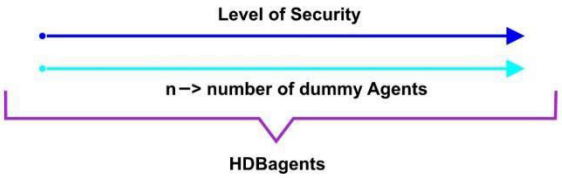

Figure 6. Level of security

IV. CONCLUSION

In conclusion, the security issues of data transfer have been one of the most critical challenges software agents and researchers face. Too many researches and solutions have been proposed to solve security issues but although these solutions have partially or completely improved the security of data transfer, on the other hand, there are attackers who also have been developing their methods and tools to break into data transforming layers. The man-in-the-middle attacks have also been improved with more advanced tools and methods. In this research, we proposed a way to confuse the middle attackers by generating a definite set of dummy agents for every real agent, this confuses attackers between the real agent and dummy agents and real data with fake data. This method helps to prevent man in the middle attacks. The selection of dummy agents for any particular real agent is based on the proposed dummy agent generation module which help in getting more similar type of definite dummy agents.

In future, this model can be implemented and tested with real data. The dummy agent module can be improved to

Volume: 13 Issue: 04 Pages: 5024-5028(2022) ISSN: 0975-0290

select more smart dummy agents from historical database. Also, generation of imaginary dummy agents can be considered in case the historical database has no agents.

ACKNOWLEDGEMENTS

The authors appreciate the support of Arab Open University in Kingdom of Saudi Arabia to complete this research work.

REFERENCES

[1] B. Alluhaybi, M. Shady, A. Alzhrani, and V. Thayananthan, "A survey: Agent-based software technology under the eyes of cybersecurity, security controls, attacks and challenges," *Int. J. Adv. Comput. Sci. Appl.*, *vol. 10*, no. 8, p. 1-21, 2019.
[2] R. J. Victorelli, *Remote procedure call management* U.S. Patent 10 p. 015-283, Jul. 3, 2018.
[3] Y. Song, A. J. Carter, J. D. Ehrlich, and S. M. Meyer, *Message passing in a distributed graph database*, U.S. Patent 9 990 443, Jun. 5, 2018.
[4] D. B. Lange and M. Oshima, *Seven good reasons for mobile agent's Commun*. ACM, vol. 42, no. 3, p. 88-89, Mar. 1999.
[5] M. Wooldridge, and N. R. Jennings *Software Engineering with Agents: Pitfalls and Pratfalls" in IEEE Internet Computing, **3*** (3), May/June 1999.
[6] B. Alluhaybi, M. S. Alrahhal, A. Alzahrani and V. Thayananthan, Dummy-Based Approach for Protecting Mobile Agents Against Malicious Destination Machines, *IEEE Access, vol. 8*, p. 129320-129337, 2020.
[7] Amira Hassan, Internet of Things (IoT) Technologies for Empowering E-Education in Digital campuses of Smart Cities, *International Journal of Advanced Networking and Applications (IJANA), Volume 11* Issue 1, p. 4142-4149, 2021.
[8] A. Caglayan and C. Harrison, *Agent Sourcebook*. Hoboken, NJ, USA: Wiley, 1997.
[9] F. Bergenti, E. Iotti, and A. Poggi, "Core features of an agent-oriented domain-speci_c language for JADE agents," in *Trends in Practical Appli-cations of Scalable Multi-Agent Systems, the PAAMS Collection*. Cham, Switzerland: Springer, p. 213-224, 2016.
[10] F. Linna and L. Jun, "A free-roaming mobile agent security protocol against colluded truncation attack," in *Proc. 2nd Int. Conf. Educ. Technol. Comput.*, *vol. 5*, Jun. 2010, p. V5-261.
[11] D. Xu, L. Harn, M. Narasimhan, and J. Luo, ``An improved free-roaming mobile agent security protocol against colluded truncation attacks," in*Proc. 30th Annu. Int. Comput. Softw. Appl. Conf. (COMPSAC)*, vol. 2, p. 309-314, Sep. 2006.
[12] B. Amro, "Mobile agent systems, recent security threats and counter measures," 2014, *arXiv:1410.4147*. [Online]. Available: http://arxiv.org/abs/1410.4147
[13] B. Bhushan, G. Sahoo and A. K. Rai, Man-in-the-middle attack in wireless and computer networking — A review, *3rd International Conference on Advances in Computing,Communication & Automation (ICACCA)* , p. 1-6, 2017.
[14] S. Srivastava and G. C. Nandi, "Fragmentation based encryption approach for self protected mobile agent," *J. King Saud Univ.-Comput. Inf. Sci.*, *vol. 26*, no. 1, p. 131-142, Jan. 2014.
[15] L. Badger, "Self-protecting mobile agents obfuscation techniques evaluation report," *Netw. Associates Lab., Rockville, MD, USA, Tech. Rep*. 01-036, 2002.
[16] B. S. Yee, "A sanctuary for mobile agents," in *Secure Internet Program- ming*. Berlin, Germany: Springer, 1999, p. 261-273.
[17] P. Bagga and R. Hans, Mobile agent system security: A systematic survey, *ACM Comput. Surv., Vol. 50,* (5), p. 65, Nov. 2017.

Biographies and Photographs

BANDAR ALLUHAYBI received the B.S. degree in computer science from King Abdulaziz University, Saudi Arabia, in 2009, the M.S. degree in engineering system management from St. Mary's University, San Antonio, TX, USA, in 2012, and the Ph.D. degree in computer science from King Abdulaziz University. He is a Lecturer at FCS, Arab Open University, KSA. His current research interests include information security, computer networks, networks security, big data, and high-performance computing.

MOHAMMED SULIMAN received the BCA and MCA degree in computer applications from Bangalore University, India, in 2004, 2007 respectively. He is a Lecturer at FCS, Arab Open University, KSA. His current research interests include information security, network security, cloud computing, big data, and software engineering.