ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 8, Issue 10, October 2021

Impacts of Mobile Agent Communication

[1] Maitanmi O. Stephen, [2]Ayorinde, Oduroye P. [3]Ajayi Adebowale O. [4]Adigun Taiwo O.
[5] Oluwatosin Ajiboye, [6]Ajayi Oluwabukola F. [7]Otuneme Nzechukwu C
[1][5][7] Babcock University, Software Engineering Department, Ilisan Remo, Ogun State, Nigeria
[2]Caleb University, Computer Science Department, Imota, Lagos State, Nigeria.
[3][6]Babcock University, Computer Science Department, Ilisan Remo, Ogun State, Nigeria.
[1] *maitanmio@babcock.edu.ng* [2]*poduroye@yahoo.co.uk*[3]*ajayia@babcock.edu.ng*[4]*adigunt@babcock.edu.ng*
[5] *ajiboyeoluw@babcock.edu.ng* [6]*otusileo@babcock.edu.ng*[7]*otunemech@babcock.edu.*

Abstract: Mobile agent technology offers a new computing paradigm in which a program in the form of a software agent can transfer its execution from agent. This agent masquerading itself as the original source of message. The use of mobile code has a long history dating back to the use of remote job entry systems in the 1960's. This articles illustrates agent incarnations which can be characterized in a number of ways ranging from simple distributed objects to highly secured software with algorithm that can only be interpreted by only the sender and the receiver. As the sophistication of mobile software has increased over time with its associated threats, This article studies masquerading as one of these threats and provides appropriate solution in the form of algorithm.

Keywords: Mobile agent, masquerading, encryption and decryption

I. INTRODUCTION

A mobile agent is a program which can migrate from one machine to another, performing useful action, under its own control. It has been the subject of much attention in the last decades due to its advantages in accessing distributed resumes in a low-bandwidth network. One of the instances where a mobile agent can be very effective is in a client/server model. In a client/server model, a client may need access to a huge database on a server. This requires a large amount of data transmission over the network and may significantly waste bandwidth if the data transferred is not useful at the client side.

In addition, one definition term `agent' means those relatively simple, client-based software applications that can assist users in performing regular tasks such as sorting e-mails or downloading Web pages from the Web, etc [1]. This class of agents is often referred to as `personal assistant' agents. At the other end of the scale is the concept of sophisticated software entities possessing artificial intelligence that autonomously travel through a network environment and make complex decisions on the user's behalf. Our definition therefore is the following: a mobile

agent is a program that acts on behalf of a user or another program and is able to migrate from host to host on a network under its own control. The agent chooses when and to where it will migrate and may interrupt its own execution and continue elsewhere on the network. The agent returns results and messages in an asynchronous fashion [2]

Alternatively, the agent may send itself to another intermediate node and take its partial results with it. Results are delivered back to the user whose address the agent knows.

Today the most common way of implementing distributed applications is through the client-server paradigm. In this model, an operation is split into two parts across a network, with the client making requests from a user machine to a server which services the requests on a large, centralized system. A protocol is agreed upon and both the client and server are programmed to implement it. A network connection is established between them and the protocol is carried out.

However the client-server paradigm breaks down under situations dealing with highly distributed problems, slow and/or poor quality network connections, and especially in the maintenance of constantly changing applications. In a system with a single central server and numerous clients, there is a problem of scalability. When multiple servers become involved, the scaling problems multiply rapidly, as each client must manage and maintain connections with multiple servers [3].

The use of two-tier systems or proxies only moves this problem to the network. It does not eliminate the basic problem. With client- server technology there comes a need for good quality network connections. First, the client needs to connect reliably to its server because only by setting up and maintaining the connection may it be authenticated and be secure. Second, the client needs to be assured of a correct response, since a server can crash anytime between processing the request and sending back the reply.

Third, it needs good bandwidth since, due to its very nature; client/server must copy data across the network. Finally, the protocol which a client and a server agree upon is by its very nature specialized and static [5]

II. PROBLEMS OF MOBILE AGENTS

Three problems were identified: problems stemming from an agent attacking an agent platform, an agent platform attacking an agent, an agent attacking another agent on the agent platform, and other entities attacking the agent system. The last category covers the cases of an agent attacking an agent on another agent platform, and of an agent platform attacking another platform, since these attacks are primarily focused on the communications capability of the platform to exploit potential vulnerabilities.

2.1 Masquerade

When an unauthorized agent claims the identity of another agent it is said to be masquerading. The masquerading agent may pose as an authorized agent in an effort to gain access to services and resources to which it is not entitled. The masquerading agent may also pose as another unauthorized agent in an effort to shift the blame for any actions for which it does not want to be held accountable. A masquerading agent may damage the trust the legitimate agent has established in an agent community and its associated reputation. Masquerading may take the following forms [9].

2.1.1 Agent-to-Platform

The agent-to-platform category represents the set of problems in which agents exploit security weaknesses of an agent platform or launch attacks against an agent platform. This set of problems includes masquerading, denial of service and unauthorized access.

2.1.2 Agent-to-Agent

The agent-to-agent category represents the set of problems in which agents exploit security weaknesses of other agents or launch attacks against other agents. This set of problems includes masquerading, unauthorized access, denial of service and repudiation. Many agent platform components are also agents themselves. These platform agents provide system-level services such as directory services and inter-platform communication services. Some agent platforms allow direct inter-platform agent-to-agent communication, while others require all incoming and outgoing messages to go through a platform communication agent [6].

2.1.3 Platform-to-Agent

The platform-to-agent category represents the set of problems in which platforms compromise the security of agents. These set of problems includes masquerading, denial of service, eavesdropping, and alteration.

III. ALGORITHM USED FOR THE ENCRYPTION AND DECRYPTION

3.1 RSA Algorithm

According to [7] Rivest, Shamir and Adleman is the most popular public key algorithm. There are two general types of key-based algorithms. Symmetric and public-key. Symmetric algorithms, sometimes called conventional algorithms are algorithms where the encryption key can be calculated from the decryption key and vice versa. In most symmetric algorithms, the encryption key and decryption key are the same. In public-key algorithm which is also called asymmetric algorithm are designed so that the key used for encryption is different from the key used for decryption. Furthermore, the decryption key cannot be calculated from the encryption key. The algorithms are called "public key" because the encryption key can be made public.

Mathematical notations

M	representing message
P	representing plaintext
C	representing ciphertext
E	encryption function
D	decryption function
H	Head
K	key
T	Tail

$E(M)= C$ the encryption function E operates on M to produce
Plaintext
$D(C)= M$ In the reverse process, the decryption function D
operates on C to produce ext message)
Cyphertext

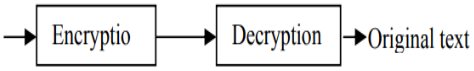

Figure 1. Encryption and Description

Since the whole point of encrypting and decrypting a message is to recover the original plain text, the following assumption must hold:

D(E(M))= M

[M, K, C, E, (.,.), D(.,.)]

E:MxK > C encryption function

D:Cx K >M decryption function

The subscript K can be introduced for the security message by both the sender and the receiver to give:

Examples of RSA Algorithm

E_k (M)= C
D_k (C)= M
D_k (E_k(M))= M

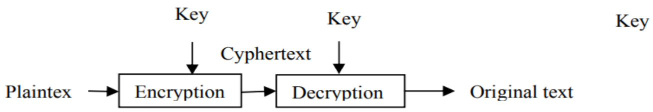

Figure. 2. Encryption and Decryption with authentication

In vegenere Cipher, the key consist of a string of K letters. These are written repeatedly below the message (from which all spaces have been removed). The message is then encrypted a letter at a time by adding the message and key letters together, working mode 26 with the letters taking values A=0, to Z=25.

For example if the key is the three letter sequence KEY then the message

M= THISISTHEMESSAGE Is encrypted using K= KEYKEYKEYKEYKEYK

To give the ciphertext

C= DPGCOQDPCESQCWEM

Using the function f(c) = (m + k) mod 26

While to decrypt back to the plaintext we use the inverse of the above function

Using Zn ≡{0,1,2,3}

F(c) = Y ≡ M+K mod 26

Y ≡ x mod n

Ymin = {x + kn}

Y = {M+K + 26q}

Such that q € Z while Y must be the smallest positive element that q can produce.

Suppose from the above analysis M=T and K= K

T = 19, K= 10 => T+ K = 19

Y = {29+ 26q} = {-23, 3, 29,...}

Y= 3

To get the inverse (M)

Y= M + K mod 26

Y= M+ K + 26q

Y-M-K= 26q

M= (Y-K) + 26q ≡ (Y-K) mod 26

M= (3-10) mod 26

-7 mod 26

M= -7 + 26q i.e (-33, -7, 19,...,)

M= 19

3.2 Software Documentation

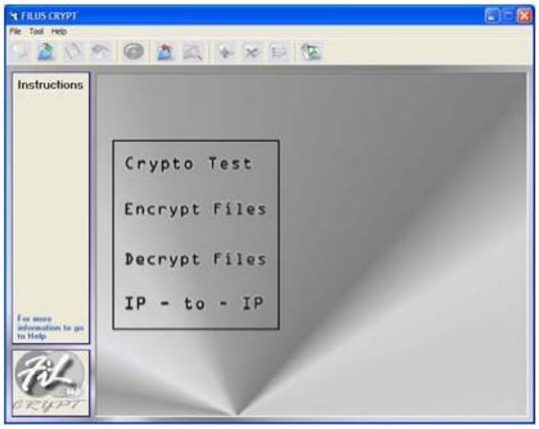

Figure. 3 Mobile Agent homepage

This software is the output of the above sample visual basic codes. This is written to encrypt your documents or files to provide the best security that your documents need so as to solve the problem of masquerading explained above.

Mobile Agent is just the name we given to the program as it moves from one computer to another. Mobile Agent is a security software application that enables you to secure store your data on your computer using strong encryption and safely communicate with other users of agent communication via Local Area Network.

Encryption

You can also encrypt several files at the same time and encrypt the listed files with a password. Files can also be stored in an archive and be encrypted at a later time if a user is not ready to encrypt the files on the list.

Decryption

Users can not encrypt without decrypting unless the file/document is no longer in use. This feature enables you to decrypt files that have already been encrypted with the ENCRYPT FILES feature. However, you can only decrypt a file at a time.

IV. ADVANTAGES OF MOBILE AGENTS

i. Disconnected operation
- Short “On-Line” times
- Low-power requirements
- Support for mobile units

ii. Low-latency interaction

V. FUTURE TRENDS

The area of mobile agent security is still in a somewhat immature state. The traditional host orientation toward security persists, and the focus of protection mechanisms within the mobile agent paradigm remains on protecting the agent platform. However, emphasis is moving toward developing techniques that are directed towards protecting the agent, a much more difficult problem. Fortunately, there are a number of applications for agents where conventional and recently introduced security techniques should prove adequate, until further progress can be made.

The next wave of security improvements for agent systems is likely to emerge from the present baseline of protection techniques, either through incremental refinements that reduce processing and storage overhead or simplify the use of the mechanism, or clever combination of complementary mechanisms to form a more effective composite protection scheme. Other peripheral topics currently neglected by researchers are also potential candidates. From the threat explained and countermeasures we reviewed earlier and in the ensuing discussion, there appears to be an opportunity for research along the following lines:

5.1 Agent Security Framework

In the past, as teams of individuals have developed agent systems, pragmatics prevailed and emphasis was placed on functionality over security. While some agent system implementations incorporate appropriate security techniques, often little regard is given to interoperability among agent systems. What is needed is an overall framework that integrates compatible techniques into an effective security model and provides an umbrella under which interoperability can exist.

The Foundation for Intelligent Physical Agents’ (FIPA) '97 and '98 standards and Object Management Group’s MASIF standards both fall short in providing the desired framework. The FIPA work is focused mainly on standardizing the agent communication language used among cooperating agents. Many of the details regarding the architecture of the agent platform require significant work before any substantive progress can be made on security. The MASIF standards on the other hand do make a clear and definitive statement on security, relying on the CORBA security services architecture. Unfortunately, although the CORBA model adequately addresses security services for an agent platform, it largely ignores any independent security services needed by an agent.

5.2 Mobile Agent Security Design Tools

Mobile agent application developers currently face a number of obstacles before they can efficiently design and develop large-scale mobile agent systems. These obstacles include: the lack of advanced development and modeling tools, the lack of mature agent standards, and the difficulty in optimizing performance under varying computational and communication loads. The limitations of agent and agent platform security mechanisms must also be overcome before agent developers can realize the full benefits of mobile agent technology. The selection of security mechanisms has a direct impact on agent migration, autonomy, disconnected operation, network latency, performance, and agent messaging. Mobile agent security design tools can help agent system developers determine the effects of employing various security mechanisms and make better decisions about functionality and performance tradeoffs.

5.3 PKI Privilege Management Extensions

Attribute certificates have long been discussed as a means of extending a public key infrastructure to allow users or other issuers to control how their authority is delegated to software that operates on their behalf. The idea is that individuals, whose identity is established through an existing PKI (e.g., PGP, MISSI, PEM, etc.), could delegate their authority by using their private key to sign a specially designed certificate, an attribute certificate. An attribute certificate contains no key material, such as the public key for an entity, but incorporates a message digest of the software along with the privilege and policy delegations. Both ANSI X9 and the IETF have made some initial attempts at standardization in this area, however, the topic has not received much attention to date from the agent community. The main area needing resolution is how to express privilege and policy within the certificate. The syntax must be able to be processed by a machine, rich enough to capture real-world privileges and policy, and simple enough for people to use. While privilege can be represented easily using a simple “privilege = attribute set” notation often employed in present day agent systems, policy is more difficult, since it must express the protection the agent must receive in conducting its activities.

VI. CONCLUSION

We have been able to successfully defined agent, mobile agent. We also went further to explain the security threats to most industries today and basic methods were implemented to solving such problems. Many of these techniques must be implemented within the framework of the agent system, while a number of them can be applied independently within the context of the application. While elementary security techniques should prove adequate for a number of agent-based applications, many applications are expected to require a more comprehensive set of mechanisms. Moreover, to meet the needs of a specific application, a flexible framework must exist in which a subset of mechanisms can be selected and applied. The trick, of course, is to select a comprehensive baseline of countermeasures which meets the philosophy of protection guiding the design of the agent system, fulfills the needs of most applications, includes compatible mechanisms, and can be extended to include other advanced mechanisms that may be invented. Clearly, this is a period where establishing such a baseline requires more experimentation and experience with alternative design choices, including those involving tradeoffs in performance, scalability, and compatibility.

REFERENCES

[1] S. Appleby and S. Steward, Mobile software agents for control in telecommunications networks. BT Technology Journal, Vol. 12. No. 2, pp.104-113. 2000.

[2] J. Baumann, Mobility in the mobile-agent-system Mole. CaberNet: 3rd Plenary Workshop 1997.

[3] S.Y Bennet, A Sanctuary for Mobile Agents. Technical Report CS97- 537, University of California in San Diego, 1997.

[4] D. Coppersmith, The Data Encryption Standard and its trength against attacks, IBM J. Res. Dev., 38, pg 243–250. 1994.

[5] C. David, G. Benjamin, H. Colin, L. David, P. Colin, and T. Gene Itinerant Agents for Mobile Computing, IEEE Personal Communications, vol. 2, no. 5, pp.34-49 2000.

[6] K. David, and S. Robert, Mobile Agents and the Future of the Internet. Department of Computer Science / Thayer School of Engineering Dartmouth College Hanover, New Hampshire 03755, 1999.

[7] H. Fritz, and G. Vinga, Eds., Time Limited Blackbox Security Protecting Mobile Agents From Malicious Host, Mobile Agents and Security, pp. 92-113, Springer-Verlag 2000.