Protecting Mobile Agents against Malicious Host Attacks Using Threat Diagnostic AND/OR Tree

Magdy Saeb[1], *Meer Hamza*[1], *Ashraf Soliman*[2]

1. Arab Academy for Science, Technology & Maritime Transport
Computer Engineering Department
2. Alexandria Stock Exchange,
Technology & Information Systems Department
Alexandria, Egypt

Abstract

Threat diagnostic, using AND/OR tree (TDT) and risk analysis, is a mechanism to protect mobile agents against malicious host attacks. The method is based on analyzing the probable causes of mobile agent failure to perform its intended function. It uses the symptoms of different types of malicious host attacks and arranges them in a logical order depending on the expected outcomes. We provide a methodical definition of attacks against mobile agents. This approach allows defining a proposed protection scheme that is used to protect mobile agents. The objectives of this paper are twofold: Firstly,, identify the different types of malicious hosts attacks based on their symptoms. Secondly we introduce our approach based on criticality analysis to identify the most critical type of attack and its associated expected failure cost.

Keywords: mobile agents, agent security, malicious host, threat diagnostic tree, and smart object class.

1. Introduction

Mobile agents are program instances that are able to migrate from one agent platform to another, thus fulfilling tasks on behalf of user or another entity. They consist of three parts: code, a data state (e.g. instance variables), and an execution state that allows them to continue their program on the next platform [1]. Mobile agent systems are intended to be use as a base for real-world applications. They transport sensitive information such as secret keys, electronic money, and other private data. Therefore security is a fundamental precondition for the acceptance of mobile agent applications. In other words, we need to have a program that actively protects itself against execution environment that possibly may divert the intended execution towards a malicious goal [2]. Many approaches aim at protecting mobile agents. There are some problems, which have to be solved before these approaches can be used [1,3,4]. Mobile agents using our proposed mechanism can be considered as a smart object class SOC. In the following sections we introduce our proposed mechanism to protect mobile agent against malicious host attacks.

2. The Predicament of malicious hosts

A malicious host can be defined in a general way as a party that is able to execute an agent that belongs to anther party and that it tries to attack that agent in some way. For example: A Mobile Travel Agent is sent out by a user to visit several airlines, find the best offer and book and pay the best flight [2]. A malicious host might spy out the price limits set by the user and the offers by competitors. It might tamper the agent to make the agent falsely believe that the host has the best offer. It might steal the mobile agent's electronic money, credit card number or cryptographic keys.

3. Threat diagnostic and/or tree (TDT)

One analytical threat derivation technique that has been designed to assist engineers during the security requirements analysis phase of computer system development is known as the threat tree approach [5]. Threat diagnostic tree approach has its origins in the use of threat trees in the mobile agent system reliability engineering, where the goal is to prevent mobile agent failures due to malicious host attacks. Our mechanism in identifying that a threat or an attack to a mobile agent has taken place is to use a threat tree analysis approach. We try to determine some symptoms for every attack class. We developed a threat tree using a relationship between the attacks and symptoms of these attacks based on the logical AND/OR relation in which attack can occur only if one the symptoms could occur. Then we can identify the attack type based on the symptoms it produces. This approach is shown in the following sections:

3.1. Protecting mobile agents from malicious hosts

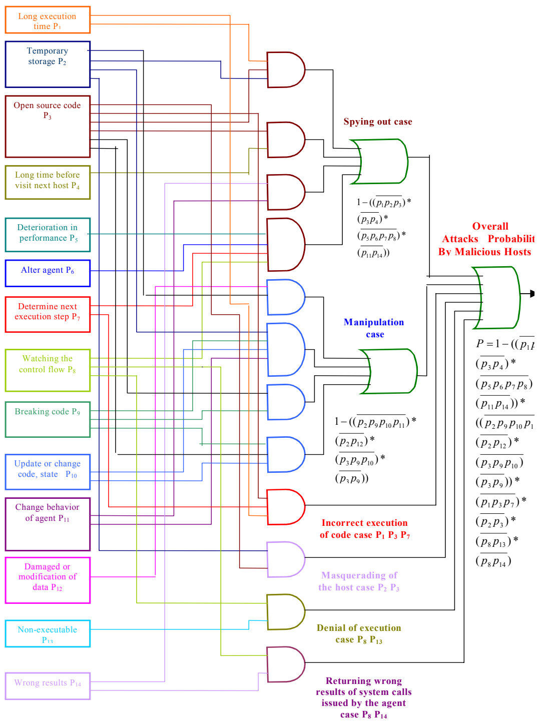
A mobile agent traverses the network with its code and data vulnerable to various types of security threats. Attacks against mobile agents are classified as active and passive attacks [6]. In a passive attack, the attacker does not interfere with the mobile agent, but only attempts to extract useful information from it. In active attacks, the attacker can arbitrarily intercept and modify code and data of the mobile agent. In the table shown below (table 1), we attempt to collect the malicious host known attacks, attack symptoms and the probability of each symptom.

Table 1: Summary of possible malicious host attacks and their symptoms

Malicious host attacks	Symptoms & Probability
Spying out code	Long execution time P_1 Temporary storage P_2 Open source code P_3
Spying out data	Open source code P_3 Long time before visit next host P_4
Spying out control flow	Deterioration in performance P_5 Alter agent P_6 Determine next execution step P_7 Watching the control flow P_8
Manipulation of code	Temporary storage P_2 Break code P_9 Update or change code, state P_{10}

	Change behavior of agent P_{11}
Manipulation of data	Temporary storage P_2 Damaged or modification of data P_{12}
Manipulation of control flow	Open source code P_3 Break code P_9 Update or change code, state P_{10}
Incorrect execution of code	Long execution time P_1 Open source code P_2 Determine next execution step P_7
Masquerading of the host	Temporary storage P_2 Open source code P_3
Denial of execution	Watching the control flow P_8 Non-executable or delay execution P_{13}
Spying out interaction with other agents	Change behavior of agent P_{11} Wrong results P_{14}
Manipulation of interaction with other agents	Open source code P_3 Break code P_9
Returning wrong results of system calls issued by the agent	Watching the control flow P_8 Wrong results P_{14}

Our objective in this work is to allow an agent to execute security-sensitive computations even in an untrusted execution environment. However, if this objective is not met due to the nature of an attack, then the agent will self-destruct.



Long execution time P_1
Temporary storage P_2
Open source code P_3
Long time before visit next host P_4
Deterioration in performance P_5
Alter agent P_6
Determine next execution step P_7
Watching the control flow P_8
Breaking code P_9
Update or change code, state P_{10}
Change behavior of agent P_{11}
Damaged or modification of data P_{12}
Non-executable P_{13}
Wrong results P_{14}
Spying out case
$1-((\overline{p_1p_2p_3})*(\overline{p_3p_4})*(\overline{p_5p_6p_7p_8})*(\overline{p_{11}p_{14}}))$
Overall Attacks Probabilit By Malicious Hosts
Manipulation case
$1-((\overline{p_2p_9p_{10}p_{11}})*(\overline{p_2p_{12}})*(\overline{p_3p_9p_{10}})(\overline{p_3p_9}))$
$P=1-((\overline{p_1p}$ $(\overline{p_3p_4})*$ $(\overline{p_5p_6p_7p_8})$ $(\overline{p_{11}p_{14}}))*$ $((\overline{p_2p_9p_{10}p_1}$ $(\overline{p_2p_{12}})*$ $(\overline{p_3p_9p_{10}})$ $(\overline{p_3p_9}))*$ $(\overline{p_1p_3p_7})*$ $(\overline{p_2p_3})*$ $(\overline{p_8p_{13}})*$ $(\overline{p_8p_{14}})$
Incorrect execution of code case P_1 P_3 P_7
Masquerading of the host case P_2 P_3
Denial of execution case P_8 P_{13}
Returning wrong results of system calls issued by the agent case P_8 P_{14}

Figure 1: symptoms for every malicious hosts attack classes

Overall attacks probability P by malicious hosts

$$P = 1 - ((\overline{p_1 p_2 p_3}) * (\overline{p_3 p_4}) * (\overline{p_5 p_6 p_7 p_8}) * (\overline{p_{11} p_{14}})) * ((\overline{p_2 p_9 p_{10} p_{11}}) * (\overline{p_2 p_{12}}) * (\overline{p_3 p_9 p_{10}})(\overline{p_3 p_9})) * (\overline{p_1 p_3 p_7}) * (\overline{p_2 p_3}) * (\overline{p_8 p_{13}}) * (\overline{p_8 p_{14}})$$

4. Measurements and risk analysis

In this section, we use two phases; the first one is the qualitative step of identifying, characterizing, and ranking hazards. The second phase is a quantitative step of risk evaluation, which includes estimating the likelihood (e.g. frequencies) and consequences of a certain hazard occurrence. This phase depends greatly on the reliability calculation of the system components, and the criticality of its constituents. [7].

4.1. Quantitative analysis

There are two major parts: Determination of the likelihood, (e.g., prob. of symptoms P_i), of an undesirable event E_i. Evaluation of the consequence, C_i of this hazardous event and the choice of the type of consequence may affect the acceptability threshold and the tolerance level for the risk. The actual expected risk value RSK is given by:

$$RSK = \Sigma P_i C_i \quad \forall \; i = 1, 2,..n$$

4.2. Criticality analysis

Criticality analysis [8] is based on normalized sensitivity analysis of the reliability expression found from the AND/OR tree. To clarify some of the terms used in kind of analysis, we discuss the following set of equations:

The overall reliability R = 1 – Probable attacks on agents by malicious hosts, i.e., R = 1-Q

Where, R = f (p_i, q_i) and p_i , q_i are the reliability and the unreliability both of them assume real values on the closed interval (0,1). Reliability function R is given by:

$$R = 1 - [1 - ((\overline{p_1 p_2 p_3}) * (\overline{p_3 p_4}) * (\overline{p_5 p_6 p_7 p_8}) * (\overline{p_{11} p_{14}})) * ((\overline{p_2 p_9 p_{10} p_{11}}) * (\overline{p_2 p_{12}}) * (\overline{p_3 p_9 p_{10}})(\overline{p_3 p_9})) * (\overline{p_1 p_3 p_7}) * (\overline{p_2 p_3}) * (\overline{p_8 p_{13}}) * (\overline{p_8 p_{14}})]$$

Components' reliabilities are allowed to take values between 0 and 1. The criticality measure of event j is given by [8]:

$$ICR_j = \frac{\partial R}{\partial p_j} / \frac{p_j}{R}$$

To compute the partial derivative, we use probability of boundary condition, where

$$\frac{\partial R}{\partial p_j} = \{R \mid p_j = 1\} - \{R \mid p_j = 0\},$$

as shown in reference [8].

4.3. Ranking of critical malcious host attacks

Preliminary experiments were carried out with a java code to create a 1000 random malicious host generator (RMH). The RMH provided six malicious host attack classes with fourteen malicious host attack symptoms. This data along with the probable attacks and reliability of mobile agents are shown in the table given below.

Table 2: The Probability of malicious hosts attack cases

	Malicious host attack cases	Q (Probable Attack)	R (Reliability)
1	Spaying	0.538	0.462

2	Manipulation	0.451	0.549
3	Masquerading	0.121	0.879
4	Denial of Execution	0.264	0.736
5	Incorrect execution of code	0.270	0.730
6	Wrong Results	0.263	0.737

In this case the criticality measure of event

$$ICR_j = \mathrm{IST}_j * \frac{p_j}{R}$$

where IST = Index of Structural Importance

Using the last equation we calculate the index of criticality malicious host attack cases. The following table (table 3) shows the ranking of criticality.

Table 3: Ranking of the critical malicious host attacks cases

Important measures		Simulation results			
Components of malicious host attacks	Malicious host attack cases	IST value	ICR value	IST Rank	ISR Rank
1	Spying	0.538	1.16	1	1
2	Manipulation	0.451	0.51	2	2
3	Masquerading	0.121	0.13	6	6
4	Denial of Execution	0.264	0.35	4	4
5	Incorrect execution of code	0.270	0.36	3	3
6	Wrong Results	0.263	0.35	5	4

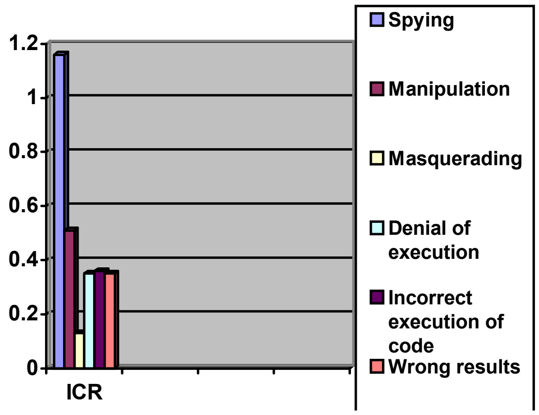


Figure 2: Ranking of the critical malicious host attack cases

5. Summary and Conclusion

Security is critical when executable code is transferred across a network. Agents themselves need protection against hostile hosts that would look for opening their code or modifying them. In this respect, we have provided the following:

- A survey of various known malicious host attacks on mobile agents,
- A detailed quantitative analysis of the given known attacks and their symptoms using an AND/OR threat diagnostic tree structure,
- Assigning probabilities of these attacks, we were able to compute an expression for the overall probability of adequate behavior of a mobile agent in a hostile environment of malicious hosts,
- To avoid the difficulty of estimating the actual values of the attack probability, we resorted to criticality analysis based on probability of boundary conditions. This type of analysis is analogous to normalized sensitivity analysis.
- While we realize that the estimation of the model parameter is not a trivial task to achieve, yet we believe the methodology discussed is a good quantitative starting point to protect agents against host attacks.

Mobile agents using our proposed mechanism can be considered as a smart object class SOC. The question that remains open to discussion is where to put this mobile agent protective measure? At this stage, we recommend to add this analysis code as part of the operating system. The advantage of this approach is the dynamic monitoring and estimation of various probabilities of attacks. However, the main disadvantage is that is can be circumvented by advanced malicious attackers. The other alternative is to have such protective measures added to the mobile agent code itself. In this case, the agent will self-destruct when an attack has taken place. The overhead encountered with this alternative approach is the main problem of applying it in all types of mobile agents. Future work, using both approaches, will shed some light on the required optimal course of action.

6. References

[1] Fritz Hohl, "A framework to protect mobile agent by using reference states", University of Stuttgart, Germany, March, 2000.
[2] Toms Sander and Christian F. Tschudin, "Protecting Mobile Agent Against Malicious Hosts", International Computer Science Institute 1947 Center Street, Berkeley, CA 94704, USA. Sander , tschudin@icsi.berkeley.edu. Springer-Verlag, pp. 92-97, 1998.
[3] Fritz Hoh1, "Time Limited Blackbox Security: Protecting Mobile Agents from Malicious Hosts", Institute of Parallel and Distributed High-performance System (IPVR), University of Stuttgart, Germany, Fritz.Hoh1@informatik.uni-stuttgart.de Springer-Verlag pp. 44-46, 1998.
[4] General Magic, The Typescript Reference Manual, http://www.genmagic.com/telescript/ documentation/TRM/,1996.
[5] Edward G. Amoroso, Fundamentals of Computer Security Technology, Prentice-Hall International, Inc., 1994.
[6] Warwick Ford, "computer communications security – principles, standard protocols and techniques", Prentice Hall, 1994.
[7] Mohamed Modarres, Mark Kaminskiy, Vasiliy Krivstov, Reliability Engineering and Risk Analysis, Marcel Dekker, Inc.,1999.
[8] M. Saeb, R. Schinzinger, D. Nyrienda, " On Measures of Network Reliability and Critical Components", IEEE. Asilomar Conference on Circuits, Systems, and Computers, Asilomar, California, 1987.