Secure Recording of Itineraries through Co-operating Agents

Volker Roth
vroth@igd.fhg.de
Fraunhofer Institut für Graphische Datenverarbeitung
Rundeturmstr. 6
64283 Darmstadt
Germany

January 12, 2003

Abstract

Security is a fundamental precondition for the acceptance of mobile agent systems. In this paper we discuss protocols to improve agent security by distributing critical data and operations on mutually supporting agents which migrate in disjunct host domains. In order to attack agents, hosts must form coalitions. Proper selection of itineraries can minimize the risk of such coalitions being formed.

Keywords: mobile agent security, malicious host, distributed applications, itinerary, agent cooperation

1 Introduction

"It is difficult to exaggerate the value and importance of security in an itinerant agent environment. While the availability of strong security features would not make itinerant agents immediately appealing, the absence of security would certainly make itinerant agents very unattractive" [2]. Unfortunately, some security issues in itinerant or mobile agent systems are hard to solve. A prominent example is the problem of *malicious hosts* which receives increasing attention. The underlying problem stems from the fact that mobile agents are executed in the *security domain* of the agent server which provides the agent's runtime environment.

The most conservative approach is to assume that each host on a mobile agent's itinerary is hostile and willing to collaborate with other malicious hosts on the agent's route. This assumption is as realistic as the assumption that hosts can be generally trusted, though. A more reasonable assumption is probably the following:

> Given a particular mobile agent, at any point in time, a certain percentage of hosts might be malicious.
>
> Not all malicious hosts are willing to collaborate with other hosts in attacking a mobile agent.

The percentage of malicious hosts likely depends on the gain which can be expected from successfully attacking mobile agents weighted against the costs of mounting the attack as well as the risk of detection and the consequences of being detected. Of course, collaboration yields more power but it also requires close coordination and increases the danger of leaks which might lead to disclosure.

2 Related Work

A number of approaches are investigated in order to protect mobile agents against malicious hosts. Many of the current proposals deal with the a–posteriori recognition of tampering and the identification of the culprit.

Vigna proposes a scheme which allows to identify a host that tampered with an agent [8] based on tracing. The protocol requires hosts to commit to *execution traces* of agents being run by it. One of the drawbacks of this protocol is the size of traces which must be stored and verified.

Meadows proposes to lay out baits called "protection objects" in order to detect tampering on mobile agents [3]. This approach is criticized by Sander and Tschudin due to a "lack of cryptographic strength" [7]. In the very same paper both authors promise to shake at the widespread belief that certain operations such as digitally signing a document off–line in a secure way cannot be performed by a mobile agent on a malicious host [7]. The proposed solution is based on function hiding via *homomorphic encryption schemes*. An in–depth analysis and discussion of this approach remains to be done, though.

Furthermore code obfuscation is investigated as a possible means to protect agents against analysis and manipulation by malicious hosts. Trusted hardware environments or *secure coprocessors* are sometimes considered as an alternative for

software–only approaches with regard to protecting agents against malicious hosts. This approach is taken for instance in the *Sanctuary project* [1].

Yee describes an approach using two agents which is illustrated using the airfare booking agent scenario which is frequently used to discuss mobile agent security related problems [1]. The protocol described requires a fixed itinerary for both agents, tolerates at most one malicious host and only "provided a solution for a special case", though. Nevertheless we believe that using cooperating agents has merit for improving the security of mobile agent applications in the face of malicious hosts.

3 White, Gray and Red Hosts

For ease of discussion we assume three categories of hosts each labeled with a color. Given a particular mobile agent, a *white* host is completely trusted. Presumably there is at most one host shining in bright white – the one of the agent's owner, since it is the one which is controlled and operated exclusively by him. Hosts which are not completely trusted and thus might become malicious are *gray*. Those hosts which might collaborate with at least one other host in an attack on the mobile agent are *red*.

White hosts are extremely rare, and demanding that a white host is available to a mobile agent would limit the flexibility of the mobile agent model. After all, the only white host might be a laptop which needs to be disconnected while the agent roams the network. The agent's world hence consists basically of gray hosts as well as a number of potential red ones.

Various hosts should be gray for most agents e. g. those of the employer's company, universities, and nonprofit organizations, in particular those which have a dedication towards privacy and security. Aggressive red hosts might attack those networks in order to turn gray hosts into involuntary red ones; however this carries considerable risk for the attacker.

The security of mobile agents against attacks by malicious hosts can be improved by distributing critical operations of a mobile agent between two cooperating agents, each of which operates in one of two disjunct nonempty sets of hosts $\mathcal{H}_a$ and $\mathcal{H}_b$ which hold the following condition $\mathcal{C}$:

> No red host of either set is willing to cooperate with a red host of the other set.

Note that red hosts in the same set might still cooperate in attacking the agent, and gray hosts might still try to attack an agent on their own.

The general idea is to do perform critical tasks such as the authorization of negotiations between a host and an agent in the other (cooperating) agent and to split or secretly share data between agents which might be stolen by a single host. In the remaining sections we illustrate the idea by giving example protocols. For those protocols to work we need to make two additional assumptions:

- hosts provide an authenticated communication channel to the two cooperating agents
- the authenticated identity of the remote host, the authenticated identity of the host the agent came from as well as the local host's identity are provided to the hosted agent

As Yee already pointed out "if an agent is running on an honest server, both these answers (for the peer identity and the local host's identity) will be correct. . ." [1]. In our case we assume that a gray host is honest with respect to providing the three identities mentioned above as long as it does not profit from lying.

4 Tracing Loose Routes

A simple yet effective attack on a mobile agent is for a malicious host not to let the agent migrate to the servers of competitors. This particularly affects mobile agents with *loose itineraries* in comparison to agents whose itineraries are defined a–priori, because deviations from a fixed itinerary are easier to spot and prove.

We would like to record the actual loose route taken by an agent without any possibility of manipulation by the hosts on its route. Providing $\mathcal{C}$, the following protocol for two cooperating agents should have the desired property. Each agent *b* records and verifies the route of its cooperating agent *a* as follows:

Definition: Let $h_i \in \mathcal{H}_a$ be the ith host being visited by agent *a* and let id(h_i) be the identity of host h_i. Let $prev_i$ be agent *a*'s idea of the identity id(h_{i-1}) of its previous hop. i shall denote the identity of the next hop agent *a* wants to take while being on host h_i. The agents' routes end at their origin, that is h_n=h_0 for a route with n hops.

Initialization: Let h_0 be the origin of agents *a* and *b*. h_0 has to be a white host with respect to *a* and *b*. For agents *a* and *b*, 0 is set to the first hop of their respective itineraries. Both agents are subsequently sent to their first hops.

Step $i, i \in \{1, \ldots, n\}$**:** Agent *a* sends the next hop i and the previous hop $prev_i$ to agent *b* over the authenticated channel; Agent *b* thus also learns id(h_i). Agent *b* verifies that id(h_i)=i-1$\wedge$$prev_i$=id($h_{i-1}$) and appends i to the stored route.

Security of the protocol: It is straightforward to see that if host h_i forwards agent *a* to a host h_{i+1} with id(h_{i+1})$\neq$i then host h_{i+1} must either successfully masquerade as the host with id i or it has to deny communication between the cooperating agents. On the other hand, if host h_{i+1} permits the communication and properly authenticates itself (in other words, h_{i+1} is honest regarding the protocol) then agent *b* discovers that host h_i sent agent *a* to the wrong destination.

If id(h_{i+1})$\neq$i then host h_{i+1} cannot put agent *a* back on its route by sending *a* to the host with identity i because agent *b* recorded id(h_i) as $prev_{i+1}$. As a matter of consequence, h_{i+1} must either be honest (identifying h_i as a cheater in the process) or h_{i+1} has to collaborate with h_i in putting the agent back on its expected route. The last case is equivalent to h_i sharing a copy of some known digital data with another party – hardly something which can be prevented or detected at all.

A malicious host h_{i+1} might incriminate a *honest* host h_i by claiming to have received agent *a* from some other host h', hence implicating that h_i sent *a* to h' instead of the host with identity i=id(h_{i+1}). The protocol is not able to decide which one of the two hosts is the culprit. However, if h_{i+1} really received agent *a* from h' then h_{i+1} should be able to produce a copy of *a* which is signed by h' given some additional agent protection mechanisms are implemented (see [4]).

The protocol has some drawbacks, though. Firstly, setting up the authenticated communication channel for each migration is a costly operation. Secondly, if agent *a* is killed, one of two hosts might be responsible and the protocol cannot decide which one. In addition to that, some host h_{i+1} might take two agents a_1 and a_2 both being received by the same host h_i and switch the recording of the route of a_1 to agent b_2 and vice versa. The protocol must be enhanced to cope with this situation.

5 Discussion and Conclusions

It must be ensured with reasonable confidence that agents *a* and *b* never enter the same set of hosts with regard to $\mathcal{C}$. Hence the approach taken in this paper should be complemented by results from research in agent routing policies such as the one indicated by Swarup [6]. Hosts which do not want to forward agents to the servers of certain competitors might still do so. However, the itinerary of the cooperating agent will always show this fact.

Providing $\mathcal{C}$, cooperating agents also provide an approach to tackle security problems of other critical operations such as payment authorization. Payment protocols such as Chaum's digital electronic cash protocol (as described e. g. in [5]) can in principle be adapted to work with two mutually cooperating agents. The protocol ensures detection of double spending with cheater identification.

The owner of the agents prepares a money order m as described in [5]. The identity strings are prepared by randomly choosing keys k_i^l, k_i^r, $i = 1, \ldots, n$ and computing

$$\{I_i = (E_{k_i^l}(I_i^l, H(I_i^l)), E_{k_i^r}(I_i^r, H(I_i^r))) \mid i = 1 \ldots n\}$$

where E is a suitable encryption function and H is a strong cryptographic one–way hash function. The money order as well as the keys are secretly shared between agents *a* and *b*. If *a* now wants to purchase goods from host h_i then *a* first sends an offer which is signed by h_i to *b*. Agent *b* verifies that the signature is valid and h_i's identity (as determined by the authenticated communication channel) is the one given in the according public key certificate. Agent *b* goes on to check whether the offer is acceptable and finally decides whether payment should be made.

If the decision is positive, *b* stores the offer, requests the host's *selector string*, transfers its share of m, and opens the selected halves of the identity strings by sending the appropriate key shares.

Host h_i reconstructs the money order m, verifies the bank's signature and makes sure that agent *b* properly opened the selected halves of the identity strings. The protocol goes on as described in [5].

Clearly, host h_i cannot manipulate the payment *decision* without cooperation from *b*'s host. Neither of the hosts involved can steal the money order without the cooperation of the other. However, any two hosts may profit from a joint attack on the cooperating agents by sharing the additional wealth gained from defrauding the agents compared to the profit gained from honest behavior. Therefor the itineraries of both agents must be chosen with great care in order to assure $\mathcal{C}$ with reasonable

confidence. Security can be improved by additional measures [4] which ensure that payment can only be made by b while being on particular hosts.

In summary, protocols based on assumption $\mathcal{C}$ have merit since they are less susceptible to attacks by coalitions of hosts than single agents. $\mathcal{C}$ is a generalization of the trusted third party concept which is less restrictive and easier to meet.

References

[1] BENNET S. YEE. A Sanctuary for Mobile Agents. DARPA Workshop on Foundations for Secure Mobile Code, Monterey, CA, USA, March 1997. Position Paper.

[2] CHESS, D., GROSOF, B., HARRISON, C., LEVINE, D., PARRIS, C., AND TSUDIK, G. Itinerant agents for mobile computing. *IEEE Personal Communications* (October 1995), 34–49.

[3] MEADOWS, C. Detecting Attacks on Mobile Agents. DARPA Workshop on Foundations for Secure Mobile Code, Monterey, CA, USA, March 1997. Position Paper.

[4] ROTH, V., AND JALALI, M. Access control and key management for mobile agents. *Computers & Graphics 22*, 3 (1998). Special issue *Data Security in Image Communication and Networks*.

[5] SCHNEIER, B. *Applied Cryptography*, 1 ed. John Wiley & Sons, Inc., 1994, section 6.7, pp. 120–122. Digital Cash Protocol #4.

[6] SWARUP, V. Trust Appraisal and Secure Routing of Mobile Agents. DARPA Workshop on Foundations for Secure Mobile Code, Monterey, CA, USA, March 1997. Position Paper.

[7] TOMAS SANDER, AND CHRISTIAN F. TSCHUDIN. Protecting Mobile Agents Against Malicious Hosts, November 1997. accepted for the forthcoming special volume "Mobile Agent Security" in the series Springer Lecture Notes in Computer Science.

[8] VIGNA, G. Protecting Mobile Agents Through Tracing. In *Proceedings of the 11th European Conference on Object-Oriented Programming (ECOOP) 1997* (Jyväskylä, Finland, June 1997).