A Secure Intelligent Trade Agent System

X. Yi, X. F. Wang, K. Y. Lam

Department of Information System and Computer Science
Faculty of Science, National University of Singapore
Lower Kent Ridge Road, SINGAPORE 119260
E-mail: {yix,wangxiao,lamky}@iscs.nus.edu.sg

Abstract. In this paper, a secure intelligent trade agent system is developed. In this system, an intelligent trade agent can be authenticated and supplied certain authorized agent execution environment for it to run by a host. The owner of a malicious intelligent trade agent is easily dug out. A host can only legally modify the information relative to it in the agent because the owner of the agent can be conscious of any little unauthorized modification made by any host. The secure intelligent trade agent system has two extra features: 1. The payment in the system is anonymous to servers (e.g., shops, companies). 2. It is convenient for the system to charge. So far, any security weakness in the secure intelligent trade agent system has not been found yet.

1 Introduction

Electronic commerce on the Internet has quickly grown in recent years in view of its low expense and high efficiency. However, the amount of business information available on the Internet is so large that it becomes nearly impossible for customers and merchants to visit each site on the networks, analyze the information and make sound business decisions as to the trade of goods or services. So far, one of the best solutions [1] is utilizing intelligent trade agents to trade for humans.

Mobile agent has the ability to migrate itself across nodes of a network in order to perform its tasks and report back its findings. The agent typically gathers and analyzes data from a multitude of nodes on the network, and presents a subset of the data as information to user, agent or system. An intelligent trade agent is an intelligent agent with the ability to visit many different sites on the networks, analyze the data at each site and make decision as to whether goods should be traded at the specific site.

The security of intelligent trade agent system can be split into two broad areas [2] [3]. The first involves the protection of host from malicious trade agents while the second involves the protection of trade agents from destructive hosts.

An intelligent trade agent system is an open system. Just like in any open system, the host nodes are subject to a variety of attacks such as leakage, tampering, resource stealing and vandalism.

In addition, because its codes is executed by a host, an intelligent trade agent has to expose its data and codes to the host environment. Therefore, a malicious host can always scan the agent for information, alter the agent states and codes, even kill the agent. Current consensus is that it is computationally impossible to protect an intelligent trade agent from a malicious host. Instead of tackling the problem from a computational (difficult) point of view, current research [4] is looking at sociological means of enforcing good host behavior.

In this paper, we develop a secure intelligent trade agent system. In this system, an intelligent trade agent can be authenticated and supplied certain authorized agent execution environment for it to run by a host. The owner of a malicious intelligent trade agent is easily dug out. A host can only legally modify the information relative to it in the agent because the owner of the agent can be conscious of any little unauthorized modification made by any host.

The rest of the paper is organized as follows: Section 2 describes the secure intelligent trade agent system. Section 3 analyzes the security of the proposed system. Section 4 states how the proposed system provides the service charge function. The paper is closed in Section 5.

2 Description of the secure intelligent trade agent system

Similar to most of the secure distributed system, the proposed secure intelligent trade agent system utilizes a public key cryptosystem (e.g., RSA[5]) and a hash function (e.g., MD4[6], MD5[7]) to encipher and decipher data, provide certificates and generate digital signature.

Throughout the following discussion, each participant X in the system has a pair of keys associated with it, one of which being publicly known (X_p, X's public key), the other one only known to X (private or secret key X_s). X's public key is used to encrypt data meant to be read by X; X can decrypt the result using its private key: $D = X_s(X_p(D))$. X can use X_s create a digital signature, which can be verified by any party using X_p. The hash value of a message M with the hash function H is denoted as $H(M)$.

In addition, we need a hierarchical structure of certification authorities to provide certificates for all participants in the system. Their public keys of these certification authorities are known to every entity. In order to simplify description here, we assume that only a certification authority (CA) is in the system. In the case of a multitude of certification authorities, the description is almost the same as that of a certification authority except that some symbols are slight different.

X's certificate ($Cert_X$) issued by the CA is composed of the certificate content ($CertCont_X$), which mainly contains serial number, encryption algorithm identifier, hash function identifier, signature algorithm identifier, CA's name, CA's public key, X's name, X's public key, validity period and so on, and the CA's signature on it.

$$Cert_X = CertCont_X || CA_s(H(CertCont_X)) \tag{1}$$

The secure intelligent trade agent system aims to satisfying the need: a user U, who may be a subscriber or a salesman, wishes an Agent Service Center (ASC) to buy or sell some goods or services in a list of servers denoted as Server $S^{(1)}$, Server $S^{(2)}$, $\cdots$, Server $S^{(n)}$. These servers may be some shops or companies. The goods or services to buy and sell in different servers may be same so as to seek for the optimal trade.

Based on the need, we proposed the secure intelligent trade agent system as follows:

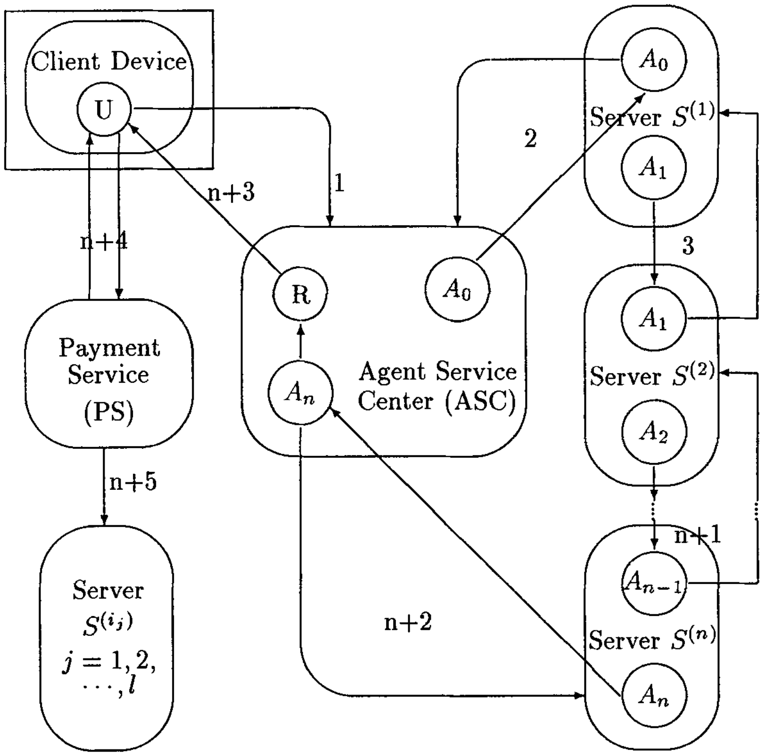

Fig.1: The proposed secure intelligent trade agent system

In order to ensure the security of the proposed intelligent trade agent system, the secure structure of the intelligent trade agent needs to be considered firstly.

2.1 Structure of an intelligent trade agent

In the proposed system, the basic structure of an intelligent trade agent sent by the ASC can be divided into three distinct portions from the point of view of security.

1. Agent Certificate — ASC's certificate ($Cert_{ASC}$) issued by a certification authority (CA). On basis of (1), we know

$$Cert_{ASC} = CertCont_{ASC} || CA_s(H(CertCont_{ASC}))$$

2. Secure Portion — Sessions $1, 2, \cdots, n$. Each session is assigned to a specified server and contains some codes and data which will be executed in a specified server. For example, Session i is assigned to Server $S^{(i)}$ and takes the form of

$$Session\ i = S_p^{(i)}(ASC_s(CD_i, t)) \tag{2}$$

where CD_i denotes the codes and data which will be executed in $S^{(i)}$ to fulfill a trading mission ($i = 1, 2, \cdots, n$). It consists of table of contents (a map to the structure of a session which expresses the size, type and importance of all components of the session), components and next node (where the agent will go).
3. Plain Portion — Static portion and Dynamic portion, which are clear to all participants in the intelligent trade agent system. Static portion includes format illustration, encryption algorithm identifier, hash function identifier, signature algorithm identifier and some possible error actions (the actions the agent should take should an error occur while the agent runs on the environment supplied by a server). Some possible error actions include discarding the agent, delivering an error notification to a specified address, routing the agent to a server. Dynamic portion includes Time Stamp (the time when the trade agent is sent out by a server), $Cert_{send}$ (the sender's certificate issued by the CA) and $Sign_{send}$ (the signature of the sender on the part (M) from Agent Certificate to Time Stamp in the trade agent).

$$Sign_{sned} = Send_s(H(M)) \tag{3}$$

The structure of an intelligent trade agent can be illustrated in the following figure:

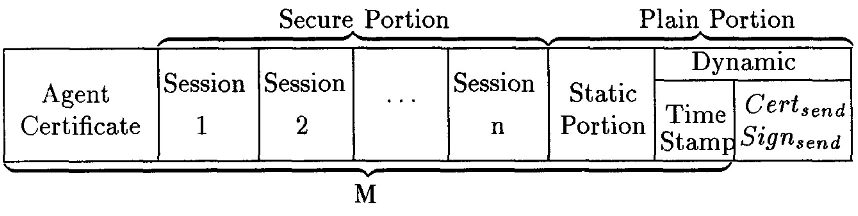

Fig.2: The structure of an intelligent trade agent

2.2 Structure of Login Data Bases (LDB) for servers

In the proposed system, each server involved in the Internet trading need a Login Data Base (LDB) saving some information from passing trade agents in order to provide the evidence that a trade agent is authorized and not illegally modified by a server when any problem occurs.

Because there are probably a lot of trade agents entering a server to trade every day, it is impossible for a server to keep whole passing trade agents in its records. Otherwise, the LDB will occupy a great deal of storage space. In view of it, the structure of record of the LDB needs to be very optimal.

Our solution to this problem is setting up the LDB in the following record structure:

Agent Certificate	Time Stamp (0)	Previous Node	Time Stamp (1)	Signa-ture (1)	Negotica-tion Result	Next Node	Time Stamp (2)	Signa-ture (2)

Fig.3: The structure of record of a Login Data Base (LDB)

In the above structure, the meaning of each field is illustrated as follows:

1. Time Stamp (0)—the time when the trade agent is generated.
2. Previous Node—the certificate of the previous server from which the agent comes.
3. Time Stamp (1)—the time when the agent is out of the previous server.
4. Signature (1)—the signature of the previous server on the trade agent.
5. Negotiation Result—the trading decision made by the intelligent trade agent and the server.
6. Next Node—the certificate of the next server to which the trade agent has gone.
7. Time Stamp (2)—the time when the agent goes to the next server from the server.
8. Signature (2)—the signature of the next server on the trade agent received by it.

2.3 Trading procedure of the proposed system

Generation of an intelligent trade agent

Step 1: User $U \Rightarrow$ Agent Service Center (ASC)

1. User U, who wishes to buy or sell some goods or service in a list of servers (Server $S^{(1)}$, Server $S^{(2)}$, $\cdots$, Server $S^{(n)}$), get access to a client device connected to an Agent Service Center (ASC). From the client device, he transmits

$$Cert_U, ASC_p(R_U, t), U_s(H(R_U, t)) \tag{4}$$

to the ASC, where R_U states the user's trading requirement (specifications of goods or services to buy or sell, a list of servers to trade, trade rules, etc.) and the current time t acts as time stamp of this communication.
2. After receiving (4) from U, the ASC checks the following items:
   (a) Is $Cert_U$ issued by the CA?
   (b) Has User U registered the ASC before?

(c) Does the signature U on (R_U, t) correct?
(d) Is the time stamp t valid?
If all answers are certain, the ASC executes 3. Otherwise, the ASC refuses the requirement of User U.
3. On basis of R_U, the ASC generates an intelligent trade agent A_0 according to the structure shown in Fig.2.
4. A_0 is activated and starts to roam the list of servers on Internet.

An intelligent trade agent roaming network

For simply description, the ASC is formally denoted as Server $S^{(0)}$ and $S^{(n+1)}$.

Step i+1: Server $S^{(i-1)} \Rightarrow$ Server $S^{(i)}, i = 1, 2, \cdots, n+1$.

1. After trade agent A_{i-1} roams into Server $S^{(i)}$, $S^{(i)}$ checks the following items:
(a) Is Agent Certification issued by the CA?
(b) Is $ASC_p(S_s^{(i)}(Session\ i))$ significant codes and data? Is t valid?
(c) Is Time Stamp of A_{i-1} valid?
(d) Is $Cert_{send}$ issued by the CA?
(e) Does Equation $S_p^{(i-1)}(Sign_{send}) = H(M)$ hold?
If all answers are certain, the $S^{(i)}$ executes 2. Otherwise, A_{i-1} is treated on basis of the possible error actions in Static Portion of A_{i-1}.
2. Server $S^{(i)}$ replies Server $S^{(i-1)}$ with
$$Cert_{S^{(i)}} || S_s^{(i)}(H(M)) \qquad (5)$$
It is saved into Server $S^{(i-1)}$'s LDB.
3. Server $S^{(i)}$ provides A_{i-1} with an agent execution environment, in which A_{i-1} automatically gathers business information from authorized resources supplied by $S^{(i)}$, analyses it, negotiates with $S^{(i)}$ and finally reaches a negotiation result (NR_i) about buying or selling some goods or services on basis of CD_i. NR_i states whether $S^{(i)}$ agrees to sell or buy some goods or services under certain conditions. If so, it should include Server $S^{(i)}$'s certificate, account number, the current time, the detail about selling or buying some goods or services (such as cost, quantity and quality, etc.). If not, it only need to include No and the current time.
4. Session i, Time Stamp, $Cert_{send}$ and $Sign_{send}$ of A_{i-1} are respectively replaced with $ASC_p(S_s^{(i)}(NR_i))$, the current time, $Cert_{S^{(i)}}$ and $Sign_{S^{(i)}}$. The variant version of A_{i-1} is denoted as A_i.
5. A_i continues to roam into Server $S^{(i+1)}$ and then replies $S^{(i)}$ with $Cert_{S^{(i+1)}} ||$ $S_s^{(i+1)}(H(M))$. Server $S^{(i)}$ opens a new record of Login Data Base (LDB) and fills it with corresponding items according to the structure of record shown in Fig.3.

Step n+3: Agent Service Center $\Rightarrow$ User U.

After the trade agent sent out by the ASC returns, the ASC recovers Session $1, 2, \cdots, n$ of it to n negotiation results $NR_1, NR_2, \cdots, NR_n$ and checks whether NR_i is meaningful and whether the time included in NR_i is valid. If so, the ASC discards those which include NO and chooses out those which are optimal about buying or selling same goods or services. The chosen results are supposed to be $NR_{i_1}, NR_{i_2}, \cdots, NR_{i_m}$, where $m \leq n$. The ASC transmits

$$R = Cert_{ASC}, U_p(S_p^{(i_1)}, S_s^{(i_1)}(NR_{i_1}); S_p^{(i_2)}, S_s^{(i_2)}(NR_{i_2}); \cdots;$$
$$S_p^{(i_m)}, S_s^{(i_m)}(NR_{i_m}); t), ASC_s(H(Contracts, t)) \quad (6)$$

to User U, where $Contracts = (NR_{i_1}, NR_{i_2}, \cdots, NR_{i_m})$ and t is the current time.

Payment and goods delivery

Step n+4: User U $\Leftrightarrow$ Payment Service (PS)

1. After receiving R from Agent Service Center, User U authenticates the ASC with $Cert_{ASC}$, recovers $Contracts$ and checks the signature of the ASC on $(Contracts, t)$ and validity of t. If no problem occurs, User U makes his final decision about this trading on basis of $Contracts$. The final decision is assumed to be buying or selling some goods or services in $S^{(i_{j_1})}, S^{(i_{j_2})}, \cdots, S^{(i_{j_l})}$, where $l \leq m$. Then User U provides Payment Service with:

$$Cert_U, PS_p(S_p^{(i_1)}, S_s^{(i_1)}(NR_{i_1}); S_p^{(i_2)}, S_s^{(i_2)}(NR_{i_2}); \cdots; S_p^{(i_l)},$$
$$S_s^{(i_l)}(NR_{i_l}); Decision, t), U_s(H(Contracts, Decision, t)) \quad (7)$$

   where Decision states User U agrees to buy or sell some goods or service in Server $S^{i_1}, S^{i_2}, \cdots, S^{i_l}$ and includes User's account number, the detail about buying or selling and the method for sending or receiving these goods or services.
2. Payment Service authenticates User U with $Cert_U$, checks the signature of User U on $(Contracts, Decision, t)$ and validity of t, checks the signature of Server S^{i_j} on NR_{i_j} $(j = 1, 2, \cdots, l)$ and makes sure whether Decision is compatible to Contracts. If without questions, the PS transfers an amount of money from User U's account to Server $S^{(i_j)}$'s account or from Server $S^{(i_j)}$ account to User U's account on basis of Contracts and Decision and then transmits:

$$Cert_{PS}, U_p(TAP_U, t), PS_s(H(Contracts, Decision, TAP_U, t)) \quad (8)$$

   to User U, where TAP_U (Transfer Account Payment) states how much money has been transferred from User U's account to Server $S^{(i_j)}$'s account or from Server $S^{(i_j)}$ account to User U's account and t is the current time, where $j = 1, 2, \cdots, l$.

Step n+5: Payment Service (PS) $\Rightarrow$ Servers $S^{(i_j)}, j = 1, 2, \cdots, l$.

Payment Service transmits

$$Cert_{PS}, S_p^{(i_j)}(NR_{i_j}, TAP_{S^{(i_j)}}, t), PS_s(H(NR_{i_j}, TAP_{S^{(i_j)}}, t)) \quad (9)$$

to Server $S^{(i_j)}$. $TAP_{S^{(i_j)}}$ states how much money has been transferred into or from Server $S^{(i_j)}$'s account due to NR_{i_j} and t is the current time, where $j = 1, 2, \cdots, l$.

Finally, both User U and Server $S^{(i_j)}$ authenticate Payment Service, recover out TAP_U and $TAP_{S^{(i_j)}}$, check the signature of PS and then send or receive some goods or services in a specified place. The trading procedure is successfully fulfilled.

3 Security analysis on the intelligent trade agent system

3.1 Security of Agent Service Center against User U

In the proposed secure intelligent agent system, User U must provide his certificate ($Cent_U$) and signature on (R_U, t) if he wishes to obtain trade agent services from Agent Service Center. An unauthorized user who intends to enter the ASC is certainly refused because he is not able to show an authorized certificate and signature in the same time. In addition, replaying attack is impossible and of no significance in view of the existence of the time stamp.

3.2 Protection of servers against malicious agents

A trade agent is unique in that its codes is executed by a server, thus an executing agent has automatic access to some of a server resources. With this level of access trade agents can mount attacks by altering other local agents, propagating viruses, worms and Trojan horses, impersonating other users and mounting denial of service attack. The standard approach to this problem is to reject all unknown codes from entry into servers. It is not a viable solution in a mobile agent environment. Both Telescript [8] and Safe-Tcl [9] offer approaches to solve the problem.

In the proposed intelligent agent system, each session of a trade agent takes the form of ciphertext shown as (2). Only Agent Service Center can generate such sessions because its secret key and the time stamp are involved in (2). Therefore, Session i provides not only the codes and data which will be executed by Server $S^{(i)}$ to fulfill a trading mission, but also a non-repudiation evidence that the ASC can not deny having generated Session i. Once any problem occurs when Server $S^{(i)}$ runs CD_i on an agent execution environment, the ASC is probably malicious and will be accused. So if the ASC is a trusted part, no malicious trade agent will be generated.

3.3 Protection of agents against malicious servers

In order for a trade agent to run, it must expose its codes and data to the host environment which supplies the means for that agent to run. The host can always scan the agent for information, alter the agent states and codes, even kill the agent. Thus, the trade agent is unprotected from the host.

Current consensus is that it is computationally impossible to protect mobile agents from malicious hosts. Instead of tackling the problem from a computational (difficult) point of view, current research [4] is looking at sociological means of enforcing good host behavior.

The proposed system provides the protection of agents against malicious servers in the following aspects:

1. In the generation of a trade agent, Agent Service Center adopts segmenting encryption so that the hidden codes and data in Session i is exposed only to Server $S^{(i)}$ because $S_p^{(i)}$ is involved. Therefore, Server $S^{(j)} (j \neq i)$ can not scan Session i to obtain the codes and data (CD_i) which will be executed by $S^{(i)}$. In addition, after reaching a negotiation result NR_i, Server $S^{(i)}$ has hidden it to $ASC_p(S_s^{(i)}(NR_i))$ which can be recovered to NR_i only by the ASC. So the other servers can not know what negotiation result is reached between the trade agent and Server $S^{(i)}$. In a word, the proposed system effectively prevent a server scanning other servers' information.
2. Although Server $S^{(i)}$ is permitted to modify a trade agent in order to put a negotiation result, Time Stamp, $Cert_{S^{(i)}}$ and $Sign_{S^{(i)}}$ in it, any malicious manipulation (such as cutting or manipulating other servers' sessions) will be detected by Agent Service Center (because the ASC will be conscious of any malicious action if the trade agent has taken a error action, or any session can not be recovered to significant negotiation result or even complete disappears). In this case, the ASC will carry out the following check procedure and dig out the malicious server.
   (a) The ASC asks all servers in this trade to commit their records (denoted as $Rec^{(i)}$ $(i = 1, 2, \cdots, n)$) about the trade agent and $S_s^{(i)}(NR_i)$ $(i = 1, 2, \cdots, n)$. The motivation to prove themselves to be innocent drives all servers except a malicious one to provide true records about the trade agent.
   (b) On basis of A_0 and $Rec^{(i)}, S_s^{(i)}(NR_i)$ $(i = 1, 2, \cdots, n)$, the ASC can reconstruct $A_1, A_2, \cdots, A_n$ in a recursive way.
   (c) Suppose Servers $S^{(i)}$ $(i = 1, 2, \cdots, m-1, m \leq n)$ have no problem, the ASC checks whether $S_p^{(m+1)}(Signature\ (2))$ is equal to $H(M)$ (where M is the part of A_m from Agent Certificate to Time Stamp and $Signature(2)$ is from $Rec^{(m)}$). If so, Server $S^{(m)}$ has no problem. If not, Server $S^{(m)}$ will be identified as a malicious one because it can not provide the non-repudiation of receipt from Server $S^{(m+1)}$ which states Server $S^{(m+1)}$ has successfully received the correct A_m. Server $S^{(m)}$ should repeatedly have transmitted A_m to Server $S^{(m+1)}$ until $S_p^{(m+1)}(Signature\ (2)) = H(M)$ or have delivered an error notification to the ASC.

3. If a trade agent is killed by a malicious server, the above check procedure 2 can be also carried out to dig out the breeder. The first server which can not provide the correct record about the trade agent will be identified as a malicious one.

3.4 Anonymous payment from users to servers

In the proposed system, the servers in the transaction only know they trade with Agent Service Center. They are not able to make sure who asks the ASC to trade with them. Although some servers can directly send goods or services to the user or receive goods or services from the user, the user can also ask servers to send or receive goods or services in the ASC on basis of Transfer Account Payment $TAP_{S(i)}$ issued by Payment Service if the user wishes to be anonymous. Finally, the user receives goods or services from the ASC or sends goods or services to the ASC.

4 Service Charge function

4.1 Service charge function for Agent Service Center

When an authorized User U applies for obtaining services from Agent Service Center, he should transmits (R_U, t) to the ASC to describe what goods or services to buy or sell in a list of servers and his signature on (R_U, t). The ASC sets up a data base to save these information from users. According to the data base, the ASC can count the number of services supplied to User U and calculate the expense which should be paid to the ASC by the User U.

4.2 Ticket charge function for servers

The Login Data Base (LDB) of a server (mentioned in 2.2) provides not only the evidence which proves the server only legally modifies the agents passing through the server, but also a ticket charge function for the server. From the LDB, the server can count the number of records whose agent certificates are same and then ask the owners of the agent certificates to pay for their entrance into the server.

5 Conclusion

A secure intelligent trade agent system is proposed above. The proposed system can effectively prevent the attack from destructive trade agents and malicious hosts because the owner of a malicious intelligent trade agent in the system is easily dug out and a host can only legally modify the information relative to it in the agent.

The secure intelligent trade agent system has two extra features: 1. The payment in the system is anonymous to servers (e.g., shops, companies). 2. It is convenient for the system to charge.

So far, any security weakness in the secure intelligent agent system has not been found yet. However, the security of the proposed system needs further intensive investigation. The interested parties are welcome to attack this proposed system and the authors will be grateful to receive the result of any such attacks.

References

1. Jaco van der Merwe and S.H.von Solms, Electronic Commerce with Secure Itelligent Trade Agents, Proceedings of ICICS'97, LNCS 1334, Novermber 1997, pp.452-462.
2. D.Chess, C.Harrison and A.Kershenbaum, Mobile Agents: Are They a Good Idea. Technical Report, March 1995, IBM T.J.Watson Research Center, NY.
3. D.Chess, B.Grosof, C.Harrison, D.Levine, C.Parris and G.Tsudik, Itinerant Agents for Mobile Computing. Technical Report, October 1995, IBM T.J.Watson Research Center, NY.
4. L.Rasmusson and S.Janson, Simulated Social Control for Secure Internet Commerce. In New Security Paradigms'96, ACM Press, September 1996.
5. R.Rivest, A.Shamir and L.M.Adleman, A Method for Obtaining Digital Signature and Public-Key Cryptosystem, Communications of ACM, v.21, n.2, February 1978, pp.120-126.
6. R.L.Rivest, The MD4 Message Digest Algorithm, RFC 1320, April 1992.
7. R.L.Rivest, The MD5 Message Digest Algorithm, RFC 1321, April 1992.
8. J.White, Telescript Technology: The Foundation of the Electronic Market Place, General Magic white paper 1995.
9. N.Borenstein, EMail with a Mind of its Own: The Safe-Tcl Language for Enabled Mail, IFIP WG 65 Conference, Barcelona, May, 1994, North Holland, Amsterdam, 1994.