Security Issues in Mobile Agents

E C Vijil
School of Information Technology
vijil@it.iitb.ac.in

16 January 2002

Overview of the Talk

- The Mobile Agent Paradigm
- Security Threats and Counter Measures
- Security in Data Collection Agents
- Our Proposals
- Conclusion and Future Work

The Mobile Agent Paradigm

- An executing program that can migrate from machine to machine in a heterogeneous network
- Execution environment provided by supporting hosts
- Follows either a pre-assigned path or determines its itinerary dynamically

Client/Server vs Mobile Agents

- Client/Server
  - ⋆ Data resides on the server
  - ⋆ Services provided by the server
  - ⋆ Interaction through the UI provided by the Server
  - ⋆ Network Connection retained for the entire duration of the transaction

Client/Server vs Mobile Agents

- Client/Server
  - ⋆ Data resides on the server
  - ⋆ Services provided by the server
  - ⋆ Interaction through the UI provided by the Server
  - ⋆ Network Connection retained for the entire duration of the transaction
- What if
  - ⋆ The user has very specific requirements?
    - ∗ Give me the list of books published this year by last year's best selling author?
  - ⋆ Application is data intensive?
    - ∗ Give me all postings referring to my paper in sci.crypt newsgroup
  - ⋆ You cannot remain online for the entire duration of the transaction?
  - ⋆ Dynamic Deployment of Software

Where are Mobile Agents useful?

- Everything that can be done using mobile agents can also be done using CS
- No ‘killer application’ for mobile agents
- Mobile Agents more efficient for some applications
  - ⋆ Data Intensive Operations
  - ⋆ Disconnected Operations
  - ⋆ Dynamic Deployment of Software
  - ⋆ Highly user specific applications

Security Threats

- Agent can attack the platform
  - Denial of Service
  - Unauthorized access
  - Masquerading
- Platform can attack the agent
  - Most difficult to tackle
  - Eavesdropping
    - Could be exposing proprietary algorithms
    - Privacy concerns
  - Alteration of data and code
  - Masquerading
    - Lowest price finding agent

Problem Scope

- Data Collection Agents
  - ⋆ Problem of Malicious Hosts
    - ∗ Idenitifying the malicious host making deletions
    - ∗ Detecting attacks by Colluding Malicious hosts

Data Collection Agents

- Visit multiple sites to collect data
  - ⋆ Typical Example: Shopping agents
- Security Issues
  - ⋆ Modification of Data
  - ⋆ Deletion of Data
  - ⋆ Colluding Malicious hosts
- Ajanta Mobile Agent System
  - ⋆ A mobile agent framework designed with security in mind
- Assumptions
  - ⋆ There exists a reliable Public Key Infrastructure (PKI)
  - ⋆ There are no intruders in the medium

Modification of Data by Malicious Hosts

- A Malicious host modifies the data added by other hosts

- Solution - ReadOnlyContainer
  - ⋆ Array of data items collected from each host
  - ⋆ Sign each data item using the host's private key
  - ⋆ Encrypt using the initiator's public key if necessary
  - ⋆ Data structures
    - ∗ V: item1, item2, item3
    - ∗ S: sign1, sign2, sign3
  - ⋆ Owner verifies the signature of each data item

Deletion of Data by Malicious Hosts

- A Malicious host deletes the data added by other hosts
- Solution - AppendOnlyContainer
- Notation
  - ⋆ E_A : Encryption using *public* key of A
  - ⋆ D_A : Encryption using *private* key of A
  - ⋆ $Sig_A(X)$: Signing of data X using private key of A

AppendOnlyContainer

- Initialization at the Owner's site
  - ⋆ $checkSum = E_{owner}(N_a)$
- Updation of checksum by a host C adding dataitem X
  - ⋆ $checkSum = E_{owner}(checkSum + Sig_C(X) + C)$
- Verification at the Owner's site
  - ⋆ The owner decrypts and separates the fields in the checksum
    - ∗ $D_A(checkSum) \Rightarrow checkSum + Sig_C(X) + C$
  - ⋆ And verifies the signature
    - ∗ $E_C(Sig_C(X)) == hash(X)$
    - ∗ This is repeated for all data items
    - ∗ If verification succeeds we will be able to recover the original random nonce

AppendOnlyContainer - An Example

- Hosts A, B, C adds items X, Y, Z respectively - Vector V contains the individual data items.

- Initialization
  - ⋆ $checkSum = E_O(nonce)$

- Updation of checksum by host A adding dataitem X
  - ⋆ $checkSum = E_O(E_O(nonce) + Sig_A(X) + A)$
  - ⋆ V contains : X

- Updation of checksum by host B adding dataitem Y
  - ⋆ $checkSum = E_O(\overbrace{E_O(E_O(nonce) + Sig_A(X) + A)}^{\text{checksum after the addition of X}} + Sig_B(Y) + B)$
  - ⋆ V contains : X, Y

AppendOnlyContainer - An Example (Contd...)

- Updation of checksum by host C adding dataitem Z
  - $checkSum =$
    $$E_O(\overbrace{E_O(E_O(E_O(nonce) + Sig_A(X) + A) + Sig_B(Y) + B)}^{\text{checksum after the addition of Y}} + Sig_C(Z) + C)$$
  - V contains : X, Y, Z

Problems with AppendOnly Container

- Can only detect that a modification/deletion has taken place
- Cannot identify the host doing the modification deletion
- Identification of the malicious host is important to prevent future modifications

Identifying malicious hosts - Proposed solution

- Main idea
  - ⋆ AppendOnlyContainer signs each data item separately
  - ⋆ Instead sign all the data carried by the agent together
- The checksum update procedure is modified as follows
  - ⋆ Original : $checkSum = E_{owner}(checkSum + Sig_C(X) + C)$
  - ⋆ Our Proposal : $checkSum = E_{owner}(checkSum + Sig_C(data) + C)$
- If verification fails while decrypting the data added by $Host_i$
  - ⋆ Either $Host_i$ or $Host_{i+1}$ is the malicious host.

SecureContainer - An Example

- Hosts A, B, C adds items X, Y, Z respectively - Vector V contains the individual data items.

- Initialization
  - $\star$ $checkSum = E_O(nonce)$

- Updation of checksum by host A adding dataitem X
  - $\star$ $checkSum = E_O(E_O(nonce) + Sig_A(X) + A)$
  - $\star$ V contains : X

- Updation of checksum by host B adding dataitem Y
  - $\star$ $checkSum = E_O(\overbrace{E_O(E_O(nonce) + Sig_A(X) + A)}^{\text{checksum after the addition of X}} + Sig_B(X, Y) + B)$
  - $\star$ V contains : X, Y

SecureContainer - An Example (Contd...)

- Updation of checksum by host C adding dataitem Z
  - ⋆ $checkSum =$
    $E_O(\overbrace{E_O(E_O(E_O(nonce) + Sig_A(X) + A) + Sig_B(X, Y) + B)}^{\text{checksum after the addition of Y}} + Sig_C(X, Y, Z) + C)$
  - ⋆ V contains : X, Y, Z

Collusion in Data Collection Agents

- Two or more hosts jointly attacking an agent
- The colluding hosts can share information
- Can they do better than hosts acting individually?

Deletion of data by colluding malicious hosts

- Two or more hosts can collude to delete data items from the AppendOnlyContainer
- Itinerary H_1, H_2, H_3, $\cdots$, H_i, H_{i+1}, $\cdots$, H_j, H_{j+1}, $\cdots$, H_n
- H_i does the following:
  1. It adds its own data D_i, to the $AppendOnlyContainer$.
  2. It recomputes the checksum. We shall denote this checksum by $checkSum_i$.
  3. It sends $checkSum_i$ to H_{j+1}.
- H_{j+1} on receiving the agent does the following:
  1. It adds its own data D_{j+1}, to the $AppendOnlyContainer$.
  2. It recomputes the checksum. But, instead of using the current value of checksum carried by the agent, it uses $checkSum_i$.
  3. It removes data items $D_i, \cdots, D_j$ from the $AppendOnlyContainer$

Detecting Collusions

- Static Itinerary
- Dynamic Itinerary
  - ⋆ Notification by hosts
    - ∗ Prevents disconnected operations
  - ⋆ Querying by the agent initiator
    - ∗ Allows disconnected operations
    - ∗ Higher message overhead

Our Approach

- Both these solutions involves message overhead which can be avoided
- Expected Number of Deleted Hosts (ENDH)
- Owner assumes k out of n hosts are malicious
- $P(i)$ is the probability that exactly i hosts are deleted
- $ENDH = \sum_{i=0}^{n-2} i.P(i)$
- Notification by Proactive Hosts
- Querying by the Agent Initiator

Our Approach (Contd...)

- Notification by Proactive Hosts
  - ⋆ Each host notifies the initiator with probability $\frac{ENDH}{n}$
- Querying by the Agent Initiator
  - ⋆ Agent initiator queries with probability $\frac{ENDH}{n}$
- Experimentation
  - ⋆ Notification by Proactive Hosts
    - ∗ Accuracy of more than 90% with about 67% reduction in the number of messages
  - ⋆ Querying by the Agent Initiator
    - ∗ Accuracy of more than 90% with about 25% reduction in the number of messages

Conclusions

- Mobile Agents are a useful programming paradigm
- Its utitility is limited if security threats are not mitigated
- Problem of Malicious hosts - Difficult to tackle
- Our solutions
  - ⋆ Identify the malicious host in data collection agents
  - ⋆ A probabilistic scheme for detecting collusions