Applying Trust Policies for Protecting Mobile Agents Against DoS

Biljana Cubaleska[1], Markus Schneider[2]

[1]University of Hagen, Dept. Of Communication Systems, Germany
[2]Fraunhofergesellschaft, Darmstadt, Germany

Overview

- Motivation: Security problems with mobile code
- Denial of Service (DoS) attacks
- Detection of malicious hosts
- Trust policy and cost reduction
- Conclusion

Security problems with mobile code

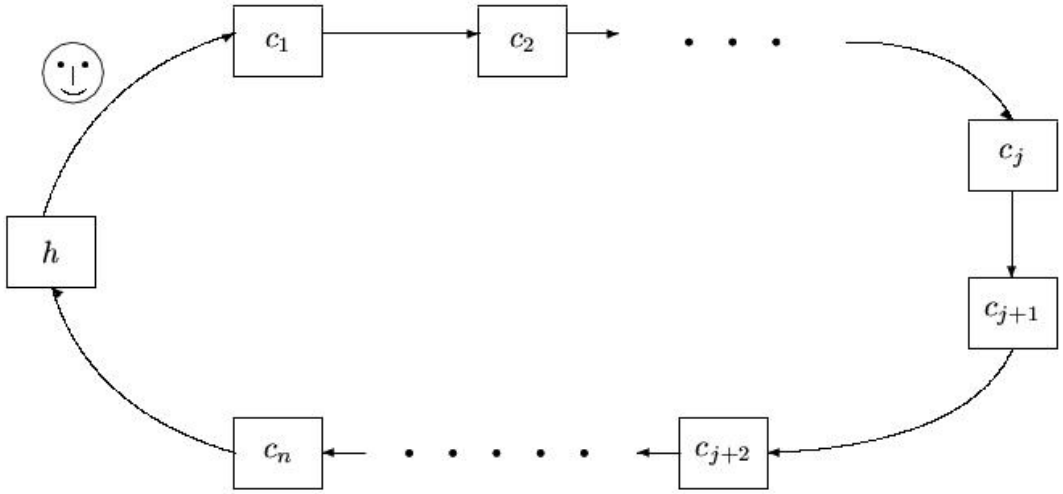
- **Mobile agents**

  autonomous programs which migrate through a network of sites to accomplish tasks on behalf of their owners

- **Security threats**
  - **Both the visited hosts and the agents are exposed to serious dangers**
  - **Malicious agent can attack the host platform**
    - E.g. unauthorized access to resources, altering or deleting it, Trojan horse functionality
  - **Malicious host platform can attack the agent**
    - E.g. Extract private information, steel digital goods, modify agent data, denial of service

Denial of Service from malicious Host

Normal case:
The hosts in the network offer their services to the agents



- **„Denial of Service“ in this context: Some host reject to give its services to the agent**
  - **The agent owner can have benefits of using the agent system only if it works properly and if the visited hosts are willing to serve the agents, i.e. these hosts make their services available**
- **Mechanisms which enable detecting the hosts performing DoS are important!**

Types of Denial of Service

- **Partial DoS: A visited host**
  - does not execute the agent, or
  - does not execute it properly, or
  - put in the mobile data false results (Problem of integrity of computation)

  but, **allows the agent to continue his journey!**

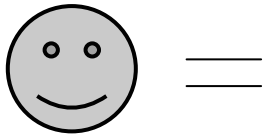
- **Total DoS**
  - A visited host is not willing to let an agent to continue its route, it deletes or „kills“ the agent
    - **The agent cannot return to his home**
    - **All results collected by the agent so far will be lost**

    A mechanism which tackles these problems is required!

Our solution against total DoS

- **Deleting agents (total DoS) cannot be *a priori* prevented**
- **We propose a mechanism for *a posteriori* identification of the attacking Host**
  - Combination of cryptographic primitives and a fixed set of rules
- **Personal trust policy**
  - The information WHO was the attacking host is used from the agent owner to build a trust model for the hosts he is dealing with
- **Preventive effect**
  - This knowledge is used from the owner when composing the future agent routes
- **Assumption: Independent results** (a computation does not require the results produced at any other host as input)

Agent components



=

Agent

binary code | mobile data | uid | route | logbook | another infos

$$agent^{j} = (bc, md^{j}, uid, r, vc^{\#(c_j)})$$

$agent^{j}$ - Agent residing at host c_j after being executed

bc - Binary code of the Agent

md^{j} - mobile data contained in the agent after execution at c_j $\Rightarrow$ $md^{j-1} \subset md^{j}$
(md^{0} could be control data given from h)

uid - Unique Identifier of the Agent

$r = (\, ip(c_1), \ \ldots \ , ip(c_j), \ \ldots \ , ip(c_n))$ - Agent route (hosts to be visited) given from h

$$vc^{\#(c_j)} = \underbrace{ip(c_{i_1}), \ \ldots \ , ip(c_j)}_{\#(c_j) \text{ elements}}$$

- Sequence of already visited hosts $i_1 \in \{1, \ \ldots \ , n\}$
vc^{0} is empty (before the first migration)

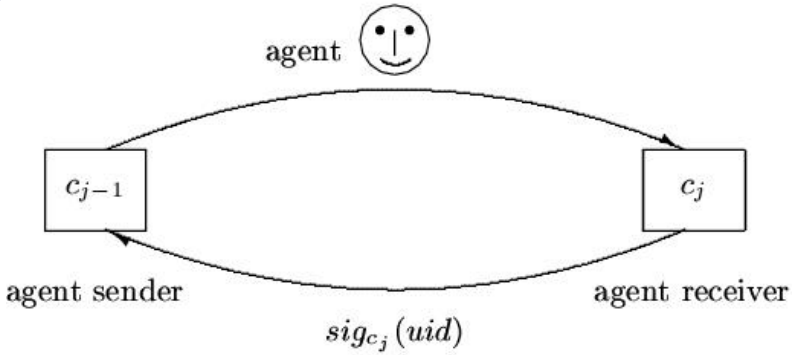
- $\#(c_j)$ number of already visited hosts

Towards the solution

- **Idea: Usage of undeniable proofs**
  - When an agent owner does not receive his agent after some waiting time, there arouses suspicion that the agent suffered DoS by a malicious host
  - The agent owner asks all hosts contained in the route to show him a proof that they correctly dispatched the agent
  - The attacking host is not able to show such a proof
- **Undeniable proofs can be realized with the technique of digital signatures**

Important step: Exchange of Agent and confirmation

- Rule: Upon receiving an agent, each host must send a confirmation to its predecessor

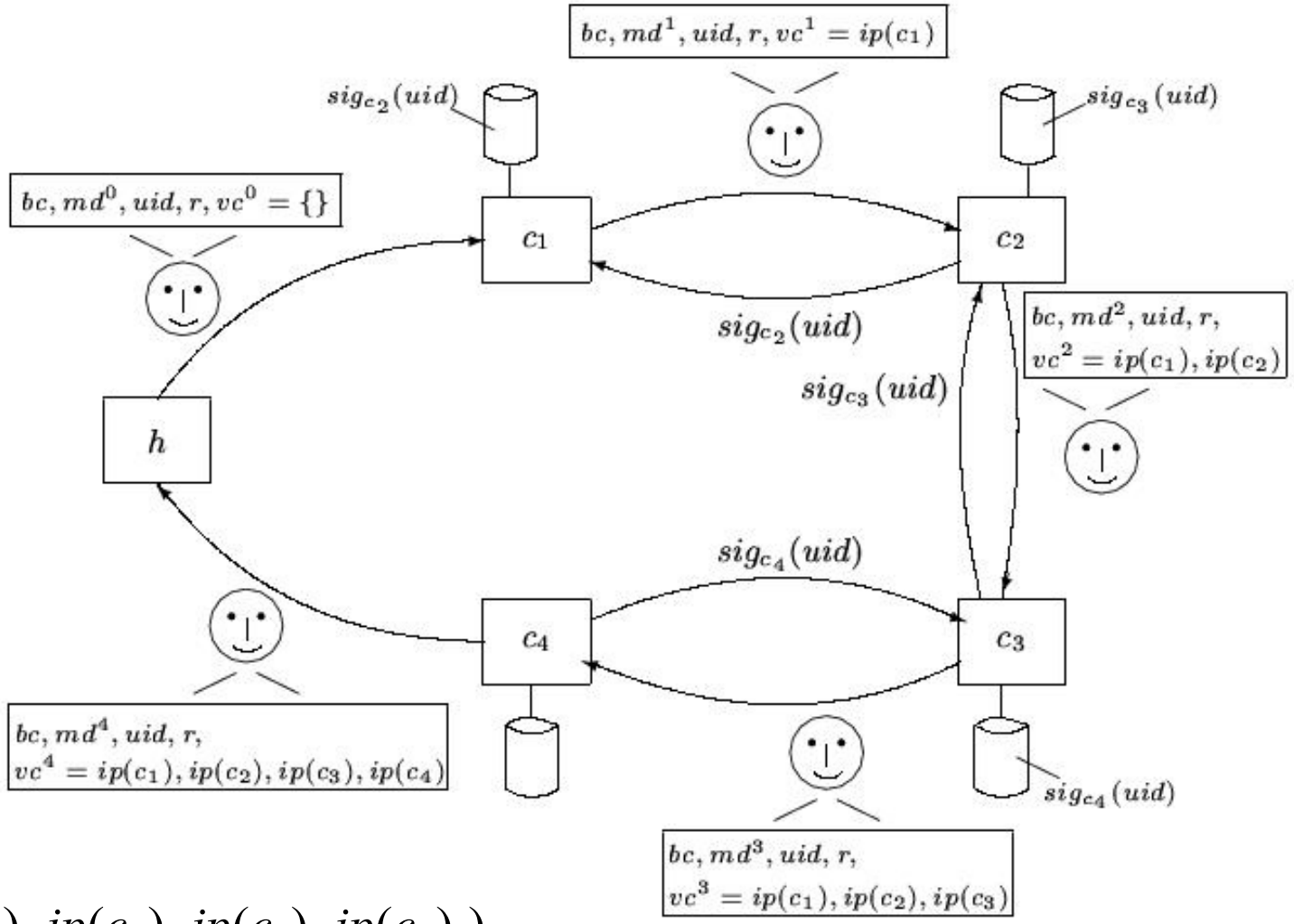


The confirmation is signature from c_J :

$sig_{cj}(uid)$

- Protocols
  - Sender protocol
  - Receiver protocol
- Investigation Procedure
  - The agent owner want to see the confirmations of all hosts that they properly dispatched the agent
- The agent owner modifies his personal trust policy

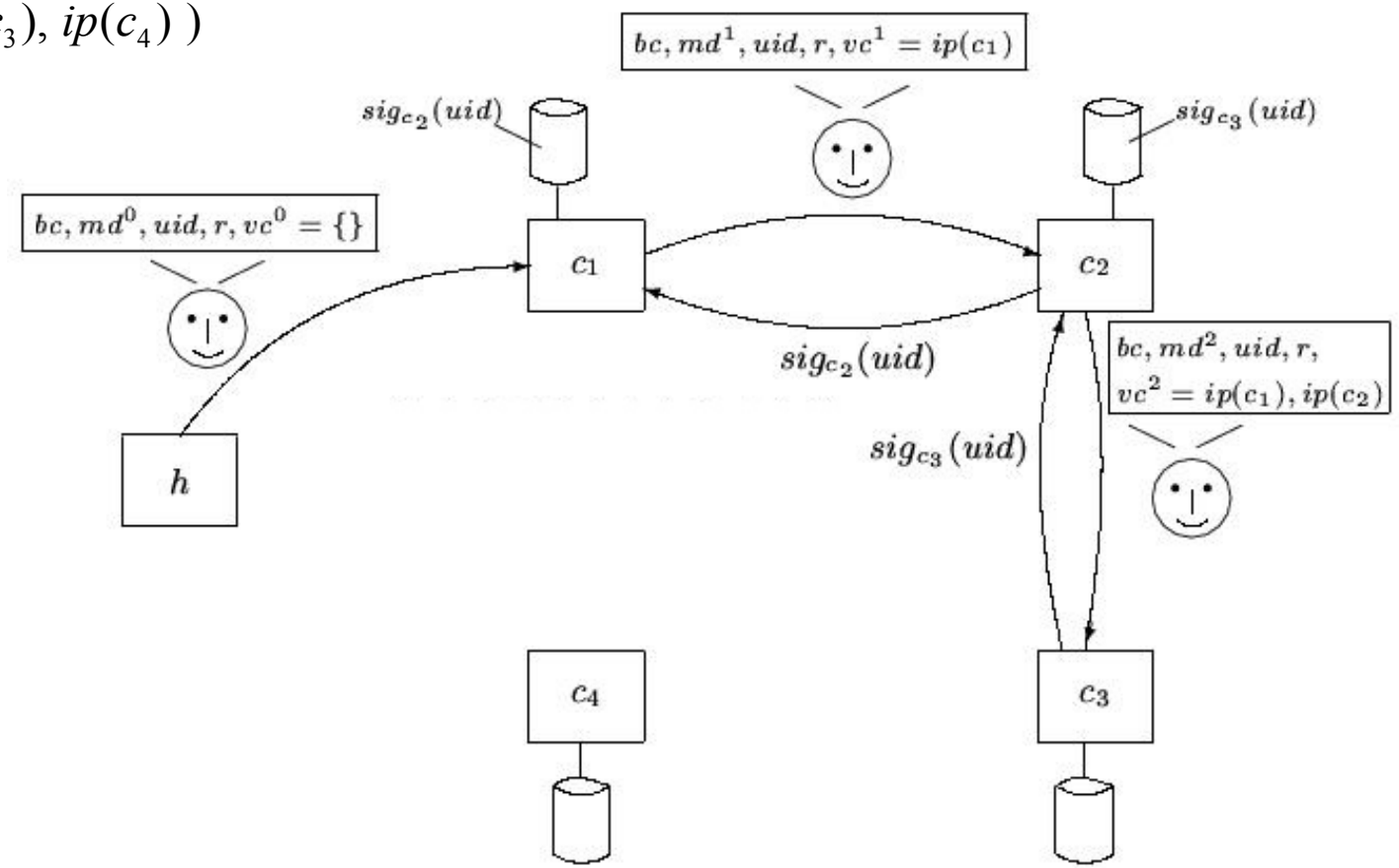
Example: Agent journey without DoS



$$r = (\ ip(c_1),\ ip(c_2),\ ip(c_3),\ ip(c_4)\ )$$

Example: Agent journey with DoS

$r = (\ ip(c_1),\ ip(c_2),\ ip(c_3),\ ip(c_4)\ )$



- c_3 performs DoS
- In an investigation procedure from h, c_3 cannot show him an evidence that it dispatched the agent to c_4

Enhancing the simple solution

- **But, what in the case when some hosts does not „play“ according to the rules?**
  - E. g. Some host does not send confirmation to its predecessor although it successfully received the agent, some host skip the next one, etc.
- **The exchanging of agent and confirmation was built in a protocols which enable correct results in all cases**
- **Some agent components must be modified and new system parameter must be added:**
  - **E.g.** buf **(each host has a buffer for each agent to be processed)**

m **(maximum number of hosts that should try to contact another host which is not answering properly)**

$$\tilde{m} = (m, sig_h(m))$$

$$vc^{\#(c_l)} = vc^{\#(c_k)}, ip(c_l), sig_{c_l}(vc^{\#(c_k)}, ip(c_l))$$ **(nested signatures)**

$$sig_{c_l}(uid, vc^{\#(c_k)})$$ **(list of visited hosts included in the confirmation)**

Sender and receiver protocol

Sender protocol:
(executed at c_j after the execution of the agent)

```
1   Execute algorithm SelectNextHost
2   If (NextHost == h)
        stop
3   Send agent to NextHost found by SelectNextHost in step 1
4   Store a copy of the agent
5   Until (confirmation not received and time-out not reached)
        wait for confirmation
6   If (confirmation received and confirmation valid)
        store confirmation in local database
    else
        go to step 1
7   Delete agent copy
8   End.
```

Receiver protocol:

```
1   Receive agent
2   Create confirmation
3   Send confirmation
4   End.
```

Selecting the next host to be visited

- Subroutine of the sender protocol

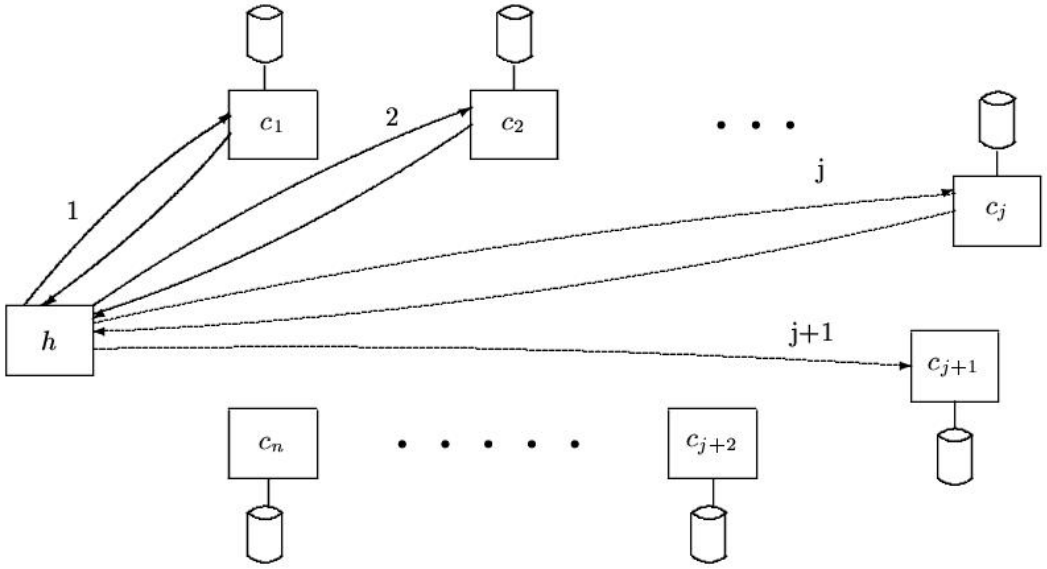
- When the next host in the route is not reachable or when it doesn‘t send a confirmation, then the next host to be visited is determined from this algorithm.

Algorithm ***SelectNextHost*** (at c_j).

```
if (#(c_j) < j)
    i = 1
    while (i < j − 1)
        if (¬in(c_i, {vc^#(c_j)}) ∧ ¬in(c_i, buf))
            if (card({c_{i+1}, ..., c_{j−1}} ∩ {vc^#(c_j)})) < m − 1
                append c_i to buf
                NextHost = c_i
                i = j − 1
            else
                i = i + 1
        else
            i = i + 1
else
    i = j + 1
    while (i ≤ n)
        if (i == n)
            NextHost = h
        else
            if (¬in(c_i, {vc^#(c_j)}) ∧ ¬in(c_i, buf))
                append c_i to buf
                NextHost = c_i
                i = n + 1
            else
                i = i + 1
```

Investigation Procedure

$h \rightarrow c_j$ Request
$c_j \rightarrow h$ Evidence



- Consists of consecutive application of investigation protocol
  - Agent owners request
  - Answer in which a host shows ist evidence
- The hosts are quered in the order in which they were visited, which is not necessarily the same as those given in $\tilde{r}$

Trust Values

- The agent owner uses the output of the investigation procedure
- Definition
  - The agents owner trust value $trust(c_i)$ that host c_i will NOT perform DoS to his agents is given by $trust(c_i)=P(c_i)$
- The collection of trust values represents its trust policy
  - The initial values are estimated
  - Then, after each modification procedure the trust values are modificated (increased or decreased)
- The trust values are used to compose the future routes

Cost parameter = communication cost

- We consider the average number of migrations an agent really requires when its route contains n entities

- Let $r = (c_1, c_2, \ldots\ldots c_n)$
  $trust(c_i) = P(c_i) = p_i$ for $i = 1, \ldots n$
  X – discrete random variable that specifies the number of migrations that have been made during the agent journey
  (The sample space can consist of all values from $X=1$ to $X=n+1$)
- $P(X=i)$ for $i=1, \ldots, n$ probability that the agent migrate until host c_i, but not further
- $P(X=n+1)$ probability that the agent returns home

$P(X=1) = 1-p_1$
$P(X=i) = p_1 \ldots\ldots p_{i-1}(1-p_i)$ for $1 < i <= n$
$P(X=n+1) = p_1 p_2 \ldots\ldots p_n$

Trust policy exploitation for cost reduction

- **Expected value:**

$$E[X] = \sum_{i=1}^{n+1} i \cdot P(X = i) = 1(1-p_1) + 2\,p_1(1-p_2)$$

$$+ \ \ldots\ldots\ + n\,p_1 \ldots p_{n-1}(1-p_n) + (n+1)\,p_1 \ldots p_n$$

- **We are interested in minimum of** $E[X]$
  - **Necessary and sufficient condition**
  - **The value of** $E[X]$ **depends on the trust values of the hosts and on the ordering of the hosts in the route**
  - **The value that the agent will not suffer denial of service attack does not depend on the ordering:** $P(X=n+1) = p_1 p_2 \ldots p_n$
- **Number of possible routes: n!**
  - **Which of these routes leads to minimum** $E[X]$**?**

Minimization of *E[x]*

- **Theorem**

  Let $c_1, \ldots\ldots, c_n$ be hosts that are contained in an agent route in order to be visited in the given order. Assume that the hosts have trust values $trust(c_i) = p_i$ with $0 < p_i \leq 1$ for $i = 1, \ldots, n$.

  Then the expected value $E[X]$ is minimum if and only if

$$p_1 \leq p_2 \leq \ldots\ldots \leq p_n$$

  - **With the results of this theorem, the agent owner has a recipe how to create a route based on the trust values in his policy:**

    Increasing trust values ensure a reduction of the costs consisting of average number of connections (migrations in the agents journey or actions in the investigation procedure)

Conclusion

- Problem of DoS Attacks in mobile agent systems
- Protocols for *a posteriori* identification of the culprit host
  - The attacker can be uniquely identified
  - The proposal ensures that a host cannot be excluded from the agents journey
  - Works in case of collusion of malicious hosts
  - Output is used for adaptation of the owners trust policy
- Exploitation of the trust policies to minimize some costs which are of interest for the agent owner
- The solution has a preventive power

Thank You for Your Attention!

Biljana.Cubaleska@fernuni-hagen.de
http://cs.fernuni-hagen.de