

Security Solutions for Killer Mobile Agent Applications

Sergi Robles

Sergi.Robles@uab.es

CCD Research Group

Department of Computer Science

Universitat Autònoma de Barcelona (UAB)

Outline

1. Advantages of Mobile Agents
2. Why they are not everywhere
3. A new venue for applications
4. Agent-driven Security Architecture
5. Challenges

1. Advantages of Mobile Agents ←
2. Why they are not everywhere
3. A new venue for applications
4. Agent-driven Security Architecture
5. Challenges

1. Advantages of Mobile Agents

What is a Mobile Agent?

Opposite to “stationary” agent.

Agent that can move along with its own data and state from a source execution environment to another and resume execution there.

→ They are not bound to the system: they are free to travel among the hosts of a network.



1. Advantages of Mobile Agents

It is really a combination of two well-known technologies:

$$\text{Mobile Agents} = \text{Agent Technology} + \text{Mobile Code}$$

Agent Technology: Mid 90's. Mainly artificial intelligence community.

Mobile Code: At least 30 years. Widely used, normally with a one-hop migration model:

Postscript, applets, SQL, Internet worms, etc.

Advantages of using Mobile Agents

There are many reasons:

- ▶ Reduce network load
 - Move the computation to the data.
- ▶ Reduce network latency
 - Act locally in critical RT scenarios.
- ▶ Encapsulate protocols
 - Dynamic loading of new protocols, overcoming the legacy problem.

1. Advantages of Mobile Agents

- ▶ Asynchronous and autonomous execution
→ Open connections are not required.
- ▶ Dynamic adaption
→ Autonomous reaction to changes to maintain optimal configurations.
- ▶ Natural heterogeneity
→ Seamless system integration, computer and transport independence.
- ▶ Robustness and fault-tolerance
→ Ability to react to unfavorable situations

1. Advantages of Mobile Agents

But if they are so good, and have so many advantages, ...

...Why are they not used everywhere??

1. Advantages of Mobile Agents
2. Why they are not everywhere ←
3. A new venue for applications
4. Agent-driven Security Architecture
5. Challenges

2. Why they are not everywhere

But if they are so good, and have so many advantages, ...

...Why are they not used everywhere??

Three main reasons:

- ▶ Lack of Applications
- ▶ Lack of Security
- ▶ Lack of Awareness

2. Why they are not everywhere

Lack of applications

Many potentials uses:

- e/m-Commerce
- Personal assistance
- Secure brokering
- Distributed Inf. Retrieval
- Telecommunication network services
- Workflow and groupware
- Monitoring and notification
- Information dissemination
- Parallel processing

Nevertheless, there is a lack of compelling robust applications.

2. Why they are not everywhere

Lack of security

There are good techniques for solving many security issues, but they are difficult to use.

→ Main difficulties:

- ▶ Code/data Integrity and Privacy.
- ▶ *Malware* and malicious hosts.
- ▶ Common infrastructures.
- ▶ Usability.
- ▶ Fault-tolerance.

2. Why they are not everywhere

Lack of awareness

→ Most scientific and industry communities are not aware of mobile agents advantages.

Apart from technical issues, there have been “advertising” issues.

- ▶ Mobile code applications require no agent capabilities.
- ▶ Compelling applications for intelligent agents have enough with stationary agents.

2. Why they are not everywhere

- Security issues, and missing awareness, make killer applications hard to appear.



- Few applications does not incentive security research neither community knowledge.



→ A chicken - egg problem!!

1. Advantages of Mobile Agents
2. Why they are not everywhere
3. A new venue for applications ←
4. Agent-driven Security Architecture
5. Challenges

A new venue for applications

Mobile agent are a requirement for a new venue of applications:

Sea of Data (SoD) Applications

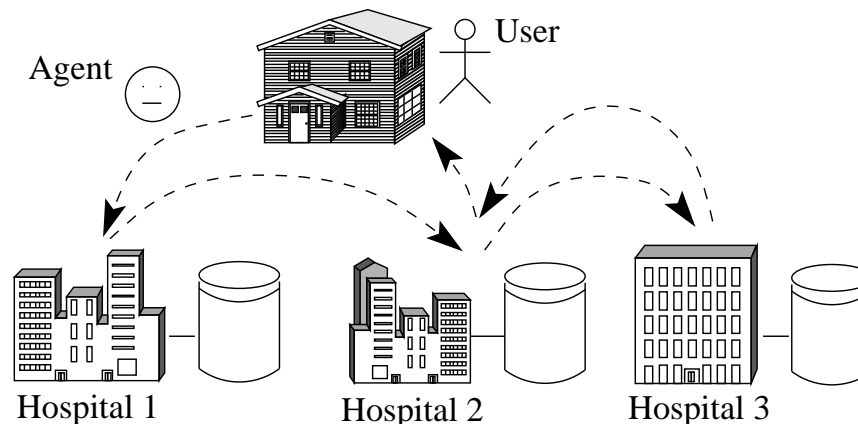
Main requirements:

- ▶ Off-line users, intermittent connectivity.
- ▶ Low bandwidth connection.
- ▶ Huge amount of distributed data.
- ▶ Legal restrictions for data access.

3. A new venue for applications

Some examples of SoD applications:

- ▶ Analysis of hyperspectral imagery.
- ▶ Distributed data mining.
- ▶ Intrusion Detection Systems.
- ▶ Training of algorithms for medical images classification.



1. Advantages of Mobile Agents
2. Why they are not everywhere
3. A new venue for applications
4. Agent-driven Security Architec. ←
5. Challenges

A key contribution to solve security issues

Our research group has been working on

- ▶ Code/data Integrity and Privacy.
- ▶ *Malware* and malicious hosts.
- ▶ Fault-tolerance.
- ▶ Authentication and Resource Access Control.

A key contribution to solve security issues

Our research group has been working on

- ▶ Code/data Integrity and Privacy.
- ▶ *Malware* and malicious hosts.
- ▶ Fault-tolerance.
- ▶ Authentication and Resource Access Control.

What we want to protect?



- ▶ Agent's Itinerary
 - Set of platforms the agent will visit.
- ▶ Agent's Tasks and Data
 - Code to be executed on each platform.

4. Agent-driven Security Architecture

There are many platforms implementing code protection mechanisms:

→ Grasshopper, SeMoA, Ajanta, ...

Do we need more security mechanisms?

4. Agent-driven Security Architecture

There are many platforms implementing code protection mechanisms:

→ Grasshopper, SeMoA, Ajanta, ...

Do we need more security mechanisms?

Yes!

→ Those security mechanisms are implemented following a **platform-driven** approach.

4. Agent-driven Security Architecture

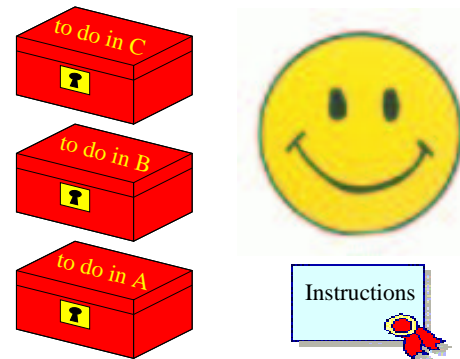
Protecting Agent Itinerary



Requirements:

- ▶ Each platform has a key-pair
- ▶ Agents can use a function that uses platform's private key to decrypt data

→ We must make sure that only agent i can get the data.



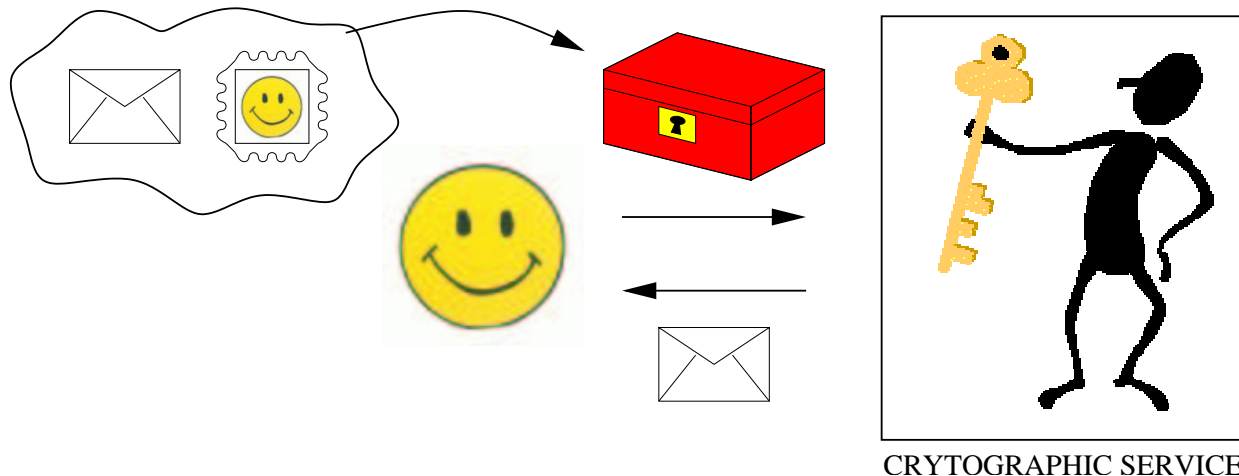
4. Agent-driven Security Architecture

Basic encrypting unit:

$$E_p(m, hash(i))$$

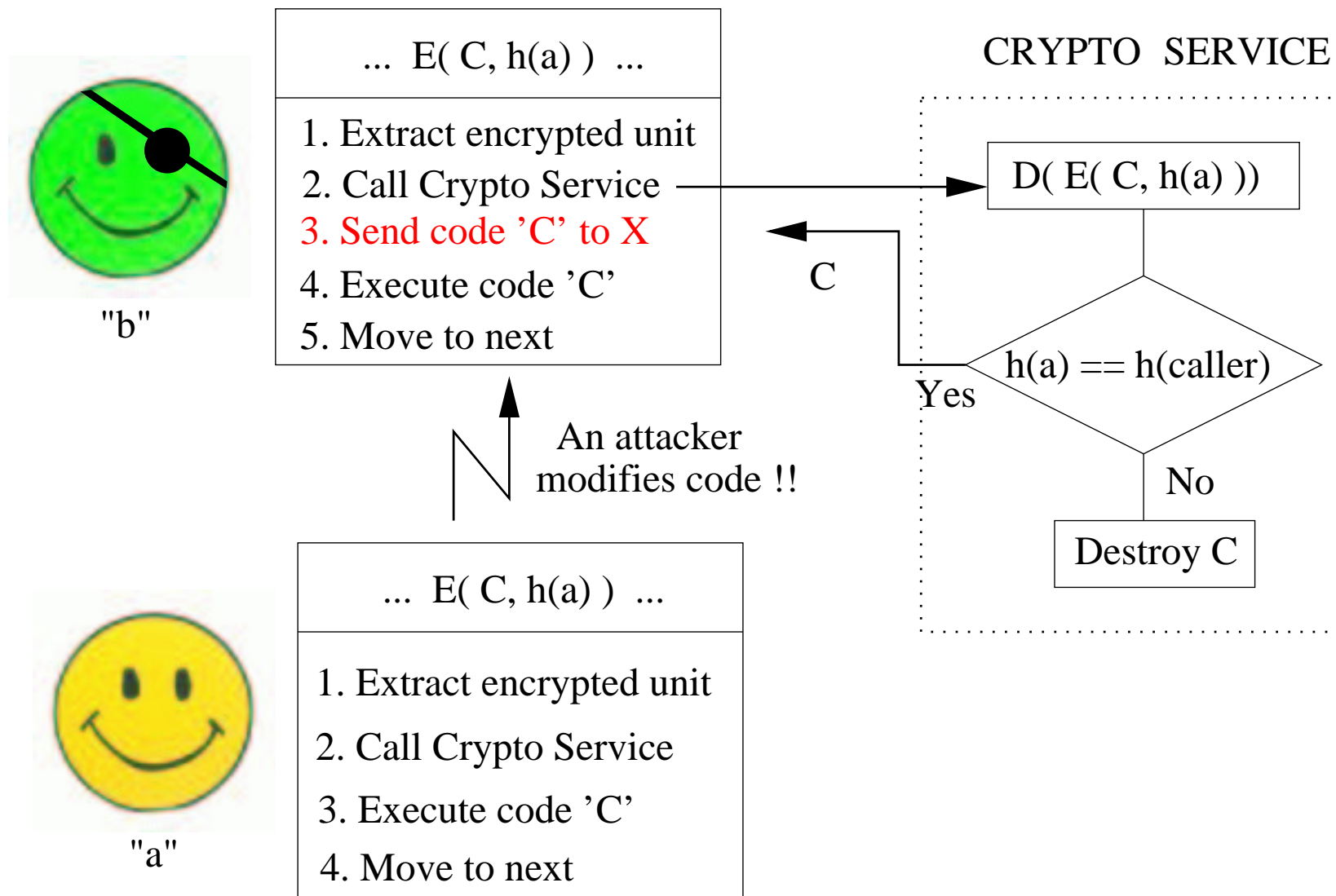
→ $hash(i)$ is a hash function of the code of the agent, a unique identifier.

We need a **Cryptographic Service** in the platform to decrypt these units under request.



4. Agent-driven Security Architecture

Example: Protection against an attack



1. Advantages of Mobile Agents
2. Why they are not everywhere
3. A new venue for applications
4. Agent-driven Security Architecture
5. Challenges ←

Challenges

- ▶ Development of killer applications
 - Implement applications that make use of mobile agent technology.

- ▶ Design and improve security mechanisms
 - Add security schemes to overcome actual limitations regarding:
 - Integrity
 - Privacy
 - Authentication
 - Usability

5. Challenges

- ▶ Scalability and Interoperability
 - Provide common infrastructures and Negotiation between different environments.
- ▶ Fault tolerance
 - Continue execution despite the unexpected occurrence of failures.
- ▶ Software designing tools
 - Include mobility considerations and develop test and debugging mechanisms for mobile agents

Conclusions

- ▶ Mobile Agents are underused: lack of applications, awareness, and security.
- ▶ There is a new venue for applications: SoD.
- ▶ There have been recent breakthroughs in security, reliability and efficiency.
- ▶ There are interesting challenges for the deployment and research of Mobile Agent technology in the next future.

Thank you!

Why they are not everywhere (cont...)

An added problem:

→ Java has enabled an easy to implement agent migration mainly due to:

- ▶ Reflexion mechanisms
- ▶ Object serialization

But Java itself include many security problems:

- ▶ No safe method to force a thread to stop
- ▶ Garbage collector can be hijacked
- ▶ Others ...

Agent-driven Security Architecture (cont...)

	Platform-driven	Agent-driven
Protection code	In the platform	In the agent
Agent internal structure	Platform must know	Any
Decryption of agent	Performed by the platform	Agent decrypts its own data
Flexibility	Low	High

Agent-driven Security Architecture (cont...)

Advantages of using agent-driven:

- ▶ There is not an imposed agent structure
- ▶ Several protection schemes can coexist
- ▶ Adding mechanisms has no impact

This approach seems to be better but. . .

If an agent needs to decrypt its data, it must be able to access the platform's private key!

→ Malware could use it to decrypt stolen data

Malicious Data Injection

Data injection

Code is protected, and data is secret, but . . .

What happens with the integrity of data?

→ Data injection is not detected!

Solution: we can design agent architectures that self protect from data injection.

But, is this possible? How we can achieve this?

Malicious Data Injection

Our solution:

1. Generate a random key pair (P_i, S_i)
2. Use private key S_i to sign all data
3. Place public key P_i inside Agent's Code.
4. Add a verification routine inside the Code



"a"

... $E(Ss(C), h(a))$...

1. key=P
2. Extract encrypted unit
3. Call Crypto Service
4. Verify $Ss(C)$ with key
5. Execute code 'C'
6. Move to next

Malicious Data Injection

Now we have code and data integrity:

- ▶ Code, including P_i and verification routine, can not be modified
→ Crypto Service would notice.
- ▶ Malicious data injections are not possible
→ Verification within the code would notice.

Added advantages:

- ▶ Same code produce different hash value
- ▶ Key-pair is disposable (do not to keep P_i)
- ▶ Verification does not use infrastructure