

# In the track of agent protection: A solution based on cryptographic hardware

Antonio Maña

[amg@lcc.uma.es](mailto:amg@lcc.uma.es)

Safe Society Labs  
University of Malaga

St. Petersburg, Sept. 2010



# Outline



- Introduction to Agents
- Description of the problem
- Trusted Computing
- SecMiLiA
- Conclusions



# Introduction to agents

- A mobile agent is defined as:
  - an autonomous,
  - reactive,
  - goal oriented,
  - adaptive,
  - persistent,
  - socially aware software entity.
- Mobile agents can actively migrate from host to host and continue its execution on the destination host
- Mobile agents include code, data and execution state
- They are not bound to the system on which they begin execution
- They are free to travel among the hosts in the network



# Description of the Problem

- Agents represent an appropriate paradigm for many new scenarios such as ambient intelligence, ubiquitous, autonomic and cloud computing
- Security problems
  - One way protection
    - Protection of agents
      - Sanctuaries
      - Obfuscation of code
      - Watermarking
      - Protected Computing
    - Protection of agencies
      - SandBoxing
      - Proof-Carrying code & Proof Referencing Code
      - (+ others like Path Histories, State Appraisal, Signed Code techniques....)




## Description of the problem

- Current agent platforms have a low level of security (Aglets, Cougar, JACK, JADE, JAVACT, AgentSpeak )
- We target a two-way protection
- In previous works we have proposed an approach based on the Protected Computing paradigm and a protocol for the remote attestation of the agencies where agents are going to run
- In this paper we present a solution based on the Trusted Computing technology and in particular:
  - An enhancement of the previous protocol based on the use of special TPM keys + results of formal verification
  - A library that implements it
  - The relation of this solution with cloud computing



## Description of the problem

- We focus on solving the malicious host problem
- Our target is to develop a trusted migration process
- Our solution contributes both to protecting agents and platforms Trusted Platform Module
- We base our solution on the TPM functionalities
- Shows a possible independent application of the TPM



## Problems of the Current approaches

- They do not provide a complete solution (only partial ones)
- Their integration in current agent tools is not easy (f.i JADE,JavaAct,...)
- Do not use state of the art security
- Too hard to apply for non security experts.

# Trusted Computing



- **Origin**

Bill Arbaugh, Dave Farber and Jonathan Smith,  
***“A Secure and Reliable Bootstrap  
Architecture”***

IEEE Symposium on Security and Privacy (1997)

- **Current Status**

**Trusted Computing Group Specifications,**  
Available from

[www.trustedcomputinggroup.org](http://www.trustedcomputinggroup.org)

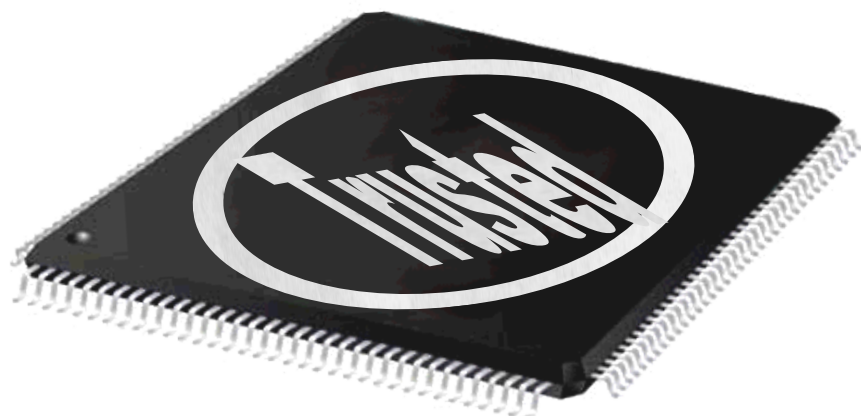


# Trusted Computing

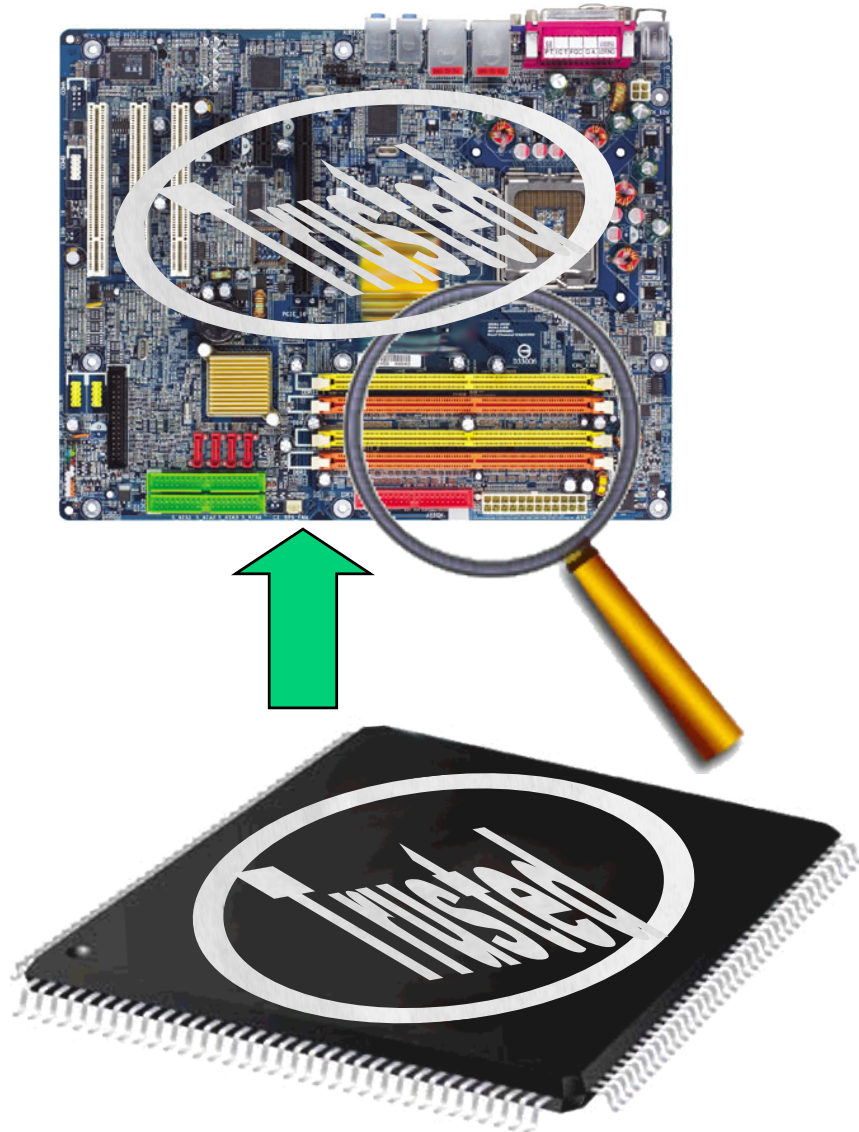


- Basis
  - A tamperproof hardware device is used to build a fully secured system **bottom-up**
  - The basic idea is to create a chain of **trust** between all elements in the computing system.
  - Normally in TC:
    - In a Trusted Computing scenario a trusted application runs exclusively on top of trusted supporting software.
    - A tamperproof hardware device analyses the BIOS of the computer and, in case it is recognized as trusted, passes control to it.
      - This process is repeated for the boot sector, the OS and the applications...

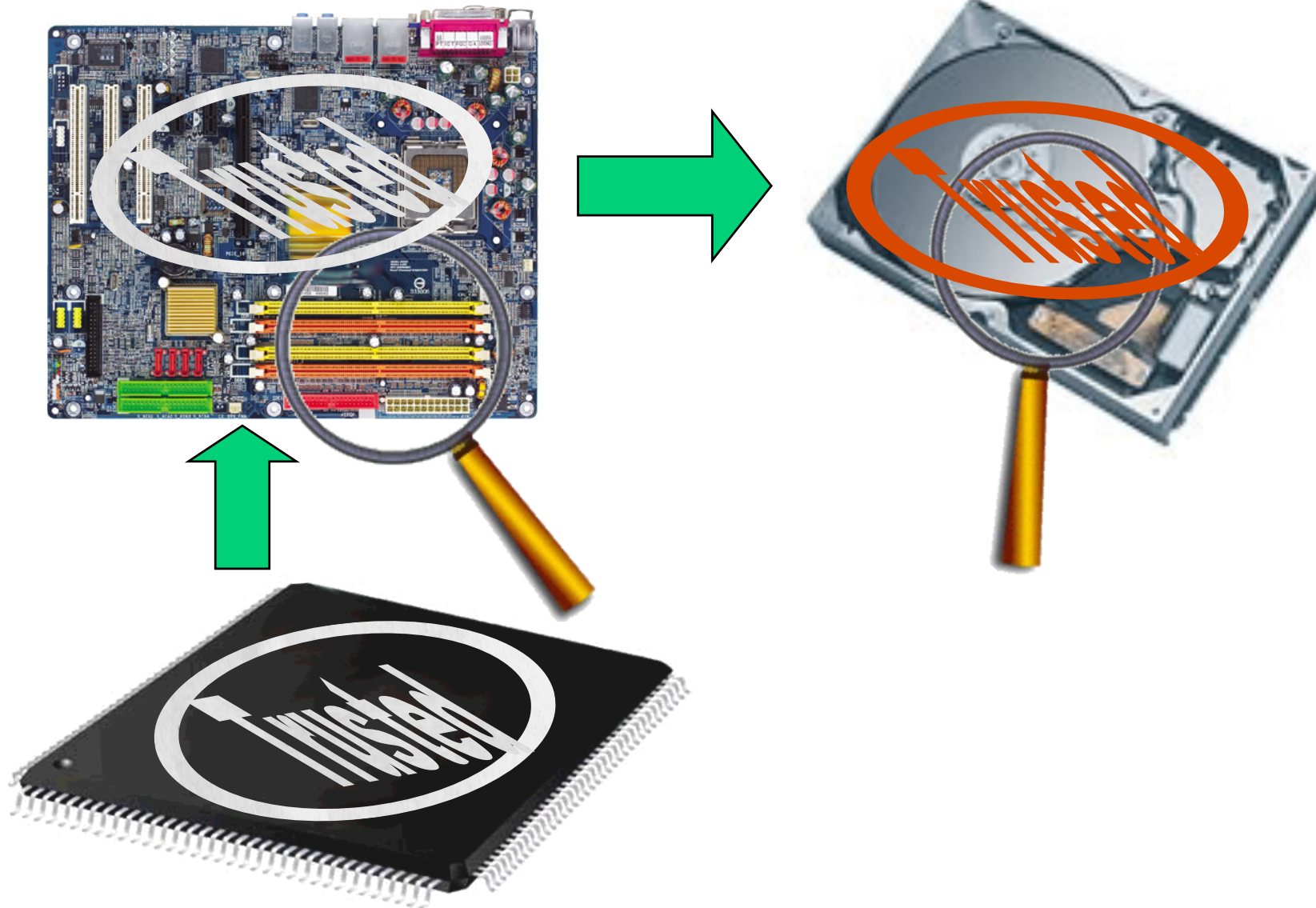
# Trusted Boot 101 Graphically



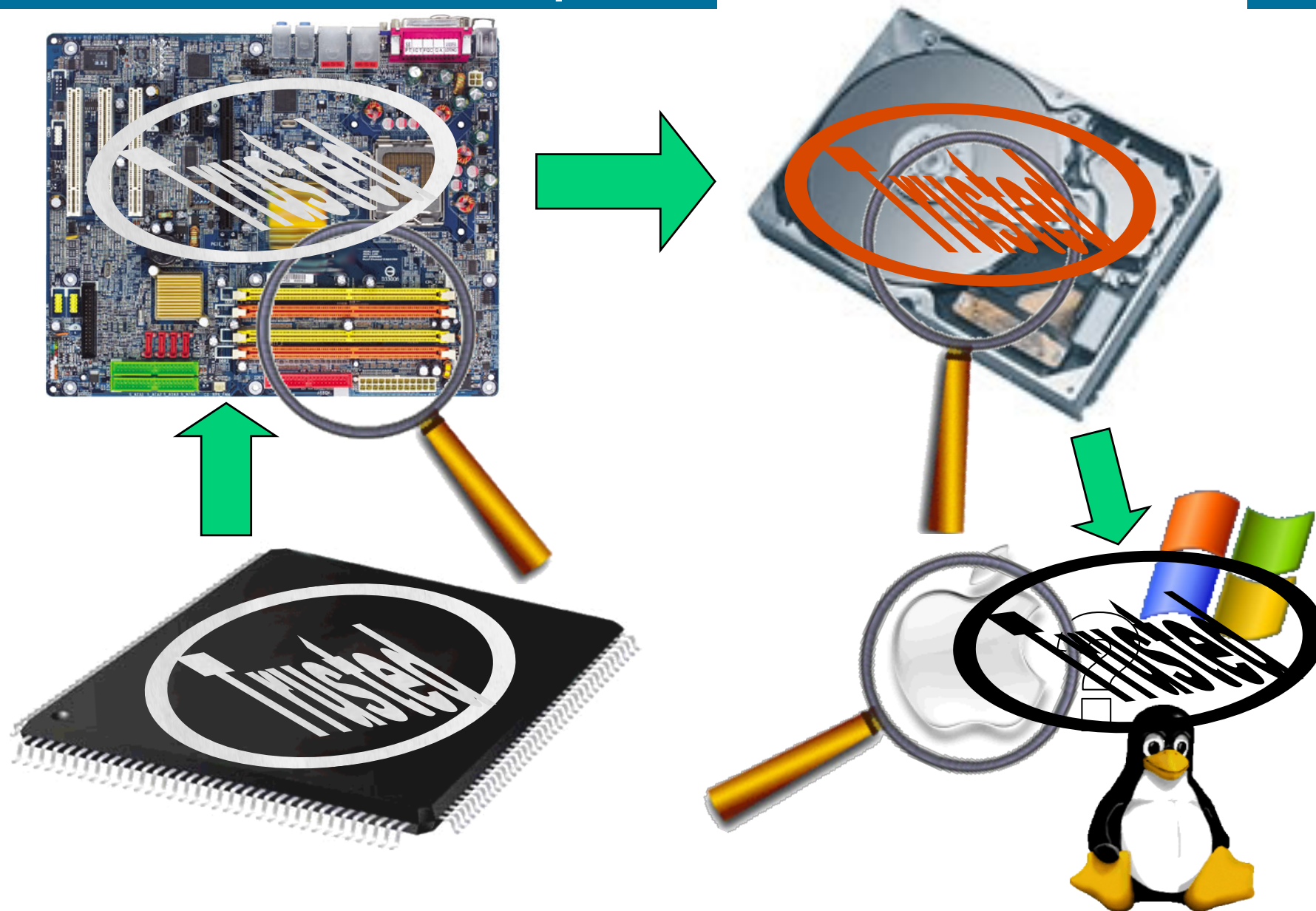
# Trusted Boot 101 Graphically



# Trusted Boot 101 Graphically



# Trusted Boot 101 Graphically





# Trusted Computing

- Main Advantages:
  - The necessary trusted hardware is integrated in the heart of the computing system
  - Fully *secure* systems are possible...
    - well, ... provided everything is perfect !
- This approach can be used to provide a secure environment for agent execution
- BUT:
  - TC is extremely tricky and difficult to apply in practice
  - TC can give a (dangerous) false impression of security



## SecMiLiA Preliminaries: FIPA & JADE

- FIPA (Federation of Information Processing Agents) is the standards organization for agents and multi-agent systems (now officially part of the IEEE).
- JADE (Java Agent DEvelopment framework) is a software Framework fully implemented in Java language.
  - It simplifies the implementation of multi-agent systems through a middleware that complies with the FIPA specifications and through a set of graphical tools that supports the debugging and deployment phases.



# SecMiLiA requirements

- Each hosting platform contains a TPM.
- The state of the Trusted Agent platform is measured and the measurements stored to the TPM PCRs.
- The initial host platform from which the mobile agent originates is considered trusted.
- Any static agent information is digitally signed by the originator.
- The use of PCR registers to store measurements (representing a trusted agency's software state) is consistent amongst all the trusted platforms.
- Every Trusted platform has registered at least one of their AIKs with a Privacy-CA which is known to the other trusted agencies participating in the system.





# SecMiLiA requirements

- Each hosting platform contains a TPM.
- The state of the Trusted Agent platform is measured and the measurements stored to the TPM PCRs.
- The initial host platform from which the mobile agent originates is considered trusted.
- Any static agent information is digitally signed by the originator.
- The use of PCR registers to store measurements (representing a trusted agency's software state) is consistent amongst all the trusted platforms.
- Every Trusted platform has registered at least one of their AIKs with a Privacy-CA which is known to the other trusted agencies participating in the system.

Platform Configuration Registers



# SecMiLiA requirements

- Each hosting platform contains a TPM.
- The state of the Trusted Agent platform is measured and the measurements stored to the TPM PCRs.
- The initial host platform from which the mobile agent originates is considered trusted.
- Any static agent information is digitally signed by the originator.
- The use of PCR registers to store measurements (representing a trusted agency's software state) is consistent amongst all the trusted platforms.
- Every Trusted platform has registered at least one of their AIKs with a Privacy-CA which is known to the other trusted agencies participating in the system.



# SecMiLiA requirements

- Each hosting platform contains a TPM.
- The state of the Trusted Agent platform is measured and the measurements stored to the TPM PCRs.
- The initial host platform from which the mobile agent originates is considered trusted.
- Any static agent information is digitally signed by the originator.
- The use of PCR registers to store measurements (representing a trusted agency's software state) is consistent amongst all the trusted platforms.
- Every Trusted platform has registered at least one of their AIKs with a Privacy-CA which is known to the other trusted agencies participating in the system.

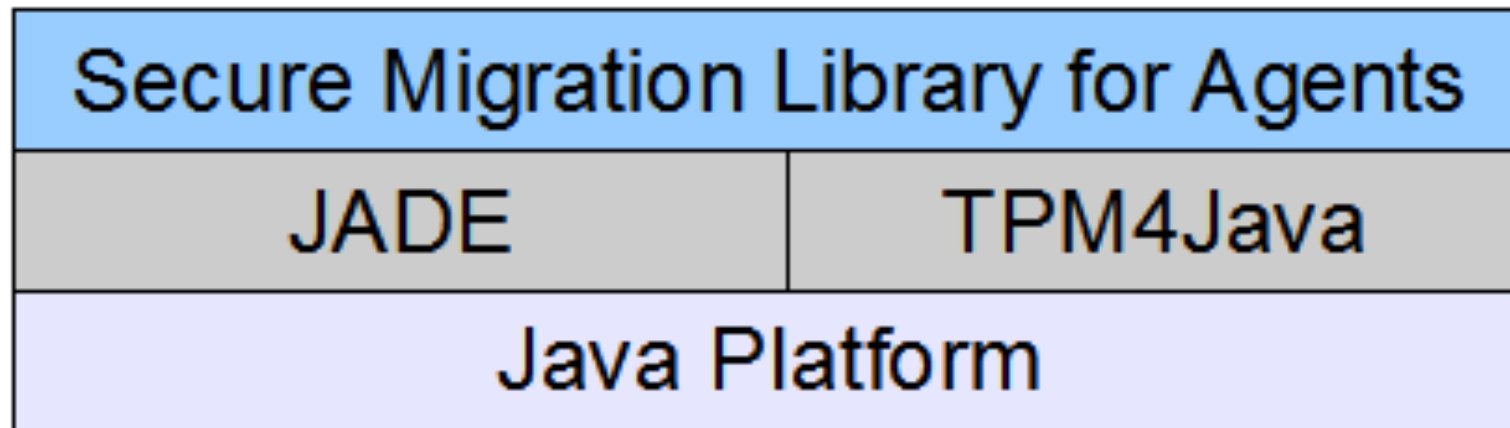
Attestation Identity Keys



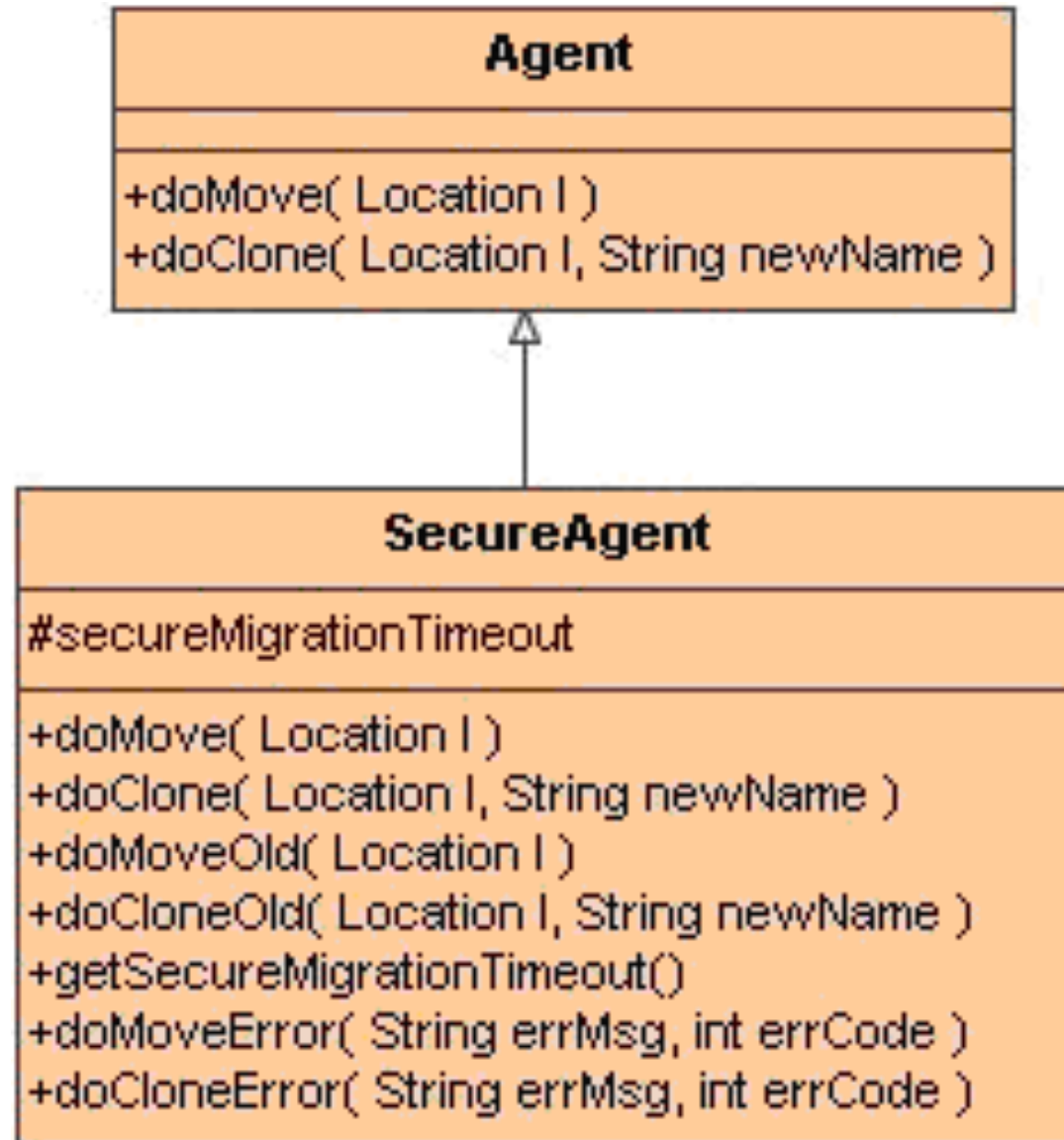
# SecMiLiA requirements

- Each hosting platform contains a TPM.
- The state of the Trusted Agent platform is measured and the measurements stored to the TPM PCRs.
- The initial host platform from which the mobile agent originates is considered trusted.
- Any static agent information is digitally signed by the originator.
- The use of PCR registers to store measurements (representing a trusted agency's software state) is consistent amongst all the trusted platforms.
- Every Trusted platform has registered at least one of their AIKs with a Privacy-CA which is known to the other trusted agencies participating in the system.

# SecMiLiA Architecture

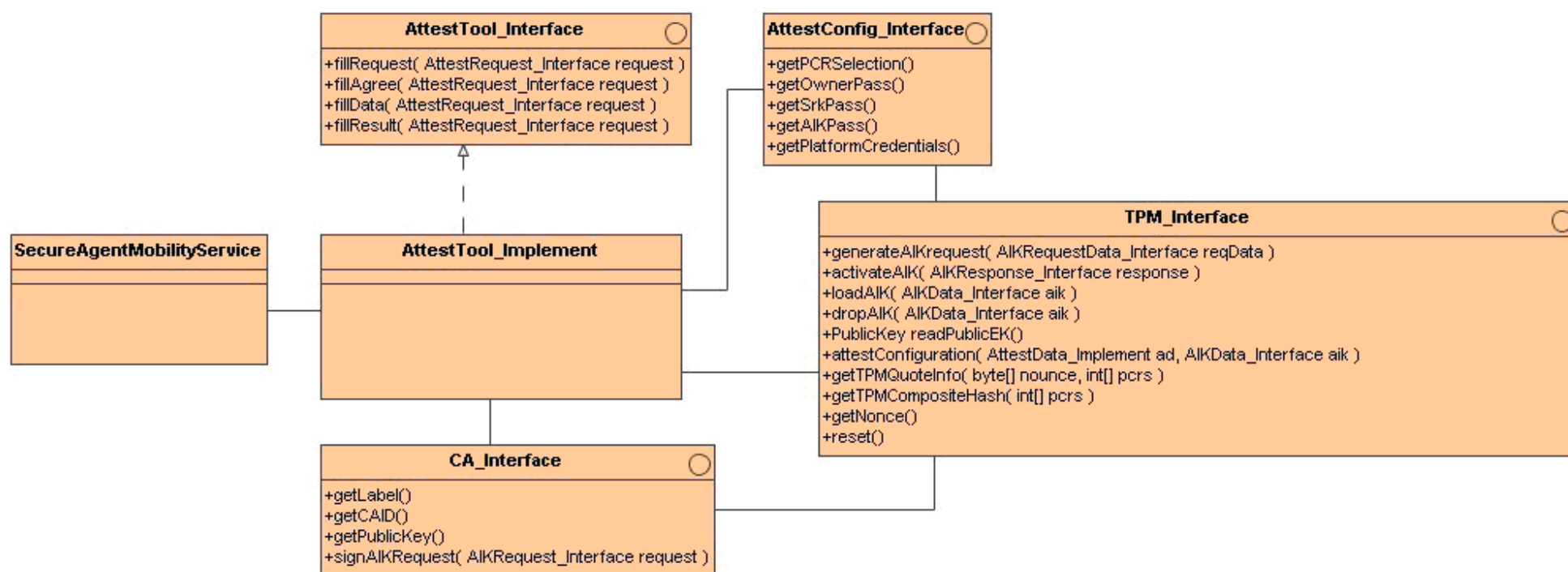


# User view



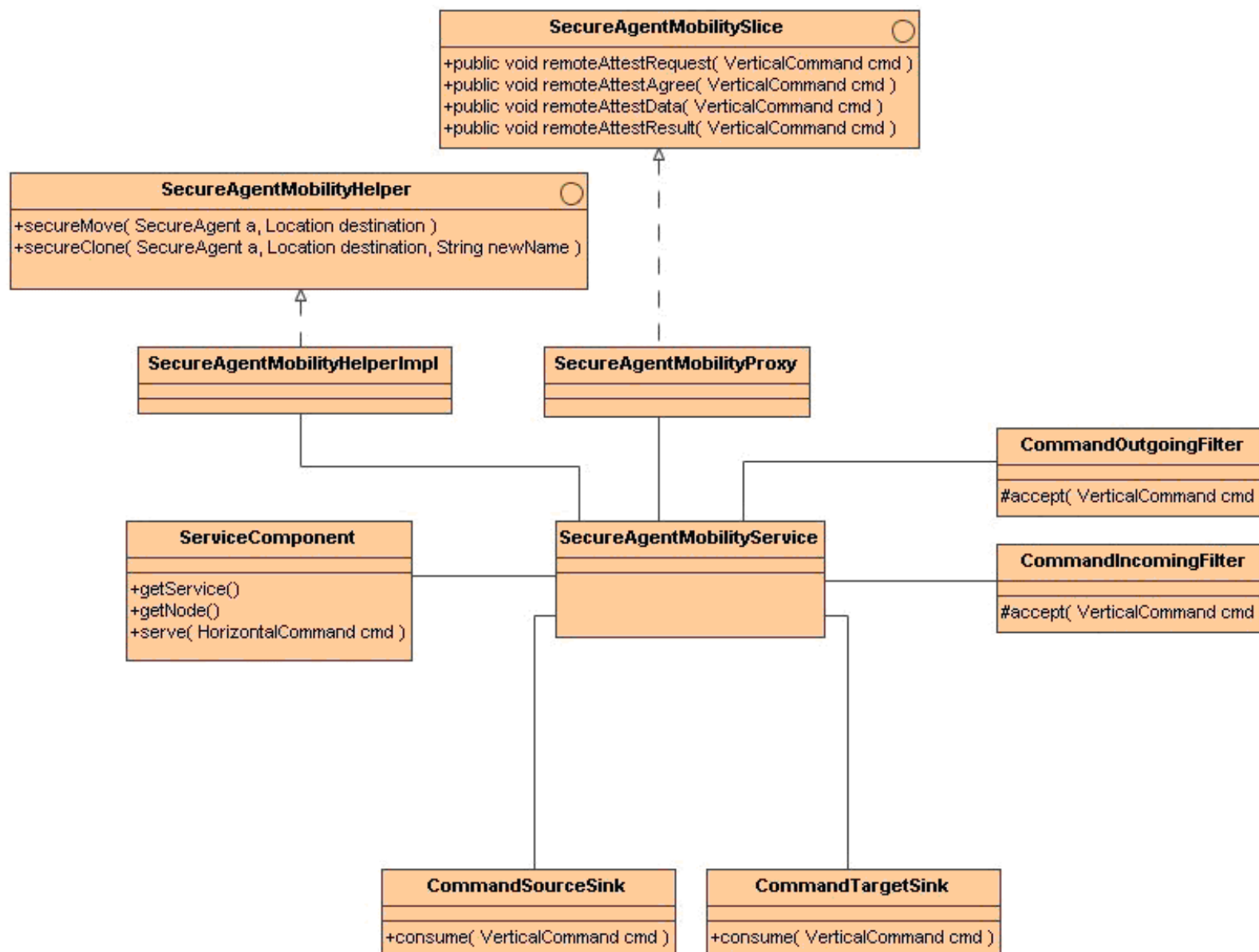


# Hiding TPM complexity





# Hiding TPM complexity for agent services







# Agents and Cloud computing

- Cloud computing will not reach all its potential until we can securely run our own software on it
- Running our own software in the cloud will present the same security challenges as running our agents in non-trusted agencies
- The presented solution can be a good basis for this purpose and we are currently addressing this problem in the PASSIVE European Project



## Conclusions and further work

- Security is essential for any practical computing system. Moreover, it is essential in agent based systems, ambient intelligence, embedded systems, cloud computing...
- Security must be easy to integrate for software developers
- Trusted computing can provide interesting tools to create security solutions for all these scenarios
- We have introduced the application of the presented system for cloud computing
- We are studying the ways to overcome the rigidity of the current model by using external attestation servers and semantic technologies



# Thanks! Any questions?





## Proof Carrying code

- **Is a general mechanism for verifying that the agent code can be executed in the host system in a secure way**
- **Every code fragment includes a detailed proof called code certificate (not to be confused with cryptographic certificates) that can be used to determine whether the security policy of the host is satisfied by the agent.**
- **As a technique designed for general mobile code, it is not difficult to apply it in agent-based technology.**
- **The use of this technique in agent systems allows agencies to verify the code certificates**
- **A combination with our TC approach is suitable**

# Protected Computing



- **Origin**

*Schaumüller-Bichl, I. and Piller, E.*

**“A Method of Software Protection Based on the Use of Smart Cards and Cryptographic Techniques” Eurocrypt’84.**  
1984.

- **Current status**

*Maña, A., López, J., Ortega, J., Pimentel, E., Troya, J.M.*

**“A Framework for Secure Execution of Software”**

*International Journal of Information Security,*

*Vol. 3, Issue 2, Springer-Verlag, 2004.*

*Maña, A., Muñoz, A.*

**“Mutual Protection for Multiagent Systems”**

*3<sup>rd</sup> Intl. Workshop on Safety and Security in Multiagent Systems (SASEMAS '06), Hakodate, Japan, 2006.*



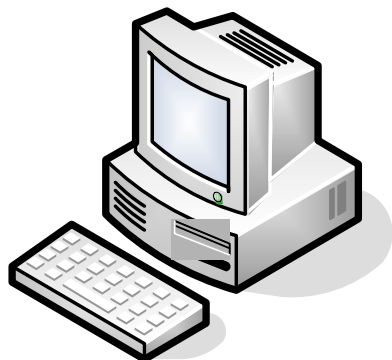
# Protected Computing

- Basis
  - Partitioning of the software elements into two or more parts. Some of the parts are executed in a secure processor, while others are executed in a normal (non-trusted) processor
  - A secure tamperproof coprocessor (not necessarily hardware) capable of executing code “on the fly” is required
  - The basic idea is to divide the application code into two mutually dependent parts.
    - The public part cannot be used to obtain the protected part
    - The communication trace between the parts cannot be used to obtain the protected part

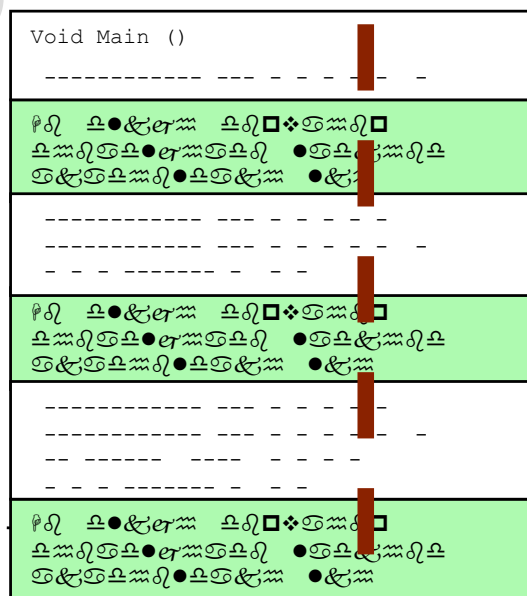


# Protected Computing

Untrusted



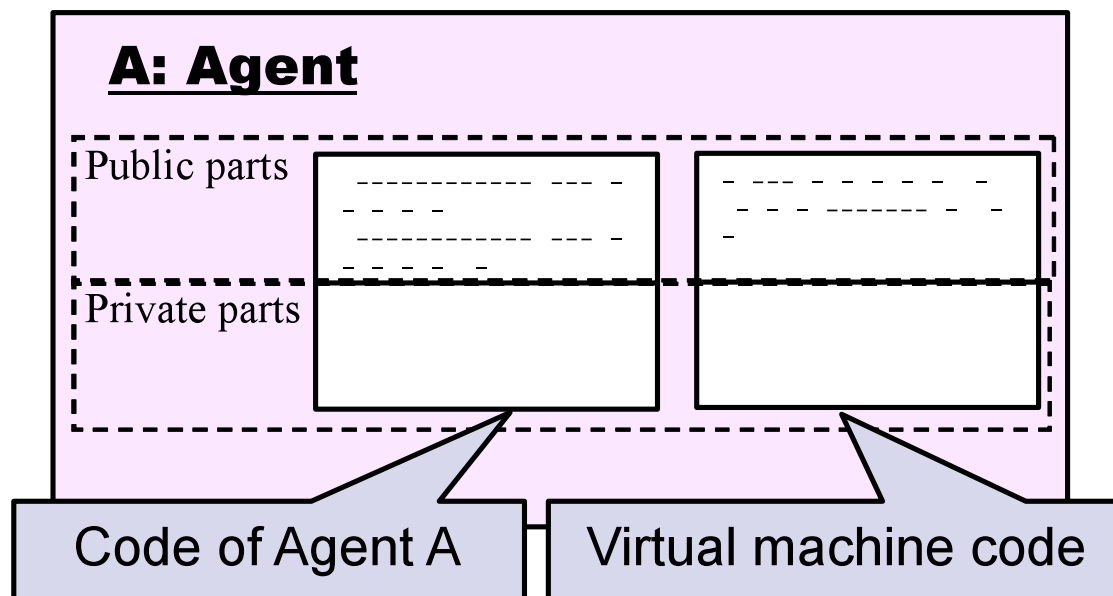
Trusted





# Protected Computing

- Dynamic Mutual Protection based on PC







# Protected Computing

- Main Advantages:
  - Independent *secure* applications are possible
  - Both applications and computer owners can control their security settings
  - Different secure coprocessors can be used (even simultaneously)
  - Mobile and replaceable devices such as Smart Cards can be used (*appropriate security*)
  - Low complexity and inexpensive solution
  - Different manufacturers (room for competition and market-driven security)
  - O.S. and HW platform agnostic