Architecture of the Secure Agent Infrastructure for Management of Crisis Situations

Branislav Šimo, Zoltán Balogh, Ondrej Habala, Ivana Budinská, Ladislav Hluchý

Institute of Informatics, Slovak Academy of Sciences,
Dúbravská cesta 9, 845 07 Bratislava, Slovakia
{branislav.simo, zoltan.balogh, ondrej.habala, ivana.budinska, ladislav.hluchy}@savba.sk

Abstract. *This paper describes the architecture and design of the secure agent infrastructure for management of crisis situations. The purpose of this infrastructure is semi-automatic control of the crisis management process and crisis management personnel support during a crisis (natural disasters or accidents). It focuses on information provisioning from human actors and management of resources needed for crisis mitigation and resolution. One of the key features is creation of a secure agent execution environment, which enables secure information exchange among trusted parties.*[1]

1 Introduction

During the crisis it is important to effectively manage distributed material and human resources needed for crisis mitigation. Timely information delivery about the crisis situation and fast resource deployment into the crisis area are essential for minimizing the losses. The information being collected can be very sensitive, because it can include private information about citizens affected, classified military data or other information that must be kept secret.

Effective distribution of material and human resources is one the aims of the EU FP7 project SECRICOM.

1.1 Security challenges

These problems mostly relate to distributed nature of computation execution and data processing. Many security problems are already solved such as secure communication tunneling through encryption or authorization and authentication using asymmetric cryptography. Security challenges for distributed computing can be generally divided into two groups: privacy and trust. Both of these security areas can be solved either on the side of clients (initiators) or on the side of executors (servers).

Communication, security and accessibility of information are the key factors during management of crises situations. Secure communication is a technological challenge which must be solved in a complex manner. It is important to solve interconnection of multiple communication channels but also protection from misuse of information and communication flows.

In this article we present architecture of a distributed system for secure execution of mobile code implemented as mobile services in untrustworthy computing environment using secure hardware platform module for the management of crises situations.

[1] This work was supported by the following projects: SECRICOM SEC-2007-4.2-04, SEMCO-WS APVV-0391-06, VEGA No. 2/6103/6, VEGA 2/7098/27.

2 Secure Agent Infrastructure Requirements

The role of agents in our system is primarily coordinated collection of information. Gathering of information is enacted either from legacy systems or from human end-users through mobile devices by guided dialog. In respect to requirements the overall agent infrastructure must be a secure, robust and fail resistant system. An agent as technology was selected due to the ability to fulfill such requirements through support of mobile and dynamically deployable executable code.

2.1 Infrastructure Security Requirements

In order to define concrete security requirements we must sketch the basic infrastructure in which agents will operate (Fig. 1). The network of Trusted Servers (TS) is the home platform for agents. According to [3] the platform from which an agent originates is referred to as the ***home platform***, and normally is the most trusted environment for an agent. This is also true for our agents – the network of TS is a managed set of systems with defined security policies and possibly managed by a central authority. From here agents are delegated to host platforms to gather data and information. TS host core services of the agent platform. Agents are mainly executed on remote sites which provide computational environment in which agents operate. We will refer to these sites as to ***host platforms*** *(or agent platform)*.

In general any party which wishes to join the system and to provide information from his legacy systems or users must introduce a host platform for agents. We will refer to such parties as to Host Platform Providers (HPP). From end-user requirements the following HPPs were identified so far (Fig. 1):

- *Resource Providers* – hospitals, fire brigade, police, warehouses or any other entities which can play a role in the mitigation of crisis situation,
- *Command Centers* – mobile (nomadic) centers which coordinate locally the incident site;
- *General Command Center* and *Operators* – usually located in one place or at least tightly interconnected.

The features of the agents will encompass several carefully chosen attributes:

- *Code mobility* (without execution state) – ability to move code to different platforms and execute there, within the project we do not plan to support execution state mobility (since there is no requirement for that),

- *Autonomy* – ability to deliver gathered data to one or several optional destinations,
- *Reactivity* – in some cases agents will perceive the context in which they operate and react to it appropriately (e.g., agents can monitor availability of some resource and notify the requestor).

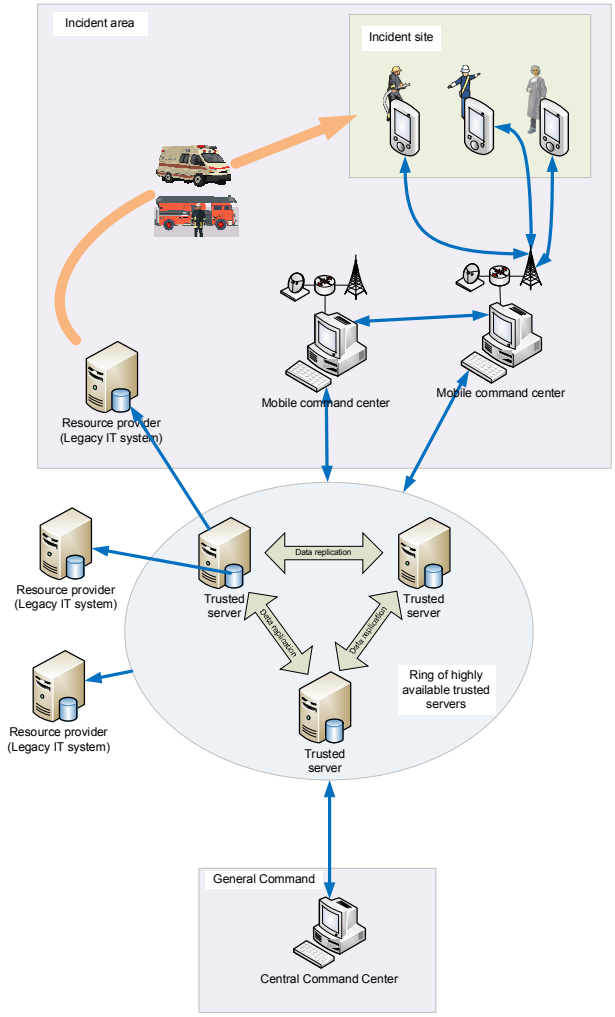

Fig. 1. Secure Agent Infrastructure deployment overview

Since agents collect information which is often classified, while at the same time requirements for action or decision traceability exist, agents must be provided with secure, trusted and attested execution environment. In the following we identify main agent-related security threats. A detailed explanation of generic mobile agent security aspects is discussed in [3]. Generally four threat categories are identified:

- Agent platform attacking an agent,
- Agent attacking an agent platform,
- Agent attacking another agent on the agent platform,
- Other entities attacking the agent system.

The last category covers the cases of an agent attacking an agent on another agent platform, and of an agent platform attacking another platform, since these attacks are primarily focused on the communications capability of the platform to exploit potential vulnerabilities. The last category also includes more conventional attacks against the underlying operating system of the agent platform.

2.1.1 The Host Platform Attacking the Agent

The main threat for agents in foreign execution environment of host platforms is the "malicious host problem". This is one of main problems in the class of "an agent platform attacking an agent". Simple explanation of "malicious host problem" is provided in [4]: "Once an agent has arrived at a host, little can be done to stop the host from treating the agent as it likes". Therefore the main requirements from the agent-side are lied out in respect to the "malicious host problem". The concrete security requirements of agents in respect to the host platform are therefore the following:

- Isolated execution environment for agent execution – not only virtual isolated execution environment but dedicated isolated hardware preferred;
- Means to attest the platform required in order to detect if the host platform is in trusted state;
- Protected storage for credential data (such as PKI's secret key).

2.1.2 The Agent Attacking the Host Platform

There are also threats stemming from an agent attacking an agent host platform. Therefore reversely a host platform has also requirements in respect to agents. These requirements are more evident when provided in context of HPPs security requirements:

1. HPPs do not want to install and execute any external application (including SECRICOM system) on their systems in line with their strategic legacy applications.
2. HPPs prefer to have a dedicated and isolated system for SECRICOM which would connect to their legacy system in a secure predefined way.
3. HPPs want to be able to control what (data), when and by who (traceability) is provided to the SECRICOM system.
4. HPPs want to be able to constrain the set of applications executable on their site. Agents must be therefore audited and verified, thus mediating trust to executable agent code.

The agent platform has the following security requirements in respect to agents:

- Isolated execution environment for agent execution - agents must be executed in isolated environment (isolated hardware preferred), so an agent can not harm legacy systems;
- Means to monitor and trace agents activity;
- Means to configure the set of agents executable on the host platform;

In order to track agents, any agent in the platform must be cryptographically signed. Only agents signed with

trusted authority and assigned to selected category will be trusted by a host system.

Agents need to send signed messages to Trusted Servers.

2.1.3 The Agent Attacking another Agent

It is required that any agent which will be used in SECRICOM will need to be audited and certified by a central authority. In turn every host platform will be configured to execute only agents which are certified. These two security policies should ensure that malicious agents will not be deployed into the infrastructure. Only breach of the set security policies might lead to potential agent-to-agent security risk.

Moreover each agent will be executed in a relatively isolated virtual environment with limited access to data of other parallel executed agents on the same host platform.

2.1.4 Other Entities Attacking the Agent System

Agents will also connect to legacy systems (third party software). Therefore a risk of attacking agent by a legacy system but also vice versa – risk of attacking legacy system by an agent exist.

The host platforms will need to provide some kind of connection to legacy systems. We explicitly presume that this will be a network connection. On any network connection there is an eavesdropping risk. Therefore another requirement which arises from agents to the host platform is:

- Secure protected connection to legacy systems.

Physical security of network connection can be achieved either by direct cable connection of the host platform with legacy system or by managed network security (managed switch with well defined security policies). The data transport security will be achieved primarily through encryption.

2.2 Agent Life Cycle and Related Security Requirements

The life cycle of an agent in the secure agent infrastructure (SAI) is the primary source of security requirements of the SAI. The creation of an agent encompasses development of its code, audit and certification. After successful certification of its code it is equipped with a private key, which is (for all of its existence) available only to the agent itself. A corresponding public key is stored and accessible in a public key registry. After its certification and "priming" with a private key, the agent is stored in an agent registry (AR). From this registry, it is downloaded to a trusted docking station (TDS or just DS) which needs to use the agent's capabilities. The downloaded copy has to be at all times secure – during transfer from the AR to the requesting TDS, and also during its deployment and execution inside the TDS. The copy inside TDS is destroyed when it finishes executing and delivers its results. The life cycle of an agent ends when its certificate expires, and after this it may be deleted from the AR, since it cannot be deployed anywhere in the SAI anymore. See Fig. 2 for a graphical representation of this process.

The life cycle of an agent poses following requirements on the security infrastructure:

- The agent must contain its private key; this key must not be known to any other entity during the whole lifetime of the agent.
- The agent must be audited before it can be used; the audit must ensure, that the agent does only what its creator states it should do, and that it does not contain any malicious code, which may jeopardize the integrity of the execution environment.
- The agent must be protected at all times from revealing its private key; it must always be either stored in a trusted device, or encrypted when it is outside of such device.
- Each audited agent must be issued a certificate, signed by its auditor, which states the capabilities of the agent as specified by its creator and verified by the auditor.
- The execution infrastructure must contain a service for storing and accessing certificates of entities inside the infrastructure; one class of these entities are also the software agents.
- All results produced by a software agent must be protected from being revealed to any entity different than their intended recipient – the client. Also, it must be asserted that their authenticity can be verified by the client upon their reception.
- The results provided by an agent must be signed by both the agent and the device running agent's code in order to ensure the trust of the results by Process Management System. Each secured device is supposed to be connected to secure docking module (SDM) providing the encryption keys authenticating the device and its user, respectively.

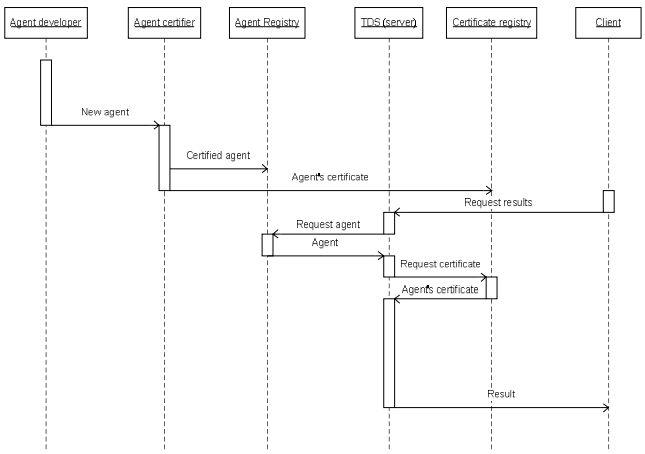

Fig. 2. Agent life cycle

3 Architecture

This section presents the architecture for systems which could profit from the combination of mobile code execution on an isolated trusted hardware connected to legacy computing resource. The architecture is designed for mobile services with agent-like features (mobility, pro-activity) which would execute on secure devices.

Such architecture in general consists of interconnected trusted (TS) and un-trusted servers (US). TS carry out the

following tasks: registry of services, users and modules, public encryption keys, the agent base (base of mobile code) or generic security politics. Each agent has features and "abilities", which are used for the enactment of certain processes. The enactment of processes is inspired by the domain of management of crises situations in which collection of information from multiple legacy environments is required. The whole process starts with the specification of a problem in the form of dialog. Further certain agent (service) will try to specify the most serious problem which was rendered by the crises situation. Based on the type of crises situation and the region where the crises has arose appropriate actions are initiated for each crises situation type.

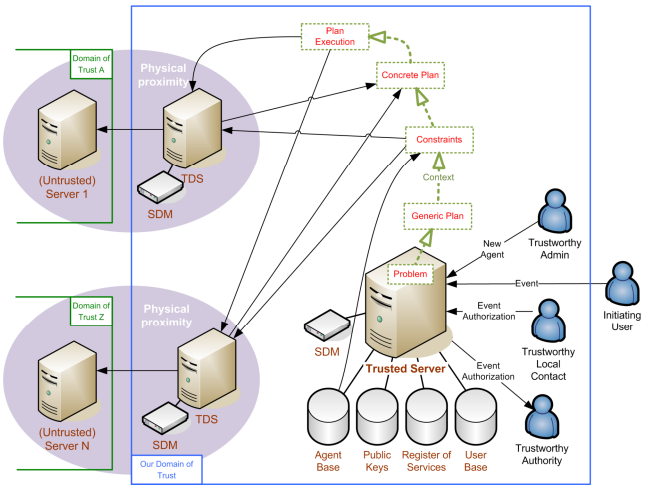

Fig. 3. The architecture for a distributed system for secure execution of agent-based mobile code based in an untrustworthy computing environment.

The system will semi-automatically generate plausible generic plans of possible solutions of rendered problems. In the next step the specification of context will be enacted in order to be able to generate the constraints of the crises situation. Relevant servers will be identified in the central database based on generated constraints. Agents which are able to query selected servers will be selected from the agent base. Information about available capacities will be retrieved from identified servers and sent back to central trusted server base. The system will then generate a concrete plan of crises situation resolution based on the retrieved disposable resource (human, material, etc.) capacities. The last step is execution of prepared plan for the concrete crises situation.

3.1 Agent Registry

Agent Registry (AR) is a service, which stores all the existing software agents in our infrastructure. The registry itself resides inside a TDS. The registry must ensure that any agent stored inside it is secure, and will be handled in a manner which will not reveal the secrets it contains to none but the authorized parties. AR has the following requirements on the security infrastructure of SECRICOM:

- Any request for a software agent to be downloaded from the registry and deployed inside a device must clearly state the recipient of the agent.
- All devices in the SECRICOM infrastructure (TDS and others) must be issued a certificate stating which agents it may receive; AR will reveal to a device only such agents, and will deny the deployment of agents for which the device is not certified.

The agent intended for deployment in a device must be protected during transport from being revealed to third parties; it must be encrypted in a manner which allows only the specific pair of a device and an agent, to which it is addressed, to be able to decrypt it and execute it.

4 Conclusion

In this paper we have described the architecture and design of the secure agent infrastructure for management of crisis situations. One of the key features is creation of a secure agent execution environment, which enables secure information exchange among trusted parties. We have also presented an architecture which is designed for execution of agent-like mobile code that executes on a secure trusted platform connected to legacy computational environment. The presented result is work in progress.

References

[1] Trusted Computing Group, URL: https://www.trustedcomputinggroup.org/home

[2] Trusted Platform Module, URL: https://www.trustedcomputinggroup.org/groups/tpm/Trusted_Platform_Module_Summary_04292008.pdf

[3] Wayne Jansen, Tom Karygiannis: Mobile Agent Security – NIST Special Publication 800-19. National Institute of Standards and Technology, Computer Security Division, Gaithersburg, MD 20899.Niklas

[4] Borselius: Mobile agent security, Electronics & Communication Engineering Journal, October 2002, Volume 14, no 5, IEE, London, UK, pp 211-218.

[5] Ana Ferreira, Ricardo Cruz-Correia, Luis Antunes, and David Chadvick: Access control: How can it improve patients' healthcare?, In: Studies in Health Technology and Informatics, 127, June 2007, http://www.cs.kent.ac.uk/pubs/2007/2625/content.pdf