Security Management in Distributed IT Applications

Ion IVAN, Mihai DOINEA, Sorin PAVEL, Sorin VINTURIS
Academy of Economic Studies, Bucharest, Romania
ionivan@ase.ro, mihai.doinea@ie.ase.ro, pavelsorin@yahoo.com, sorin.vinturis@yahoo.com

This paper presents the concept of distributed IT application in the context of knowledge society. Security requirements for data processing using very large databases are defined. Ways to optimize security for such applications are examined. Metrics for performance evaluation of security systems incorporated into distributed informatics applications are defined. The measurements are performed for the application analysis of structured text entities orthogonality.
Keywords: *security, optimization, performance, large data sets*

1 Distributed IT applications

Developed informational society presumes the implementation of applications for eGovernance, eBanking, eCommerce, eBusiness and eLearning. Distributed applications are characterized as follows:

- well-defined target group, whose elements are precisely quantified so that the specifications development, product design, testing and implementation includes features that provide guidance to the strict requirements of the group;
- processes are designed to benefit all target members by reducing waiting times, eliminating travel time, increasing resource diversity and increasing informational volume for training and decision making process;
- high level of stability by testing the continuity module since the specifications definition phase, requiring a minimum effort for adapting the user to access application resources and running without errors due to the incomplete or ambiguous messages provided by application, to successfully achieve the objective for which it is enabled by the distributed users.
- Distributed applications are classified according to different criteria, but the user-defined messages criteria divides applications as follows:
- applications with zero-length messages in which the user passes through the structure of distributed application by selecting links by keyboard or mouse; such applications correspond to the virtual museums, the presentation of personalities, cities, electronic books, electronic newspapers;
- applications with data input by values or words selection from a vocabulary defined using an easy selection method; this kind of applications are used to resolve problems characterized by a low volume of input data such weather forecast, calories requirements calculation, carry out multiple choice tests, shopping cart completion in online shops;
- applications with character strings input whose definition is based on procedures with a very low error probability, probability strictly dependent on how the strings are typed; they correspond to customer identification data entry in virtual shops, tax paying applications, all involving strings that define the name, age, address, identification number, e-mail, phone number, graduated school, occupation;
- applications in which the user must enter data by keyboard or data capture using special equipment to describe a particular situation, from which activities are defined and decisions are made; online accounting applications, telemedicine applications, customers online orders applications.
- It is noted that the development of the informational society is producing the following:

- reducing the number of activities that do not have a correspondent in distributed IT applications;
- encouragement through lowering the costs per user for everyone who benefits from the resource allocation by working online;
- creating a complete context for accessing nation-wide databases using multifunctional chips.

All these elements lead to a new approach in the software industry to achieve efficient distributed applications, for the user, the owner and the developer. Developer's profit depends on the application complexity but is directly affected by the corrective and maintenance processes. The application's profit depends strictly on the customers number, the transactions number and is affected by the compensations due to inadequate functioning.

2 Working with large data sets

Distributed applications are developed to meet the requirements of target groups with a large number of persons, e.g. the tax payment application is build at national level. The application for competence and education management is built at national level, too. The application for driver license testing is developed so that every citizen might take the exam, no matter the location. Another application built at a national level is the one that manages people entering or leaving the country. The target group for these applications is the whole population of Romania. The payroll and human resources software must be designed as unique distributed applications, in which resources are accessed by organizations themselves, gaining unique and effective procedures for correct problem solving.

Failure of meeting these requirements leads to the situation where the same problem has two completely different solutions using two different applications although the specifications are the same. Also the applications go through all stages of development cycle, including auditing and certification.

Only this kind of applications requires a framework with very large databases, because:

- very large databases operate with virtual databases obtained by local databases concatenation; small databases built following a unique template for public schools, are concatenated and lead to the database representing all pupils in gymnasium for the whole country;
- databases are continuously updated by adding information, with the exact identification of the moment, place, the operation and the person which generated the update process; this is the reason for the fast growing of data volumes which requires version management processes;
- the level of homogeneity in databases is extremely low because of the diversity of data: text, images from photography, images from scanning, animated sequences; the very large databases access procedures implies strings concatenation operations which identifies a homogenous collectivity or a single element within a collectivity;
- database are designed so that the target group elements access only the feature that defines the personal objective, and generate access procedures in other related databases in order to solve the problem;
- databases require access rights for application management so that the confidentiality terms are met.
- The viability of the very large datasets oriented software is ensured by:
- certifying initial data by target group members in order to insure that the start of application is based upon correct, complete and real data which meet the target group requirements;
- guaranteeing a way that makes the target group responsible for the update processes by using passwords and electronic signatures;

High-technology development in current society, citizen oriented software proliferation, along with new IT laws issuance, lead to production of software that

works with very large scale datasets:

- telecommunication operators records data about each call or message sent or received through the network, and keeps the information for a six-months period;
- internet and e-mail providers record data for each IP address in their own administration, about visited web pages, along with exact time of access, and also about every electronic message sent;
- public administration keeps payment history for millions of citizens;
- national providers of gas and electricity operate millions of bills annually;
- on-line search engines integrate content management of billions of websites.

These applications presume the existence of a large number of users, using 10^7~10^{10} data sets to manipulate and use. Distributed applications using large or medium large data sets must be prepared to work optimally with these amounts of data.

Depending on the degree of uniformity, the formula that shows the occupied space by an amount of data:

- in perfect uniformity - entities have the same number of properties of equal size:

$$VD = NE * NP * SP$$, where:

VD - volume of data measured in bytes;
NE - number of entities;
NP - number of properties of an entity;
SP - space of memory (measured in bytes) occupied by a property;

- if entities have the same properties but different sizes:

$$VD = NE * \sum_{j=1}^{NP} SP_j$$, where:

SP_j – memory space occupied by the *j* property;

- in minimum uniformity - entities have different number of properties and different sizes:

$$VD = \sum_{i=1}^{NE} \sum_{j=1}^{NP} SP_{ji}$$, where:

SP_{ji} – memory space occupied by the *j* property of the *i* entity.

Uniformed data sets with minimal loss of information involves:

- the intersection of data sets;
- the reunion of data sets with the default values of missing property set;
- file formats conversion;
- splitting of data volumes in uniform subdivisions;
- synchronization of applications.

The characteristics of applications that work with very large data sets are referring to:

- the acquisition and construction of the data sets;
- data volume stocking;
- the quality of data sets, the data inconsistency being controlled through a process of data quality management optimization;
- data sets management operations such as creating, updating, deleting, archiving, refreshing must be optimized so that large data sets that are not homogenous be processed in real-time; information browsing, classification and extraction from data is done through implementing specific algorithms, such as reduction algorithm presented in [5]:
    - using flexible key retrieval with multilevel filtering for easy selection of information;
    - statistical data processing to increase efficiency of operations.
    - Working with very large data sets involves exposure to a wide variety of security risks like:
    - security risks related to large volume of data entry per time unit, multiple sources of data, different ways and standards for acquisition;
    - integrity risks due to different levels of data quality, data sets uniformity and different using purposes;
    - data sets risk management due to the multiple classification possibilities and large hardware/software resources necessary for optimal handling of

large data sets.

According to [4], the structure of an application that works with very large data sets differs depending on the nature of data sets: word records databases, specific predefined object sets, multimedia files sets, documents databases; application purpose - acquisition, storage, administration, homogenate, using of data sets; the application's operation and usage environment – open, protected, closed; the software/hardware application development context.

The very large datasets oriented software architecture includes two main components:

- the data collection itself or the very large database
- specialized modules/units for:
  - data acquisition;
  - datasets homogenization;
  - datasets classification;
  - datasets selection;
  - datasets sorting;
  - datasets clusterization;
  - datasets export;

The organization and communication is illustrated in figure 1.

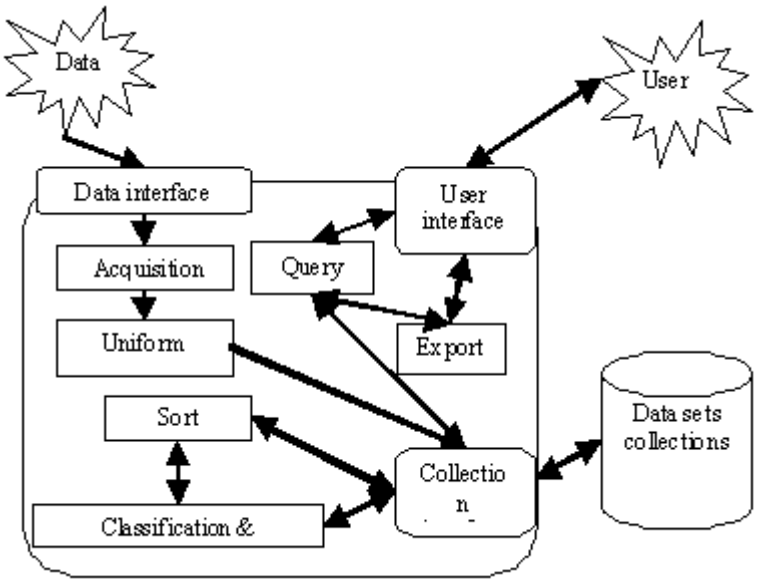

Fig. 1. General architecture of a distributed application - with very large data sets

The framed architecture components function in perfect synchronization, any bad functionality might lead to important loss. Giving this situation, any distributed application that works with very large datasets needs to assure data viability and security in each process featured.

3 Optimization

Each distributed application is designed as variants which differ from each other by the:

- technology being used;
- the resource consumption;
- the components quality level;
- degree of user satisfaction;
- cost of maintenance processes.

In this context, after analyzing each project variants is implemented a variant that is considered optimal in relation to one or more criteria. The security and reliability requirements are essential for distributed applications.

Each software application requires its own security approach. Each approach implies definition of indexes that measures the security level and permit election of the best one based on specific criteria.

Risk is defined as a threat that exploits the vulnerabilities in distributed systems. It considers the following categories of risks, classified by potential source:

- risks due to uncontrollable events like natural disasters, external agents;
- risks due to political or economic circumstances;
- risks arising as a result of technical or

technological problems;
- risks due to human behavior, lack of training of staff that works with distributed systems.

As defined in [1], vulnerability represents a weakness in the distributed applications through which available resources are unauthorized exploited. Vulnerabilities can be of different nature such as:
- hardware;
- software;
- human;
- communication;
- storage related.

If the optimization objective is to minimize the potential vulnerabilities of distributed applications, the optimization process is conducted by heuristic optimization scheme presented in [2], figure 2:

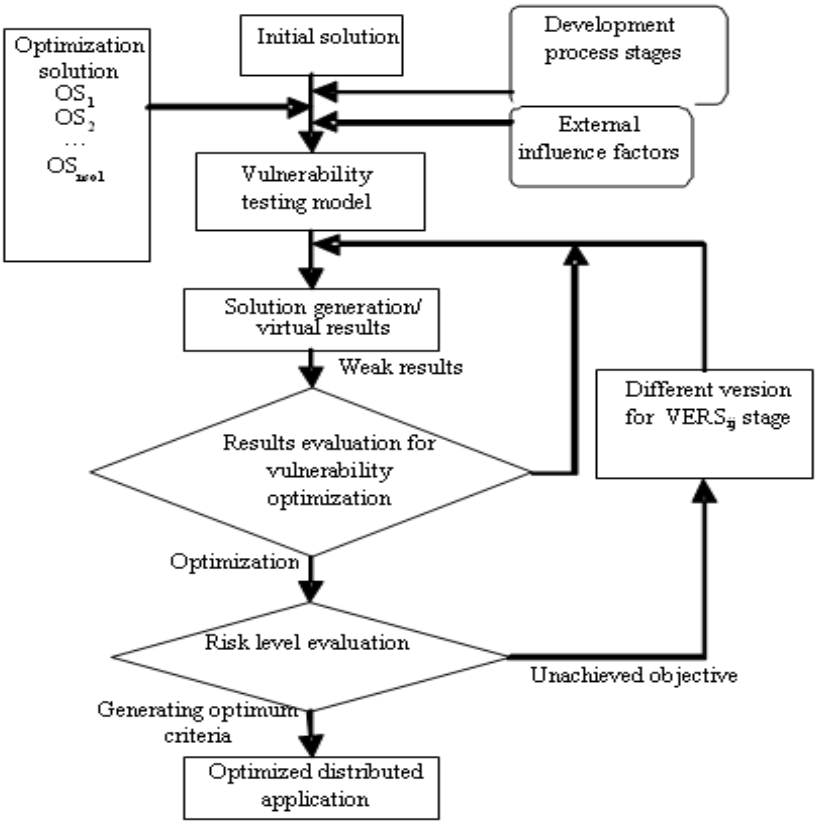

Fig. 2. Heuristic vulnerability optimization

Through automated data acquisition concerning the application behavior is created the basis for statistical processing of the extracted information on:
- the highest probability risk;
- the application procedures associated to the risk;
- times and costs involved to remove the case.

If application structural changes occurred, security optimization processes are resumed.

If the optimization objective is minimizing time spent on operations with very large datasets, then the process should face three challenges: data homogenization, data clusterization and data selection.

Data homogenization: while gathering the data, despite the nature of the data, its structure or even its manner of acquisition, it will always exist a percent of qualitative improper datasets: incomplete record attributes/fields, wrong tags, abnormal values or information impossible to render; the causes for these incoherencies can be:
- input human errors;
- incomplete data transmission;
- natural disasters/accidents;
- broken sensors;

In order to apply *data clusterization* algorithms for speeding selection processes, the data need to be structural and functional homogenized. The first step is completed by the data acquisition module/unit which ensures the same structure for all input datasets. Before entering the collection, the validation executes:

- filling incomplete fields with:
- default values;
- last records average – in proper cases;
- framing numerical values in pre-set ranges;
- verifying fields/attributes with pre-set format – e.g. e-mail address, PIN;
- parsing multimedia files and signaling errors;
- handling files depending on their extension;
- adding special fields/attributes/tags for later interpretation;

The application integrates automated methods of handling errors. If a certain error persists, the user is instantly asked about the problem value or warned about the incoherent dataset.

Dataset sorting and clusterization: in order to optimize the selection process, sorting and clusterization algorithms are implemented over the entire data collection; the objective of clusterization is forming physical or/and logical zones called clusters in which the datasets are structurally or functionally similar; the degree of similarity is given by the distance between datasets, which can be interpreted in many ways and can be computed in different forms; the applied clusterization algorithm depends on the data format and functionality:

- K-means algorithm;
- fuzzy algorithm;
- QT algorithm;
- graphs algorithm;

Sorting is run within a cluster having a certain pre-set or given criterion. The sorting unit is automated for optimizing selecting time.

Dataset selection: the unit/module takes the client/user data demand and searches within collection; the effects of all the other processing units are clearly visible at this level, when every set is:

- structural and functional complete;
- attached to a proper data collection;
- assembled in a well defined cluster;
- positioned within the cluster depending on its values;

In these circumstances, selection means running through the structural layers step by step and not covering the entire collection. This way, the time for operations is minimized and the accuracy of results is assured.

4 Performance evaluation

Software application performance must be evaluated given different perspectives. Form the owner point of view, a distributed application is effective if:

- the target group using the application and the real one is the same;
- the number of full usages for problem solving is high;
- administration and penalization costs are low;
- the working time without need of reengineering processes is the longest possible.
- From the user perspective, the distributed application is not effective unless:
- it covers the whole range of difficulties/problems the user deal with;
- the effort in user interaction is minimum;
- the success-usage rate is close to 100%;
- number of steps and selections for completing a successful workflow is smallest possible.

The effectiveness of software applications are perceived by the developer side during the construction process and especially while debugging the current use phase and during maintenance. The performance is as clearly visible as the developer takes care of the physical resources assigning and managing issues, depending on the needs.

Performance is a generic concept based on the quality of the application and, therefore, a precise system of quality characteristics.

Related to the security management, the quality characteristics of a distributed application are: reliability, portability, scalability and maintainability.

These features are distinguished by the user given the determinist and finite way of interaction. From the owner perspective, security effectiveness processes are quantified by:

- time of running without unwanted occurrences;
- cost of debugging due to attacks;
- level of resources occupied by operations needed for reconstructing databases.

From the developer perspective, security components of the distributed application are materialized by:

- resources taken for assuring continuity of running processes through observation;
- costs involved in debugging and saving intermediate stages, which keep the profit/loss ratio at an acceptable level.

Performance evaluation is made upon a list of objectives defined for a distributed application. The security objectives for a distributed application are categorized as being:

- preventive - objectives that prevent a threat from being carried out or limit the ways in which it can be carried out;
- detective - objectives that detect and monitor the occurrence of events;
- corrective - objectives that require actions in response to potential security violations, anomalies, or other undesirable events, in order to preserve or return to a secure state or limit any damage.

This strategy corresponds to the chronology of threat control measures and the corresponding priorities established for preventing security vulnerabilities. The chronology of threat control measures consists of:

- anticipate/prevent - threat types and sources are anticipated a priori, preventive action can be taken to reduce the likelihood of a threat being instantiated and the severity of its consequences;
- detect imminent known or suspected attacks, whether or not they are successful;
- characterize - attacks are defined so that appropriate short-term responses and long-term recovery actions can be formulated;
- respond - short-term responses are implemented to quickly isolate the consequences of threat instantiation;
- recover - long-term recovery measures are deployed to eliminate or mitigate the consequences of the same or similar threats in the future.
- For preventing security vulnerabilities, control measures have been established for:
- elimination of security vulnerabilities through ongoing security assurance activities, removing security vulnerabilities by the (re)specification, neutralizing security vulnerabilities by the (re)development of a resilient security architecture.
- minimization of vulnerabilities being exploited and the severity of the consequences from threat instantiation by designing and deploying a robust defense and in-depth security architecture.
- monitoring of any attempt to exploit residual vulnerabilities through ongoing diverse multisource monitoring, reporting, and analysis of anomalous security events and rapid preplanned informed responses to contain consequences.
- communication to end users, system administrators, and system owners about residual security vulnerabilities. These are accomplished by in depth training about the correct operation and use of a system and its security features, and comprehensive warnings about residual security vulnerabilities and consequences of misuse.

Security requirements implement the security objectives stated above. Like any other requirements specification, security requirements for distributed applications should be:

- correct and unambiguous expressed;
- complete and consistent, at the appropriate level of detail;
- ranked for importance and stability;
- verifiable and modifiable.
- They should be defined in parallel with the distributed application requirements. Security requirements should be expressed as:
- technical features – e.g., access controls;
- assurances – e.g. background checks for application developers;
- operational practices – e.g. awareness and training.

Security requirements generally include both requirements for the presence of desired behavior and requirements for the absence of undesired behavior. It is normally possible to demonstrate, by use or testing, the presence of the desired behavior.

The security requirements for distributed application are derived from a number of sources including:

- legislation and policy;
- applicable standards and guidelines;
- functional needs of distributed application;
- cost-benefit trade-offs.

Incomplete security requirements may expose an organization to loss or fraud; the specifications should include the following basic categories of security:

- authentication is used to verify the identity of users, devices, or other entity; authentication is accomplished using passwords and challenge-and-response protocols;
- confidentiality requires that the information is not disclosed to unauthorized persons, processes, or devices; confidentiality is accomplished through access controls, and protection of data through encryption techniques;
- data integrity requires that data is unchanged during the transfer from sender to receiver and have not been accidentally or maliciously modified, altered, or destroyed; data integrity is accomplished through checksum validation and digital signature schemes;
- non-repudiation is a method which guarantees that neither the sender nor the recipient can deny that the message exchange took place; this method requires strong authentication and data integrity, as well as verifying that the sender's identity is connected to the data that are being submitted;
- availability requires mechanisms to be in place to ensure that resources are available when requested and recovery mechanisms are effective when a failure occurs.

The selection of the security requirements depends on the type of protection needed, and is influenced by three key factors:

- sensitivity and value of the assets being protected;
- criticality of the mission the application performs;
- consequences from the assets or system being lost, compromised, misappropriated, corrupted, destroyed, misused, or rendered inoperable or unavailable for an extended period of time.

Beside these basic security requirements, a number of additional security requirements need to be taken into consideration while building distributed applications:

- authorization - refers to enforcing access control to ensure only authorized individuals/roles are allowed to execute tasks within a workflow by adhering to the workflow dependencies;
- separation of duty - these are additional constraints associated with the workflow to limit the abilities of agents to reduce the risk of fraud;
- delegation - refers to the delegation of authority to execute a task;
- conflict-of-interest - refers to preventing the flow of sensitive information among competing organizations participating in the workflow;
- safety analysis - refers to the analysis of studying the propagation of authorizations from the current state; this helps in answering questions such as

whether a user can gain access to execute a task.

Security requirements may suffer changes during time, triggered by different factors:

- identification of and response to new threats;
- changes into application security policies;
- changes in mission or intended use;
- new cost or schedule constraints;
- new technology.

For a formal testing of the security requirements and specifications, implemented in the distributed applications, security certification or accreditation are performed.

5 Result analysis

Given a homogenous collectivity formed by people with relative the same age, +/- 3 years, having the same level of education, that generates structured entities following a template. The software application analyzes generated text. At first, the user accesses the database, giving a string for authentication. Over 95% of the users complete the process within a specified operational time. 70% passed the phase of introducing structured entity components. 60% completed the whole steps regarding structured entity generation. Some of the steps allowed repetitive insertion in order to increase quality of structured entity, but only 45% got an aggregate level of quality above 0.78, thus leading to conclusion that the generated text entity is of good or very good quality. The files containing the digital format of structured entities are taken by a special module, and analyzed for substrings that could become potential sources for database attacks. In case of identifying this type of substrings, the specified segments are deleted from the file, affecting the quality of structured entity. In [6] are presented details concerning construction and restrictions applied on digital content in distributed applications.

To prevent attacks that could use the electronic mail component, the specified element was isolated by:

- physical separation from the rest of the components;
- automated message generation;
- message filtration by accepting only the ones found in the database, the application behaving as a closed system.

By isolating a component from the rest of the system, by means of strictly controlling the access to it, the risk is lowered and all characteristics of that application are preserved. This process is followed by an optimization, which, as shown in [3], involves a series of stages from evaluation to version construction, choosing optimal version and implementation. The result will consist in a higher percentage of users which will get an aggregate level of quality above 0.78.

6 Conclusions

The knowledge based society manifests a continuous development and needs secure software applications with citizen-oriented ergonomics. Distributed applications that fundaments the knowledge society needs to deliver quality characteristics along with a security level according with the threats provided by the network or the accidental or uncontrolled threats. Following the current trend of bursting usage of distributed software, security management will become a part of the development cycle along with risk management.

References

[1] I. Ivan and C. Toma, *Informatics Security HandBook*, Printing house ASE, Bucharest, 2006.

[2] I. Ivan, M. Doinea and D. Palaghiță**,** "Aspects Concerning the Optimization of Authentication Process for Distributed Applications," *Theoretical and Applied Economics*, Issue 6, June 2008, pp. 39 – 56.

[3] I. Ivan, C. Boja and D. Milodin, "Orthogonality level optimization for distributed informatics applications," *4th International Conference on Applied Statistics*, Bucharest, 20-22 November, 2008.

[4] C. Catal and B. Diri, "Investigating the effect of dataset size, metrics sets, and

feature selection techniques on software fault prediction problem," *Information Sciences*, Vol. 179, Issue 8, 29 March 2009, pp. 1040-1058.
[5] X. B. Li and V. S. Jacob, "Adaptive data reduction for large-scale transaction data," *European Journal of Operational Research*, Vol. 188, Issue 3, August 2008, pp. 910-924.
[6] S. J. Lin and D. C. Liu, "An incentive-based electronic payment scheme for digital content transactions over the Internet," *Journal of Network and Computer Applications*, Vol. 32, No. 3, pp. 589-598, 2009.
[7] M. Gertz and S. Jajodia, *Handbook of Database Security, Applications and Trends*, Springer Science, 2008.
[8] National Institute of Standards and Technology Administration, U.S. Department of Commerce, *An Introduction to Computer Security: The NIST Handbook*, Special Publication 800-12, 2005.
[9] A. Abbas, *Grid Computing - A Practical Guide to Technology and Applications*, Charles River Media, 2004.

Ion IVAN has graduated the Faculty of Economic Computation and Economic Cybernetics in 1970, he holds a PhD diploma in Economics from 1978 and he had gone through all didactic positions since 1970 when he joined the staff of the Bucharest Academy of Economic Studies, teaching assistant in 1970, senior lecturer in 1978, assistant professor in 1991 and full professor in 1993. Currently he is full Professor of Economic Informatics within the Department of Economic Informatics at Faculty of Cybernetics, Statistics and Economic Informatics from the Academy of Economic Studies. He is the author of more than 25 books and over 75 journal articles in the field of software quality management, software metrics and informatics audit.

Mihai DOINEA attended the Faculty of Economic Cybernetics, Statistics and Informatics of the Academy of Economic Studies, graduating in 2006, Computer Science specialization. Having a master degree in Informatics Security, promotion 2006-2008, he is currently a PhD candidate, Economics Informatics specialty in the same university, also teaching as assistant to Data Structure and Advanced Programming Languages disciplines. Following are the fields of interests in which he wrote a number of papers in collaboration or as single author concerning: security, distributed applications, e-business, security audit, databases, and security optimization.

Sorin Lucian PAVEL has graduated the Faculty of Economic Cybernetics, Statistics and Informatics from the Bucharest Academy of Economic Studies in 2008. He is currently following Master's in Software Project Management and the Doctoral School in Economic Informatics, both at the Academy of Economic Studies.

Sorin VÎNTURIŞ has graduated the Faculty of Automatics and Computer Science at the Faculty of Automatics and Computer Science in June 2007. In March 2009 he has graduated the Informatics Project Management Master, program organized by the Academy of Economic Studies of Bucharest. He is a PhD student at Doctoral School of Bucharest Academy of Economic Studies in the field of Economic Informatics.