Security Optimization for Distributed Applications Oriented on Very Large Data Sets

Mihai DOINEA[1], Sorin PAVEL[2]
[1]Amsterdam University, Information Management, The Netherlands
[2]Academy of Economic Studies, Economic Informatics, Bucharest, Romania
l.m.doinea@student.uva.nl, pavelsorin@gmail.com

The paper presents the main characteristics of applications which are working with very large data sets and the issues related to security. First section addresses the optimization process and how it is approached when dealing with security. The second section describes the concept of very large datasets management while in the third section the risks related are identified and classified. Finally, a security optimization schema is presented with a cost-efficiency analysis upon its feasibility. Conclusions are drawn and future approaches are identified.

***Keywords**: Security, Optimization, Very Large Data Sets, Distributed Applications*

1 Security Optimization

Optimization is always about resources. Optimization is necessary because of the intrinsic value of resources, limitation. Optimization can be applied to any kind of system, the only important aspects being the optimization criteria. Those are defined as the scope of the optimization process. If aimed values for the defined criteria were achieved, then an optimization process has been successfully undertaken.

When a defined criteria has linear solution, optimization is achieved by means of deterministic, heuristic, procedural or simulation algorithms, otherwise if the problem has a non-convex solution, genetic algorithms [1] are known to be more efficient, finding the solution by means of several mutation or cross-over operations in a predefined number of generations.

Optimization is the key to evolution. The optimization is the step through which the entire world and civilization reached the present state of evolution. Security optimization is part of the entire process of optimization for a distributed application. It deals with the security issues identified prior or after the distributed application was put into production.

The main security aspects which should be optimized for a distributed application, in order to behave properly and efficiently are those related with:

- communication between the layers of the distributed application;
- security processes viewed as resources consuming;
- the account's policy and how access privileges are managed;
- the authentication process as an important aspect for the entire security level.

When dealing with security, is essential that we take into account the optimization issues that may come along the way. If for each vulnerability identified in a security assessment process, a solution is implemented without actually trying to correlating them, then the level of security could be even lower than before, because of the possible mismatches between solutions that were found.

Optimization must take course for finding the optimum set of security solutions which must be implemented in order to cover all the vulnerabilities found in the assessment process without lowering the initial level of security and also trying as much as it is possible to minimize all the resources implied by this process.

Some of the criteria which can be approached when optimizing security in a distributed application are unfolded as follows:

- minimizing the time of encrypting sensitive information;
- minimizing the degree of memorability [2] [3] of passwords for authentication processes;

- minimizing the number of false positive results in authentication [4], if any;
- maximizing the degree of resources authenticity [5] for false claims identity avoidance;
- maximizing the degree of sensitive encrypted information which travels through the network;
- maximizing the number of source code attacks rejections.

If optimization is sawed as a stage in the existence of any kind of system, then could even be assimilated to one of the software life cycle stages from different perspectives.
From a marketing point of view the S-shape of software product life cycle [6] is depicted in figure 1, where the optimization can be implemented in one or more of the presented stages.

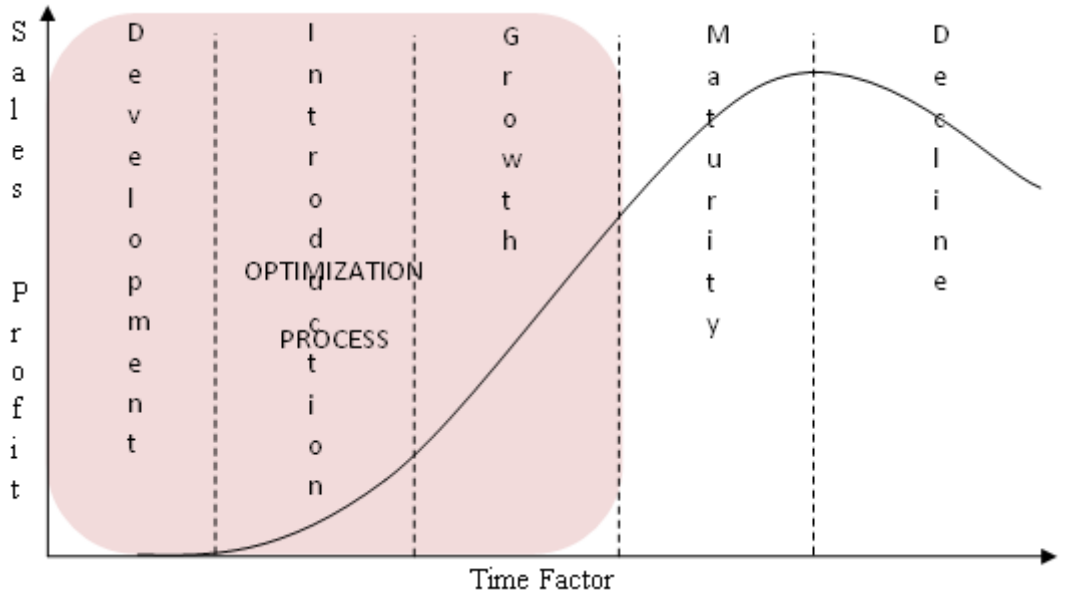

Fig. 1. Software product life cycle

Optimization comes here as a process of improvement and is necessary because, shortly after the introduction of the product on the market, an evaluation of the software impact upon users shows what need to be improved for the product to be more efficient. The optimization process is actually part of almost the whole period when the product is active on the market, until it reaches maturity, shortly before product decline, when no other improvements could be realized and the notion of optimization is changed into reengineering. Reengineering is the process of a complete change in the product design.
The optimization as part of the marketing life cycle product suits them as follows:

- development – optimization comes here only as a process of minimizing the resources needed for development and not as a feedback of users to the application's objectives;
- introduction – in this stage, shortly after the application is absorbed by the market, optimization is necessary for adding new users requests or improving the existing ones if they fail to satisfy the user's needs;
- growth – the optimization is present along the whole stage until the product reaches maturity;
- maturity – when maturity is achieved by a distributed application, all the optimization aspects were already approached and all that needs to be done is to wait for the moment off decline, when the market is saturated, and instead of optimization, a reengineering process should take place and the process life cycle starts all over again, but this time for another solution;
- decline – is the stage in which the sales are diminishing and the costs of maintaining the application in production or

on the market are getting higher than the profits that are made; in this point the application should be replaced by the new reengineered one or taken out.

From an information system point of view the classical life cycle of a software product have the following stages as presented also in [7]: requirements, analysis, design, construction, testing, installation, operation, maintenance, figure 2.

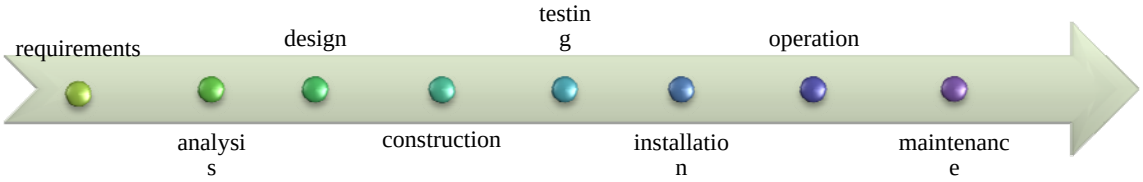

Fig. 2. Classic Product Life Cycle

Security optimization must be perceived as a must in all the stages involved in the software product life cycle. By optimizing security at each level, the following aspects should be analyzed:

- security needs should be identified to match the overall solution requirements; here the process of running through all the security concepts is made until the requirements are fulfilled based on the cost-efficiency criteria;
- in the stage of analysis, security components, previously identified, should be evaluated and the identified risks and vulnerabilities should be overlooked; security optimization here deals with finding optimal security solutions which should be implemented;
- design stage needs to carefully integrate security solutions as components which are interacting dynamically with the other elements of the system; optimization is made upon how does these security solutions interact with the overall system;
- when actually developing the system, security code optimization must be taken into consideration for avoiding as much as it is possible different code source hijacks;
- testing stage is all about verifying that the functional parameters defined in the specifications are covered by the application's functionality; this stage must address security optimization by trying to improve the efficiency of the selected security components;
- installation must have a clear and optimized procedure for implementing the security aspects in order to achieve the best level of protection and interaction;
- operation is the stage when the system is fully operational and may enter into production; at this stage, security optimization must take place in order to adjust the security parameters based on the actual user behavior;
- maintenance is the stage of adding new features or modifying the ones that already are, based on the new requirements; security optimization is described as the process of improving the general level of security by adding new security demands or by modifying the ones developed so far for getting much better scores.

Security optimization must be approached as a different but integrated part of the overall system optimization process. In order to achieve the maximum efficiency from the optimization process, a team of specialists on information systems security must define, analyze, evaluate, design, develop, test, implement and maintain the security components having at hand a well defined security policy.

2 Very Large Datasets Handling

Computerization of modern society, citizen-oriented software distribution and promulga-

tion of new IT laws has led to applications that work with large data sets:

- telecommunications operators record each call or message within the network for a period of six months;
- internet and e-mail providers record accessed sites for each IP address in its administration, together with the exact date of access and data about each email message;
- government keep track of different payments for millions of people;
- national providers of utilities – gas, electricity etc. – process hundreds of millions of annual consumer bills;
- online search engines integrate content management of billions of sites.

Situations mentioned above involve many simultaneous users, using very large data – $10^7 \div 10^{10}$ sets – and applications for data management. Due to the large quantities of data sets to be processed, applications acquire specific properties and functionalities.

Data collections represent, but are not limited to: databases, collections of text files/XML files/multimedia, data warehouse, or any combination thereof. Administration [8] requires specialized tools to harmonize the specific hardware and software aspects of large data sets.

The size or volume of database is quantitative expression of corporate data. In the practice of handling databases and files, or in human-computer interaction, data volume is understood as:

- length of a database file or the aggregate length of a collection of files in number of articles/records, or in physical space occupied expressed in bytes;
- number of documents placed in a file or database;
- number of transactions;
- processing time.

Each of these expressions is limited and reduces operationabiliy if it is singularly used. When volume data is expressed as a number of articles or records, information about the basic element is missing, namely the structure of the article or record. Thus, the database size is directly affected by:

- type and complexity of the information contained within a certain field;
- number of record features or fields;
- number of table records;
- number of database tables;
- number and complexity of integrity constraints within a table;
- number and complexity of relationships between tables;
- number and complexity of auxiliary files: indexes, stored procedures etc.

Although size escalate when it comes to very large data, the use of databases is the least costly in terms of space occupied, storing strictly numerical or alphanumeric data. Very large data sets require special treatment in terms of both their design and in the handling/use for problems solving. Design of very large data sets VLDS relates to the interaction with existing data, from preparation for collection operation to the storing for later use. Steps of the design process of VLDS are illustrated in figure 3.

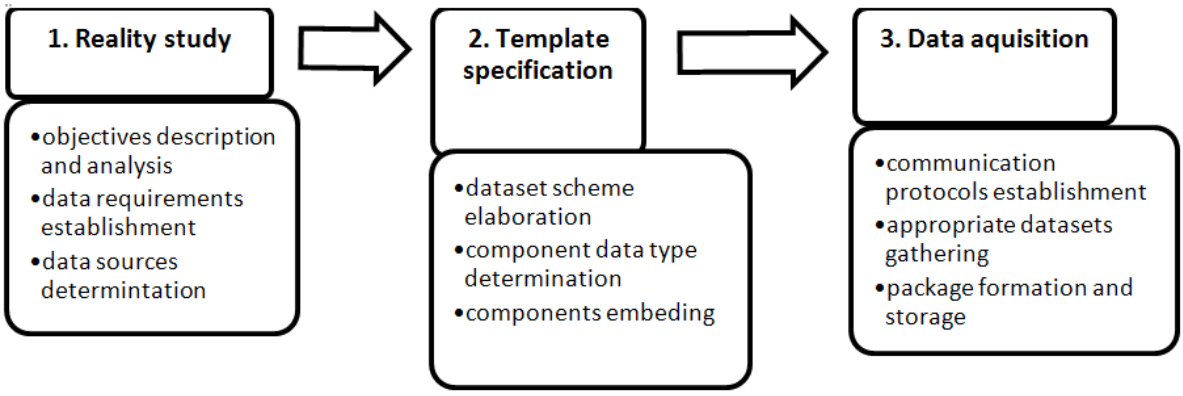

Fig. 3. VLDS design process

Design management requires a highly detailed and carefully documented approach, since VLDS will be affected both in shape and content by this phase. Neglecting the different stages or superficial treatment of any step of this process will lead to reloading data acquisition, which, given the very large size of the collection, may even lead to failure.

Very large data sets oriented applications interact with a wide and varied range of users. Therefore, its level of generality must be accepted by most users while the level of specialization must assure problem solution. At the same time, the application must always remain independent of users. Thinking, planning and designing [9] the application in this way involves the following features:

- presence or absence of users does not affect application operationability; the software product is available both for a large number of simultaneous users and in the situation of no user, without changing application functionality;
- user behavior does not affect the application structure; the degree to which the user understands or not, and use the application correctly or not, should only influence the quality of input respectively output data sets, not the operating process or the internal structure of the application;
- input does not affect processing flow; quality of data entered by the user only has repercussions on the solution offered; the application does not enter technological impasse due to lack of input parts or their incorrectness but continues processing and issues the output data, with warning of possible errors.

Using application and processing data should be transparent, so that the user is aware of its problem solving stage, knowing what data enters the process, what resources are available [10] and what processes are run at a time.

The application transparency is ensured by:

- visual selection of data entered by the user and labeling them as successfully retrieved;
- real-time messages about processing status;
- user guidance throughout the process of solving by indicating the current status, assessing the steps, the current step and/or the remaining steps to the solution;
- releasing a log-type file at the end, containing textual summary of the most relevant actions, together with related messages and solutions offered;
- granting possibility at any time to interrupt processing, to cancel the processing effects and to return to its original state.

The purpose of transparency is entrusting full control to the user which master the application actions so that no other processes interfere.

Operating processes [11] must allow users to visualize all input data prior to processing. The user verifies and is able to change, edit, detail and comment before submitting. Also, after loading the data, the application provides a short period of time for the user to return and edit entries and then retransmit. The sets are saved for future forwarding.

In case of multiple identical processes: identification data, account data, periodic processing options etc. the application contains interfaces with user-defined forms. This way, the client avoids placing the same data every time, when processes are similar. Payment orders within e-banking applications use the same work paradigm. Once completed, the orders are saved as a template and a set of preformed orders is build, which are used for various payments from the same account holder.

Also, when submitting online orders, the identification data of the recipients – delivery address, phone, e-mail – are stored with the first order and then made available for automatic completion to subsequent orders. Given the above principle, data is still editable with every command, allowing the client to change details of a particular command. Firstly, these features must be tested extensively [12] to eliminate the appearance of undesirable outcomes.

In carrying out any project, moments occur when activities and results no longer fall

within the parameters of the designed model. The reasons leading to such situations are some unexpected events that deviate the project from the planned course and therefore require special approaches. Hence, the design, development and implementation of projects should include treatment of uncertainty about the future.

3 Risks Evaluation in Managing Very Large Datasets

Risk is a measure of probability of occurrence and severity of effects of future events. It treats the matter from two perspectives:

- how likely are future events;
- how important are the consequences if it occur.

In software projects, [13] the risks diversify because of different components that enter into the development of applications: stuff, technology, equipment, methodologies, etc. Complexity of the topic makes the handling of risks to be integrated as part of project management, as risk management – MR.

Some risk models use qualitative indicators. Giving the case of a Web server that promotes a VLDS-oriented application any damage of the site is difficult to quantify as financial loss. In such cases involving "intangible assets" – like reputation - a qualitative estimation of risk is a more appropriate expression of loss.

Qualitative assessment is made on an ordinal scale, where the risks and impacts are non-numerically expressed. Risks are referred using expressions such as: rarely, occasionally, probably, frequently, and for impact: negligible, minor, major, critical, catastrophic. Probability matrix is constructed as Probability*Impact, combining the expressions for each risk probability and impact. The risk is determined as being high, moderate or low - figure 4.

Probability	Frequently					
	Probably					
	Occasionally					
	Rarely					
	Unlikely					
		Negligible	Minor	Major	Critical	Catastrophic
		Impact				

– low risk;
– moderate risk;
– high risk.

Fig. 4. The Probability * Impact Matrix

Treating issues of VLDS projects neglecting risks, is itself the highest risk which lead to interruption, abandonment or failure. Instead, focusing on risk generating items helps management to build methods – savings, provisions – to remove them, or at least to bring them to an acceptable level.

The multitude ways of approaching risks in large datasets oriented projects, requires classification having different criteria. Depending on the category to which it belongs, the risk is treated or enhanced, depending on its position within the project, risk affects the development plan.

Depending on their size, risks are divided into low, moderate and high risk. Within the very large datasets oriented applications, appear:

- small risk - loss of profit after application implementation and its launch to the public; the risk is low because the likelihood of this situation is poor due to feasibility study taken and its impact on the project itself is absent because it was already finished;
- moderate risk - functional requirements

are not expressed or explained, which affects the development cycle by replaying design, coding and testing; moderate risks delay the development process;
- high risk - project funding ceases due to changes in legislation or client's intentions; without adequate funding, a large project is likely to break and then be canceled.

Depending on the areas of risk event in the very large datasets oriented project, risks affect areas of: planning, budget, operational, technical, programmatic and security – figure 5:

Fig. 5. Risks of Very Large Datasets Oriented project

Planning risks arise when planning and scheduling is not addressed properly, or they are forced to change; in case of VLDS projects, such risks are reflected in:
- erroneous estimate of when to start or end the activities of sets of data acquisition, collection modeling etc.;
- failure to identify complex functionality – queries, indexes etc. - and the time required for their development;
- unexpected development scale project - combining several projects with large data sets into one;
- inadequate knowledge of processing large data sets technologies;
- incomplete specification of objectives for any project phase;
- lack of agreement or understanding between the client who wants the VLDS application and its developer;
- difficulty in implementing the various requirements on data sets.

Budget risks relate to poor financial planning, reflected in:
- wrong estimation of the expenditure categories - salaries, technology acquisition, data sets collection etc.;
- cost overruns of software modeling activities;
- replacement data handling technologies with more expensive ones;
- poor or inexistent monitoring of expenses;
- inadequate functioning of data sets storage instruments;

Operational risks refer to the process of project implementation and have human, system or external causes:
- failure in management priority conflicts: between the client and developer, between urgent and effective solutions, between quantity and quality of data sets;
- failure in the allocation and monitoring responsibilities within the team;

- insufficient staff, material or intangible resources: licenses, technology, operating systems for large data sets;
- lack of resource planning and assigning depending on implementation stages;
- lack of staff training in handling data sets;

Technical risks lead to failure of functionality and performance and are caused by:

- permanent changes of functional requirements, data sets purpose or their objectives;
- lack of available technology for handling large data sets or the existence of underdeveloped technologies;
- project is too complex to be implemented;
- difficult integration of project modules specialized in acquisition, storage, data sets modeling;
- inability of adopted technology to produce planned performance;

Coding risks reflect the software quality and have the following causes:

- inadequate documentation of application modules;
- lack of ability or skill of programmers in applications writing;
- lack of adequate testing of the implemented modules;
- occurrence of hardware failures;
- excessive complexity of application or data sets architecture;
- lack of responsibilities delegation between developers;
- lack of quality collaboration and communication between programmers;
- repeated changes in a project team, in technology, or the development environment.

Security risks relate to the data sets safety and their use in distributed approach is reflected in:

- lack of procedures to secure individual datasets, and also the collection;
- lack of methods for monitoring and control access to data sets;
- loss of data sets quality by successive modifications;
- vulnerabilities to cybernetic attacks.

These expectations and risk classifications don't avoid unexpected events, but encourage an informed handling of situations. Classifying risks determines first risk identification and then implementation of appropriate methods for treatment in their context.

The reason why the risks are included in the project management process is given by the impact that the specified events in the project. Between the two aspects of risk - probability and impact – the second one is most feared. It is classified as low risk, the event described by a high probability of occurring, but with negligible impact. On the other side, even with a low probability, an event with catastrophic impact is considered high risk. Experimental results relates to the effects of the events involved by the previously described risks. They are included in table 1.

Table 1. Hazardous events and their effects in VLDSO projects

Risk class	Events	Effects
Planning risks	Erroneous estimation of activity periods	Acquisition activities – of data sets, collection modeling etc. – takes longer than planned; delayed start time for other activities and hence delays in completion of the project; superficial treatment of the data sets quality due to shortage of time; time consumed in explaining additional features that were not present in the original require-
	Failure to identify complex functionalities and the time required for their development	
	Unexpected development of the project scope	
	Inadequate knowledge of current technologies	
	Incomplete specification of objectives for each phase of the project	
	Lack of understanding between customer and developer	

Risk class	Events	Effects
	Difficulties in implementing the various requirements on data sets;	ments; poor quality of the manufacturing processes of datasets.
Budget Risks	Erroneous estimates of expenditure categories	Failure to cover financial needs arising from activities such as data acquisition, data sets modeling, data storage; impossibility of demonstrating the of settlement costs; occurrence of unforeseen expenditures that deplete the financial resources and jeopardize the project.
	Activities cost overruns	
	Change to other, more expensive technologies	
	Poor or inexistent monitoring of expenditure	
	Malfunctions of instruments and their need for change	
Operational Risks	Failure of conflict management	The emergence of a hostile environment within the team, lack of concentration, lack of motivation, superficiality; poor quality in application implementation, data sets modeling; occurrence of redundancy in the module browser data sets; delays in the process because of the longer staff training for using data sets technology.
	Failure in allocation and monitoring responsibilities within the team	
	Insufficient human, material or immaterial resources	
	Lack of planning and allocation of resources in development stages	
	Inadequate preparation of staff in handling data sets	
	Lack of adequate communication between project team members	
Technical Risks	Permanent change of functional requirements	Occurrence of stress factors due to repeated changes; consumption of time using undeveloped technology; occurrence of failures due to inadequate integration of source modules.
	Lack of developed technology	
	Excessive complexity, which discourages the project implementation	
	Difficult integration of project modules	
Coding Risks	Inadequate modules documentation	Poor source code quality; neglect of data sets functionalities; errors of modeling sets, loss of data sets due to hardware; lack of continuity in programming style and functionality approach.
	Lack of programmers ability or skills	
	Lack of adequate modules testing	
	Emergence of hardware failures	
	Excessive architecture complexity	
	Lack of communication between developers	
	Repeated changes of members, or environmental technology development	
Security Risks	Access to all data sets	The data is viewed, stolen, and used for fraud Actual data are counterfeited Infecting computers that use or manage distributed application
	Duplication of data sets on network nodes	
	Altering data sets	
	Gathering malicious data sets	

Whatever the causes of undesirable events, risks must be treated properly and eventually

require finding ways to reduce the probability of occurrence and mitigate their impact on the very large datasets oriented project.

4 Optimization schema for password memorability

Distributed applications which are handling huge data sets are exposed to some vulnerabilities that are higher than usual. Just the fact that large amounts of data are flowing through the communication channels increases the likelihood of unauthorized data tampering.

Some of the security criteria presented in the first section needs to be carefully managed for not risking when dealing with this kind of applications. The following aspects should be taken into account:

- authorization is an important aspect which controls access to the resources by means of security access policies; the process of authorization should be rigorous before allowing users to reach critical resources;
- confidentiality must be implemented as a characteristic which doesn't allow unauthorized access on vital information; this is done by using cryptographic methods and must not be abusively used since the process of encryption and decryption could alter the system performances;
- the means of preventing information to be modified along the communication channel need to be efficient without accepting errors when using methods like: error detection codes for preventing data on being tampered along the process of communication by creating a cryptographic checksum on the transmitted data and passed it along with it.

Since the access to resources is very important we address this matter by proposing an authentication protocol which is meant to optimize the degree of password memorability [14] but in the same time to be efficient and lowering the level of possible false positive loggings.

In the literature, Knowledge Based Authentication, KBA [15] [16] is presented as a process which is using user's knowledge to give access to resources. The most important aspect is the personal approaching towards the user's personal characteristics that cannot be found to any other individuals. This approach is similar to biometric authentication which uses uniquely user's characteristics for allowing access as presented below:

- facial recognition [17];
- key stroked dynamics [18];
- iris recognition or eye tracking gestures [19];
- fingerprint match either by finger scan or photo [20].
- The main difference is that, when authenticating user's in distributed applications, the actual process must be going on the server, so, until reaches the stage of processing, user's login credentials are travelling throughout an unsecure area over the network. When sending biometric information over the network they could be intercepted and used for impersonating purposes to gain access to resources, so the uniqueness of such characteristics is at no use. The biometric authentication is only doable when the main process of providing access is unrolled on the local machine.

For this reason KBA method is relying on information known only by the authenticated user, being similar to biometric schemas, but as a process executed on the server side.

This paper approach is emerging from the necessity of using the power provided by the uniqueness of user's characteristics, like passphrases which only he can compute, based on its own mind process, things that no one can simulate, not even computers with their almost infinite computing power, and also by using different patterns that could reveal an impersonating attempt like:

- the number of user's repeated attempts before approval;
- the IP evaluation of user's attempts;
- the time interval frequently used for authentication;
- the user's own login patterns consisting in passphrases semantic analysis of word's probabilities, word's frequencies, number of different user's repeated at-

tempts from one single machine;

An Adaptive KBA protocol, AKBA, is proposed which will grant access based only on the unique characteristic of users and on the evaluation of login behavior that will not only improve the password memorability but will also increase the efficiency of the authentication protocol, being much harder to break as the passphrases will be hard to guess.

The AKBA protocol for allowing access will use a function as the one described above:

$$f(x) = y,$$
$$f: A \to B,$$
$$B = [0; 1],$$
$$x \epsilon A, y \epsilon B,$$

where:

f – AKBA function for evaluating authenticity;

A – the set of unique passphrases, factoids mutually independent, a user can generate;

B – the probability of user's legitimacy.

As biometric schemas allow access based on a personal user's characteristics, so the AKBA protocol will allow access based on randomly computed passphrases by the users, which should match although a pattern defined by the user on the creating account stage.

The role of this function is to compute a probability which further will be composed with other login characteristics probabilities, to tell whether the user attempt is a genuine one or not. Passphrases used in the login event, defined by $x \epsilon A$, are called factoids [15] composed by several different words which can be randomly generated by the user at each logon procedure and which must conform to a pattern previously defined when the account was created.

For being able to correctly authenticate users through AKBA protocol, a routine procedure on the user's login characteristics must be followed, resulting in a set of indicators composed by probabilities which will tell the probable nature of the login attempt. This procedure will consist in the following steps:

- storing login values on the server in case they are used for the first time;
- comparing the login values with the previously stored ones from the server;
- building a set of indicators and probabilities for the current login attempt;
- depending on the values revealed by the indicators three situation can occur:
  - the indicators aren't conclusive so the user must enroll in another authentication process just for certifying its identity;
  - the values prevent users to log in, resulting in another attempt for users;
  - positive match is found and the users can access all the application's resources put them at their disposal by the membership policy.

For the fact that the system cannot tell from the start which user is impersonating other ones identity, the only way of making the difference between them is by measuring the level of knowledge with which they are trying to access the system as presented in [15].

In the attempt of trying to hack the system, possible malicious users *j* must match the user's login behavior with respect to the time of access, the number of repeated failed login attempts, the locations used for accessing the account and others, more they have also to guess the factoids of that specific user and get a good score for the probability of hacking the AKBA function, which is the computed probability of guessing all of the factoids used in the login event:

$$P_{AKBA,j} = \prod_{i=1}^{n} p_{x_{ij}}$$

where:

$p_{x_{ij}}$- the probability of guessing factoid x_i by the user *j*.

The AKBA protocol is meant to improve the overall quality and efficiency of the login process by optimizing the following criteria:

- increase the memorability of users passwords;
- decrease the guessability of login credentials;
- ultimately, achieving user's patterns by permanent training, which are hard to

beat.

The process of validating user's login credentials is depicted in the following figure 6:

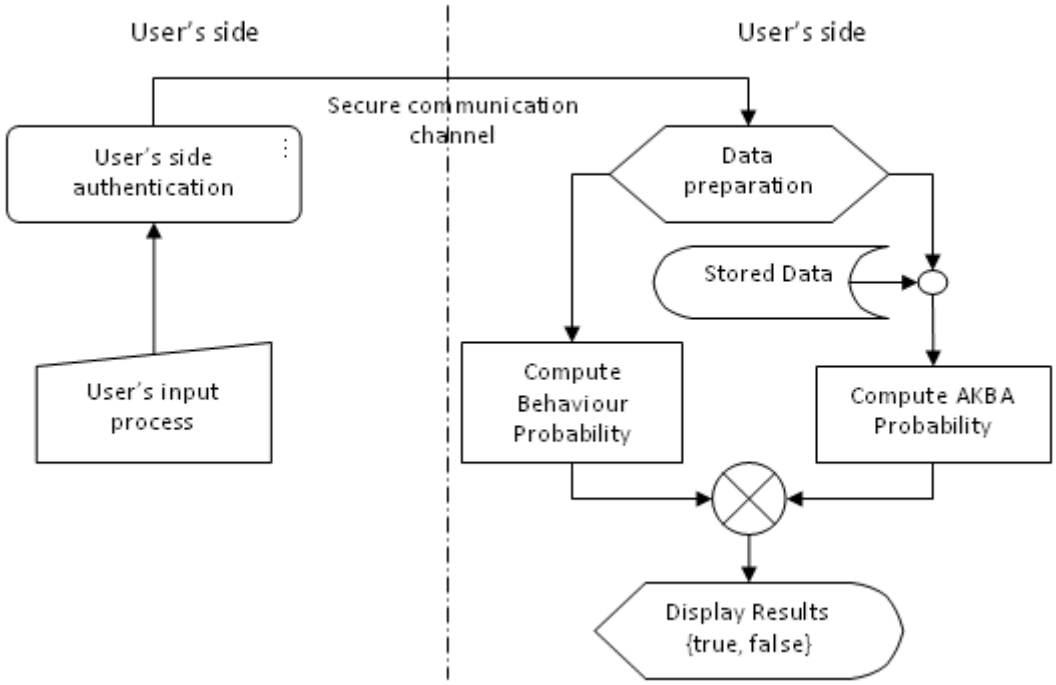

Fig. 6. AKBA Protocol Schema

Although the KBA is known for its possible false positive results, the adaptive process makes it even tougher to break. The main idea is that when users are trying to access the system, all the previous miss attempts or successful logins can be used for establishing the nature of the current attempt.

5 Conclusions

Finding ourselves in the full development of knowledge based society, each login momentum is decisive in whether or not to have access to resources. As the society is developing, so are the complex structures of information that are accumulating, and which we must carefully manage for not being in the situation when our entire ancestor heritage is lost due to some wrong used procedures.

Very large datasets are found throughout the main activity areas and many users are dealing with them, users that sometimes aren't even trained or familiar with the security issues that implicitly come along.

The trend of the actual society is to approach the Information Systems, as much as it is possible, to the current unprepared user. One of the ways of doing so is to improve the system's ability to facilitate the interaction with the user by two means: making the user to always be attracted and not lose the users' attention and by developing it in such a way that the system can be easily configured and adapted to users' needs. These two methods are some of the main system's purposes and are meant to increase the overall experiences of users when actually dealing with such systems. For this reason, all the back-stage processes must be efficient and rigorous attended to provide users with maximum satisfaction even when he or she isn't aware of the processes that are enrolling in the background.

Further studies upon the scalability and reliability of such system will be conducted and a feasibility analysis will reveal the opportunity given by this approach.

Acknowledgments

This article is a result of the project „Doctoral Programme and PhD Students in the education research and innovation triangle". This

project is co funded by European Social Fund through The Sectorial Operational Program for Human Resources Development 2007-2013, coordinated by The Bucharest Academy of Economic Studies (project no. 7832, "Doctoral Programme and PhD Students in the education research and innovation triangle, DOCECI").

References

[1] A. Kumar and M. K. Ghose, „Overview of Information Security Using Genetic Algorithms and Chaos," *Information Security Journal: A Global Perspective*, Vol. 18, No. 6, 2009, pp. 306-315.

[2] K. P. Vu, R. Proctor, A. Bhargav-Spantzel, B. L. Tai, J. Cook and E. Schultz, „Improving password security and memorability to protect personal and organizational information," *International Journal of Human-Computer Studies*, Vol. 65, No. 8, 2007, pp. 744-757.

[3] F. Steven, „An assessment of website pasword practices," *Computers & Security*, Vol. 26, No. 7-8, 2007, pp. 445-451.

[4] A. Mehler and S. Skiena, „Improving Usability Through Password-Corective Hashing," *Proceeding of the 13th International Conference of String Processing and Information Retrieval*, 2006, UK, pp. 193-204.

[5] W. Feng and Z. Q. Liu, „Region-Level Image Authentication Using Bayesian Structural Content Abstraction," *IEEE Transactions on Image Processing*, Vol. 17, No. 12, 2008, pp. 2413-2424.

[6] V. Cornescu, P. Marinescu, D. Curteanu and S. Toma, *Management: de la teorie la practica*, Universitatii Press, Bucharest, 2004, 304 pg.

[7] S. Cohen, D. Dori and U. De Haan, „A Software System Development Life Cycle Model for Improved Stakeholders' Communication and Collaboration," *Int. J. of Computers, Communications & Control*, Vol. 5, No. 1, 2010, pp. 20-41.

[8] I. Ivan and E. Dumitrascu, „Stable Structures for Distributed Applications," *Informatica Economica Journal*, Vol. XII, No. 1(45), 2008.

[9] S. Pavel, „Large datasets oriented software architecture," *Annual Conference of PhD Students in Economic Sciences*, Academy of Economic Studies, Bucharest, Romania, May 2009.

[10] D. Chen, R. Ewald, G. K. Theodoropoulos, R. Minson, T. Oguara, M. Lees, B. Logan and A. M. Uhrmacher, „Data access in distributed simulations of multi-agent systems," *Journal of Systems and Software*, Vol. 81, No. 12, December 2008, pp. 2345-2360.

[11] E. Borowski and H. Lenz, „Design Workflow System to Improve Data Quality Using Oracle Warehouse," *Journal of Applied Quantitative Methods*, Vol. 3, No. 3, September 2008.

[12] L. Ran, C. Dyreson, A. Andrews, R. Bryce and C. Mallery, „Building test cases and oracles to automate the testing of web database applications," *Information and Software Technology*, Vol. 51, No. 2, February 2009, pp. 460-477.

[13] I. Ivan, B. Vintila, D. Palaghita, S. Pavel and M. Doinea, „Risk Estimation Models In Distributed Informatics Applications," *Globalization and Higher Education in Economics and Business Administration GEBA 2009*, Iasi, România, October 2009.

[14] P. Ostojic and J. Phillips, „Memorability of alternative password systems," *International Journal of Patterns Recognition & Artificial Intelligence*, Vol. 23, No. 5, 2009, pp. 987-1004.

[15] Y. Chen and D. Liginlal, „Bayesian Networks for Knowledge-Based Authentication," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 19, No. 5, 2007, pp. 695-710.

[16] Y. Chen and D. Liginlal, „A maximum entropy approach to feature selection in knowledge-based authentication," *Decision Support Systems*, Vol. 46, No. 1, 2008, pp. 388-398.

[17] P. Dunphy and J. Yan, „Is FacePIN Secure and Usable," *Proceedings of the 3rd symposium on Usable privacy and security*, July 2007, USA.

[18] F. Bergadano, D. Gunetti and C. Picardi,

„User Authentication through Key Stroke Dynamics," *ACM Transactions on Information and System Security*, Vol. 5, No. 4, 2002, pp. 367-397.

[19] A. Luca, M. Denzel and H. Hussmann, „Look into my eyes!: can you guess my password," *Proceedings of the 5th Symposium on Usable Privacy and Security*, July, 2009, USA.

[20] B. Y. Hiew, A. B. J. Teoh and O. S. Yin, „A secure digital camera based fingerprint verification system," *Journal of Visual Communication and Image Representation*, Vol. 21, No. 3, 2010, pp. 219-231.

Mihai DOINEA received a PhD scholarship from the Academy of Economic Studies, Bucharest, Romania in Economic Informatics at the UvA Research Center. He has a master diploma in Informatics Security (2006). He is also a lecturer assistant and he teaches data structures and advanced programming languages at the Academy of Economic Studies. He published more than 20 articles in collaboration or as single author and co-published two books in his area of interest. His research interests are given as follows: informatics security, distributed applications, optimization criteria, databases, artificial intelligence, information management, security policies, mobile devices, networking and wireless communication.

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.