Annales UMCS Informatica AI X, 1 (2010) 131-144
DOI: 10.2478/v10065-010-0043-7

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI
http://www.annales.umcs.lublin.pl/

Middleware non-repudiation service for the data warehouse

Bogdan Księżopolski[1*], Zbigniew Kotulski[2, 3]

[1] *Institute of Computer Science, Maria Curie-Sklodowska University, pl. M. Curie-Sklodowskiej 5, 20-031 Lublin, Poland.*
[2] *Institute of Fundamental Technological Research of PAS, Swietokrzyska 21, 00-049 Warsaw, Poland.*
[3] *Institute of Telecommunications of WUT, Nowowiejska 15/19, 00-665 Warsaw, Poland.*

Abstract – Nowadays, storing the information is fundamental for the correct functioning of any organization. The critical factor is to guarantee the security of the stored data. In the traditional database systems the security requirements are limited to confidentiality, integrity, availability of the data and user authorization. The criticality of the database system and data repositories for modern business with the new requirements of law and governments, makes the development of new system architecture necessary which ensures sophisticated set of security services. In this paper we propose the database architecture that ensures the non-repudiation of the user queries and data warehouse actions. These security services are accomplished by means of the middleware layer in the data warehouse architecture.

1 Introduction

Nowadays, in the digital service era, increasing adoption of database systems and storing data have became a crucial issue for the public and private institutions. The huge amount of data is stored in the data repositories which have to manage a number of issues [1–6]. As the examples of these issues we can enumerate: management of data flow between the parts of the system (the client and the server of repository),

*bogdan.ksiezopolski@umcs.lublin.pl

balancing loading and management of assets, security of information, increasing system reliability and guaranteeing quality parameters. Among the enumerated problems one can emphasise information security issues. This aspect is especially important because in many cases protecting information which is stored in databases is crucial for companies.

The security of the data repository must meet three main security requirements [7]: confidentiality, integrity and availability. Confidentiality is related to the protection of data against unauthorized disclosure, integrity is related to the prevention against improper information modification and availability is related to the timely and reliable access to the database with prevention and recovery from hardware and software errors. Nowadays, these three main security requirements are guaranteed in all application environments. These security requirements can be accomplished by means of two layers, the middleware system layer and the database management system layer (DBMS).

In the DBMS the data confidentiality is ensured by means of the access control mechanism [7, 8]. This mechanism checks the rights of the user before granting the access to the objects in the database. After positive user authorization the data is transmited to the higher layer where the data is encrypted. Finally, the encrypted data is transmited to the user. Data integrity is guaranteed by means of the access control table mechanism and semantic integrity constraints. When the user wants to modify the data the access control mechanisms verify the user rights and after the positive verification grant the access to the data. The semantic correctness is verified by the set of the defined conditions. Data availability is ensured by means of the recovery subsystem and the concurrency control mechanism which corrects hardware and software failures.

Nowadays, data security requirements have evolved and the traditional ones are insufficient. The organizations store the data which are crucial for everyday operations and the access to them is needed in the real time. Furthermore, the organizations require more than storing the data [3–5, 7, 9], they need the adoption of solutions which ensure new requirements such as data quality and completeness, access control and privacy for mobile user, database survivability. These requirements can be guaranteed by means of various sets of security requirements [10–12]. While designing the database systems, we can take care of different extra security services [13] and fulfilling them is the challenge for today's systems [7]. Among them, we can enumerate non-repudiation [14]. The characteristics of this service are presented in Table 1.

In the article we would like to present the new data warehouse architecture which will guarantee the non-repudiation of user queries and actions performed by the data warehouse. The non-repudiation cannot be ensured by means of DBMS but it is possible to guarantee it by means of the middleware system layer. In this layer the security operations are proceeded before the data is transferred to the database and DBMS layer. The non-repudiation will be guaranteed between the users who make the operation in the data repository and the data warehouse. The important feature in the proposed architecture is that the non-repudiation service can be guaranteed not only

Table 1. Characteristics of non-repudiation.

Name of service	**Characteristics**
Non-repudiation of user queries	Non-repudiation of the fact of sending the query to the data warehouse by the user.
Non-repudiation of data warehouse actions	Non-repudiation of the fact of the users' queries accomplishment by the data warehouse

for all transmitted data but the particular records in the tables can be selected. For fulfilling this feature we would like to introduce the new data structure called Extra Security Label. Owing to this structure, the management of the non-repudiation service can be obtained. In the article we would like to present the schemes of data flow in the data warehouse which will ensure the non-repudiation security service.

2 Basic model of the data repository

The systems where the huge amount of data is stored, are organized into database repositories. Data repositories can be based on different architectures [1, 2]. The attempt at its standardizing takes the 'Storage System Standards Working Group', which works out the standards of storing data for local and distributed data warehouse [15].

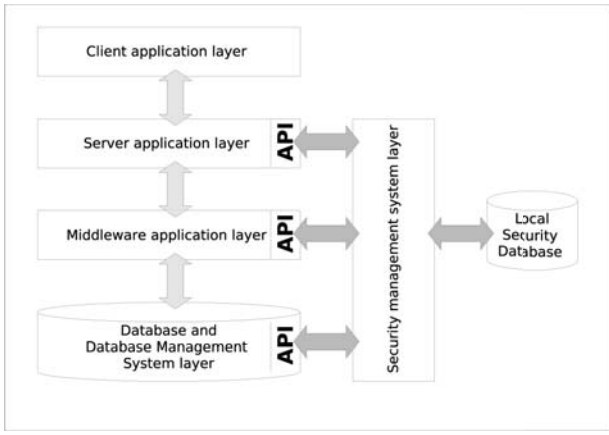

Fig. 1. The architecture of the data repository.

The model of data repository architecture based on the standard described by 'SSSWG' [15] and its simplest form described in the article is presented in Fig. 1.

The architecture includes the client site and the server site [16]. The client site can be limited just to the client application without any additional functionality and

then it is called a thin client. Another possibility is a rich client application which shifts resources and server machines onto the clients computers. Server sites consist of front-end software and back-end databases with the interface software linking the two (application program interfaces - API). In the front-end software one can enumerate: client application layer, server application layer, middleware layer, teleinformatic infrastructure layer and the security management system. The back-end databases consist of database management system and database layer. Below we present the description of the parts of the data repository architecture in detail.

Client application layer
Client application layer is responsible for an access to the data repository. By means of this layer clients can control all operations on data warehouse. The functions which are realized by that layer depend on the client type (thin or rich) [16]. If the architecture is based on the thin client, then that layer is limited just to the standard communication protocols and it can be ftp over ssh (sftp) or http over ssh (https) [12, 17–19]. Thanks to the usage of widespread communication canals we took care about common usage of data warehouse. Another possibility extends the functionality of the client site and then it is called the rich client. Then, advanced security functions are controlled by means of additional client applications [17, 19].

Server application layer
Client communication applications connect with the appropriate data repository servers. Those servers are the interfaces between the users and the back-end database layer. The security of that layer is very important because the vulnerabilities in the servers can lead to the attack on the middleware and the database layer. The complexity of the server layer depends on the number of possible communication canals which are accessible in the particular repository architecture.

Middleware layer
The issues of data repository management are accomplished in the middleware layer. The correct functioning of data warehouse mostly depends on that layer. As an example of tasks executed in the middleware layer we can enumerate: managing of the storing data for individual clients (privilege management, bandwitch management, management of access to the database), management of the access to the hardware which stores the data (defining for individual user the storing data hardware with optimal access time and capability, management of backup of databases). The management of the user and data security may be realized by that layer. In the situation when the security is the crucial aspect, the security operations should be realized by the individual layer for example, in the security management layer (Fig. 1).

Database and database management system layer
A DBMS is a complex set of software programs that controls the organization, storage,

management, and retrieval of data in a database. The DBMS accepts requests for data from higher layers and instructs the operating system to transfer the appropriate data. The DBMS includes: modelling language, data structure, database query language, transaction mechanism [16].

Security management system layer
The security management system layer can be present depending on the infrastructure. That layer controls the actions in the repository (meaning security which fulfils the security requirements for a given case). That layer cooperates with other layers and communicates with them by means of application program interfaces (APIs). The security management system is connected with local security databases which are used for storing crucial security information (keys, certificates, users, sessions).

2.1 Managing the security services - ESL

For managing the extra security services we would like to introduce a new structure to the data repository. The proposed structure is similar to the one in the Oracle database management system - version 9 [13, 20, 21] and based on the defining labels which are assigned to the rows and the users of the database. In these labels the management information about the extra security requirements will be stored. We will called it "Extra Security Label (ESL)". The system includes two types of Extra Security Label (ESL). The first one is General Extra Security Label (GESL) and contains the management information which refers to all rows in the table. The second one is the Record Extra Security Label (RESL) and contains the information which controls the extra security requirements for the individual rows in the table. The ESL consists of the components which are connected with every possible extra security service which can be ensured. In the article we limit the security service to non-repudiation. The structure of the table with Extra Security Labels which contains more than one security service is shown in Fig. 2.

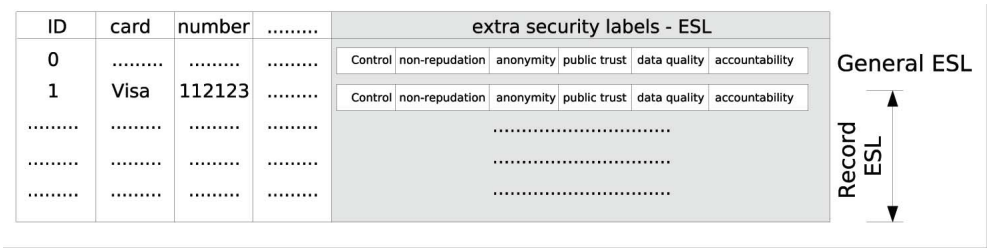

Fig. 2. The Extra Security Labels (ESLs).

The Extra Security Label consists of the following components:

- Control: these fields include the names of the security services required by the table or the record.
- Security service: these fields include the information needed for realization of a particular security service in the database system. The number of these components depends on the type of security services realized in the system.

2.2 The data flow and operations performed in the data repository

In this section we would like to present general data flow in the data repositories. This analysis is independent of the security services guaranteed by the repositories. The data flow in the repository can be divided into two groups of operations. These groups and steps are presented below:

Group 1 Access control to the data repository
Group 2 Realization of the user queries
Step 1 Access control to the specific data
Step 2 Management of the data
Step 3 Presenting the results of the queries

The first group of operations is the access control verification to the whole system of data warehouse. The positive authorization allows the user to take action in the data repository system.

The second group deals with making the queries to the specific data by the positively authorized user. This group can be divided into three steps. The first step is the access control to the specific data in the database. This access control mechanism refers to the authorization of the specific database and grants the privileges to the databases and to the specific tables. The second step is responsible for the management of the transfer data. The possible actions include not only operation on the databases but also the management issues (exp. settlement of the usage of data repository, payment options). In this step, the main operations of fulfilling the extra security services are realized. In the third step the results of previous operation are presented to the user. It could be an error following the negative authorization or the data as a response to the user query.

In the article we omit presentation of the data flow of all operations made by the data warehouse because of the page limit of the article. In the next sections we would like to focus on the data flow which guarantees non-repudiation service.

3 Extra security service: Non-repudiation

In Table 1 two types of non-repudiation are described. Data warehouse client can use a particular type of non-repudiation depending on the requirements of the specific tables into the database. In the scenarios described below we assume that the records in the

tables are configured. The record configuration is related to defining the Extra Security Labels. Another assumption in the scenarios concerns the public and secret keys of the user and the data warehouse. We assume that the data warehouse clients and the data warehouse have a secret key and the appropriate valid certificate authorized by one of the Certificate Authority. In the next sections we present the data flow of the non-repudiation of the user and the data repository actions.

3.1 Notation of parameters

In this section we would like to describe the notation of the parameters used in the article:
$ID_{operation}$ - the id of the data warehouse operation;
$query_{user_n}$ - the query of the n user;
$T(query_{user_n})$ - the timestamp of the query of the n user;
ID_{query} - the random number identifing the concrete query;
SK_{user_n} - the secret key of the n user;
PK_{user_n} - the public key of the n user;
SK_{dw} - the secret key of the data warehouse;
ID_{user_n} - the information about the n user;
$T(PK_{user_n})$ - the timestamp of the public key of the n user;
SK_{CA} - the secret key of the certificate authority;
$ID_{table_{service_n}}$ - the index to the table where the managment data of the conrete service for the n user is stored;
ID_{record_n} - the id of the n record on which the data warehouse made the actions;
R_{query} - the remarks to the query;
$T(ID_{query})$ - the timestamp made after realization of the query;
$ID_{operation_n}$ - the id of the n operations made by the data warehouse;
$T(query_{user_n}, ID_{query}, dw)$ - the timestamp of received user n query by the data warehouse (dw).

3.2 Non-repudiation of user queries

The non-repudiation of user queries means that every query made by a user is cryptographically registered by the repository. As the result of this non-repudiation service the user query to the data warehouse cannot be denied. Every time the data warehouse can verify the queries made by the user. In the architecture presented below this security service is fulfilled by means of the digitally signed information connected to specific user queries. The scheme of the non-repudiation of user queries is shown in Fig. 3. Below we describe that process.

(1) A user makes a query to the database and to the specific table in the database. This data includes: id of the made operation ($ID_{operation}$), the query ($query_{user_n}$), the timestamp of the query ($T(query_{user_n})$), the random

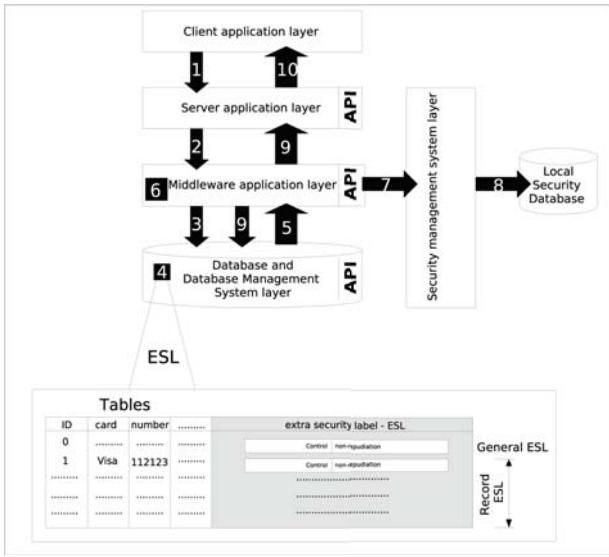

Fig. 3. The scenario of realization of the non-repudiation for user queries.

number which identifies a particular query (ID_{query}) and the certificate of the n user $((PK_{user_n}, ID_{user_n}, T(PK_{user_n}))_{SK_{CA}})$. This data is digitally signed by the secret key of the n user which receives the data:

$$(ID_{operation}, query_{user_n}, T(query_{user_n}, ID_{query})_{SK_{user_n}},$$

$$(PK_{user_n}, ID_{user_n}, T(PK_{user_n}))_{SK_{CA}}.$$

The query is sent by the client application layer directly to the server layer. After the access control mechanism the security of the transmitted data is defined [17].

(2) The server application layer receives the client data and transports it to the middleware application layer.

(3) The middleware application layer adds the time stamp of the received query ($T(query_{user_n}, ID_{query}, dw)$) and transports the user query to the database management system layer (DBMS).

(4) The DBMS layer receives the query and grants the appropriate privileges to the table for the user. After the positive verification the DBMS reads the GESL of the table and checks the control label where required security services are written. After that the DBMS reads in the GESL the metadata connected with the required service. This field includes the index ($ID_{table_{service_n}}$) to the data in the Local Security Database where detailed management data for the user n and the security service ($service$) are stored. In the next step the DBMS checks the RESL for every record refering to the user query. The control field in the RESL of the particular record includes the confirmation of

the required security services. Owing to the individual confirmations of the data one can select specific records for which extra security services will be required. In the last phase of this step the DBMS makes the user query to the appriopraite records in the database.

(5) The DBMS transports the data to the Middleware layer and the id of the records (ID_{record_n}) for which the non-repudiation of the receiver is required.

(6) The Middleware layer verifies the n user signature (SK_{user_n}) according to the data warehouse security policy (user certificate validation according to the time stamp added in step 3 - $T(query_{user_n}, ID_{query}, dw)$). It can be done by means of the OCSP protocol [22] or XAdES standard [23]. After the positive verification the Middleware layer creates the data which will be used during the verification of the non-repudiation of the n user queries. This data includes the information sent by the n user in the first step $((ID_{operation}, query_{user_n}, T(query_{user_n}, ID_{query}))_{SK_{user_n}}$, $(PK_{user_n}, ID_{user_n}, T(PK_{user_n}))_{SK_{CA}})$, the id of the records for which the non-repudiation service will be fulfilled (ID_{record_n}) and the index ($ID_{table_{service_n}}$) to the data in the Local Security Database where detailed management information of these security services is stored. The whole verification data include:

$$(ID_{operation}, query_{user_n}, T(query_{user_n}, ID_{query}))_{SK_{user_n}},$$

$$(PK_{user_n}, ID_{user_n}, T(PK_{user_n}))_{SK_{CA}},$$

$$ID_{record_n}, ID_{table_{service_n}}, T(query_{user_n}, ID_{query}, dw).$$

(7) The Middleware layer transmits the verification data to the Security Management System Layer (SMS).

(8) The SMS stores the verification data to the Local Security Database in the table which is indexed by the number gathered in step 4.

(9-10) The requested data in the n user query is transported to the n user.

3.3 Non-repudiation of data warehouse actions

The non-repudiation of data warehouse operations means that every action made by a client on the data warehouse is cryptographically registered by this client. The data warehouse cannot deny that some actions were performed by the repository. The user has cryptographically secure confirmation that stores, receives or modifies the data. This feature is obtained by the usage of the digital signature made by the data warehouse. The scheme of the presented type of non-repudiation is similar to that of non-repudation of user queries shown in Fig. 3. Below we describe this process.

(1-5) These steps are the same as in the non-repudiation of user queries.

(6) The Middleware layer creates the data which will be used during verification of the non-repudiation of the data warehouse actions. This data includes the part of the information sent by the n user in the first step

$((ID_{operation}, query_{user_n}, T(query_{user_n}, \ ID_{query}))_{SK_{user_n}})$, the id of the records for which the non-repudiation service will be fulfilled (ID_{record_n}), the timestamp of the realized query $(T(ID_{query})$, the id's of the n realized operations by the data warehouse $(ID_{operation_n})$, the timestamp of the received user n query $T(query_{user_n}, ID_{query}, dw)$ and optionally the remarks to the user query R_{query}. This information is digitally signed by the secret key of the data warehouse (SK_{dw}). The whole verification data include:

$$(ID_{operation}, query_{user_n}, T(query_{user_n}, ID_{query}))_{SK_{user_n}},$$

$$(ID_{record_n}, T(ID_{query}), ID_{operation_n}, R_{query},$$

$$T(query_{user_n}, ID_{query}, dw))_{SK_{dw}}.$$

(7-8) Steps 7 and 8 will be omitted in this scenario.

(9-10) The requested data in the n user query and the verification data prepared in step 6 are transported to the n user.

4 Security considerations

In the article the authors present the architecture which guarantees the non-repudiation security service in the storing systems. In this section we focus on the steps in which the crucial security data are created. These data are related to the information which is used during the non-repudiation verification process. The character of these data deals with the type of non-repudiation security service and the analysis is presented below.

4.1 Non-repudiation of user queries

The system data flow is presented in Fig. 3 and the crucial information is created in step 6. These data include:

$$(ID_{operation}, query_{user_n}, T(query_{user_n}, ID_{query}))_{SK_{user_n}},$$

$$(PK_{user_n}, ID_{user_n}, T((PK_{user_n}))_{SK_{CA}},$$

$$ID_{record_n}, ID_{table_{service_n}}, T(query_{user_n}, ID_{query}, dw).$$

Owing to the user certificate $((PK_{user_n}, ID_{user_n}, \ T(PK_{user_n}))_{SK_{CA}})$ the data repository can identify the particular user. The data warehouse stores the detailed information about the user query which is digitally signed by the n user's secret key (SK_{user_n}). These data include the specific user query $(query_{user_n})$, the id of the operation on the database $((ID_{operation})$, the time when the query was sent $(T(query_{user_n})$, the timestamp when the query was received by the middleware layer $(T(query_{user_n}, ID_{query}, dw))$ and the unique identification of the user query

(ID_{query}). This information can be used in case of the verification of non-repudiation of the user query.
Additionally, the data repository stores the id of the records (ID_{record_n}) on which the particular non-repudiation type was realized. Thanks to the individual selections of the table records different types of extra security service can be implemented. As an example one can show the non-repudiation of the user queries and the anonymity of the user. If the only possibility for realization of non-repudiation service was choosing the whole table in the database then it wouldn't be possible to realize the anonymity for a specific record in this table.
In step 3 the middleware layer adds the timestamp of the concrete query sent by the user $T(query_{user_n}, ID_{query}, dw)$. The timestamp is used in step 6 when the middleware layer validates the n user certificate. This process is based on checking the validity time of the n user certificate and the timestamp of the n user query. These actions are very important because the non-repudiation service can be guaranteed only for the user who has a valid certificate.
The last type of information which is stored by the data warehouse is the index to the table in the Local Security Database ($ID_{table_{service_n}}$) where the verification data for the n user and *service* are stored.

4.2 Non-repudiation of data warehouse actions

The system data flow is presented in Fig. 3 and the crucial information is created in step 6. These data include:

$$(ID_{operation}, query_{user_n}, T(query_{user_n}, ID_{query}))_{SK_{user_n}},$$

$$(ID_{record_n}, T(ID_{query}), ID_{operation_n}, R_{query},$$

$$T(query_{user_n}, ID_{query}, dw))_{SK_{dw}}.$$

The first part of information is the n user request for the data (($ID_{operation}$, $query_{user_n}, T(query_{user_n}, ID_{query}))_{SK_{user_n}}$). This information combines the user query with the verification data from the data warehouse.
The crucial operation which guarantees for the n user the non-repudiation of the data warehouse actions is the data warehouse digital signature (SK_{dw}) of the n user query information and the response information from the data repository. The response information from the data warehouse includes:

$$(ID_{record_n}, T(ID_{query}), ID_{operation_n}, R_{query},$$

$$T(query_{user_n}, ID_{query}, dw).$$

Thanks to the id of the records (ID_{record_n}) data warehouse informs the user about the records on which the non-repudiation service was realized. Another thing is the

timestamp ($T(ID_{query})$) which is generated after realization of the user query, thanks to which the user can read the time when the query was realized. The data repository sends the set of operations made on the records. This information is represented by the ($ID_{operation_n}$). Another information is the timestamp of the received query $T(query_{user_n}, ID_{query}, dw)$ which can be used as an argument in case of rejecting the user query because of not valid user certificate. Optionally, the data warehouse can have some remarks about the actions which are realized and it can be sent to the user (R_{query}).

5 Conclusions

Modern business requires a new system architecture which ensures a sophisticated set of security services. Besides the traditional ones which guarantee the confidentiality of the data, integrity of the data and authorization of the user one can enumerate extra security services. In this paper, we have presented the architecture of the database system which ensures the non-repudiation security service. In the article we have introduced the new structure of data which manages the usage of the extra security service in the system and is called Extra Security Label (ESL). This label is ready for realization in more than one security service and is designed for any extra security service. Thanks to the presented architecture one can ensure the non-repudiation security service in the data repositories. Fulfilling of this security service is crucial in many modern e-commerce and e-government applications [3, 6, 24, 25].

In the future we would like to design the architecture of extra security services [7] e.g.: anonymity, intellectual property rights (IPR), access control and privacy for mobile user, database survivability. Another important issue which has to be solved in the future work is how to combine all extra security services in one data warehouse. One of the future problems can be connected with the services which except another one, for example the non-repudiation of the sender and the anonymity of the sender. Another thing is the quality of the systems which ensure extra security services. Currently, we are implementing the data repositories where we ensure the non-repudiation security service. After accomplishing this task we would like to check the quality of the data repository where the new architecture will be used.

References

[1] Humm B., Wietek F., Architektur von data warehouses und business intelligence systemem, Informatyk-Spektrum 28 (2005): 3–14.

[2] Jarke M., Jeusfeld M.A., Quix Ch., Vassiliadis P., Architecture and quality in data warehouses: an extended repository approach, Information Systems 24 (1999): 229–253.

[3] Missier P., Lalk G., Verykios V.S., Grillo F., Lorusso T., Angeletti P., Improving data quality in practice: a case study in the italian public administration,

Distributed and Parallel Databases 13(2) (2003): 135–160.
[4] Mont C. M., Pearson S., An adaptive privacy management system for data repositories, Lecture Notes in Computer Science 3592 (2005): 236–245.
[5] Prakash N., Gosain A., An approach to engineering the requirements of data warehouses, Requirements Engineering, 13(1) (2008): 49–72.
[6] Villarroel R., Fernandez-Medina E., Piattini M., Secure information systems development – a survey and comparison, Computer & Security 24 (2005): 308–321.
[7] Bertino E., Sandhu R., Database security – concepts, aproaches, and challenges, IEEE Transactions on Dependable and Security Computing 2(1) (2005): 2–19.
[8] Bertino E., Jajodia S., Samarat P., Database security: research and practice, Information Systems 20(7) (1995): 537–556.
[9] Cui Y., Widom J., Lineage tracing for general data warehouse transformations, VLDB Journal 12(1) (2003): 41–58.
[10] B. Ksiezopolski, Z. Kotulski, Adaptable security mechanism for the dynamic environments, Computers & Security 26(3) (2007): 246–255.
[11] C. Lambrinoudakis, Gritzalis S., Dridi F., Pernul G., Security requirements for e-government services: a methodological approach for developing a common pki-based security policy, Computer Communication 26 (2003): 1873–1883.
[12] ISO/IEC19790, Security Techniques Security Requirements for Cryptographic Modules (2006).
[13] E. Fernandez-Medina and M. Piattini, Designinig secure databases, Information and Software Technology 47 (2005): 463–477.
[14] Devanbu P., Gertz M., Martel C., Stubblebine S., Authentic third-party data publication, Fourteenth IFIP Working Conference on Database Security (2000).
[15] SSSWG, Official web page of Storage System Standards Working Group (2002), http://www.ssswg.org/.
[16] Lothian P., Wenham P., Database security in a web environment, Information Security Technical Report 6(2) (2001): 12–20.
[17] Kaufman C., Perlman R., Speciner M., Database Security – Concepts, Aproaches, and Challenges (Prentice-Hall, 2002).
[18] ISO/IEC15408, Information Technology. Security Techniques Evaluation Criteria for IT Security (2003).
[19] ISO/IEC17799, Information Technology – Code of Practice for Information Security Management (2005).
[20] Levinger J., Oracle Label Security, Administrator's Guide (relase 2) (9.2) (2002).
[21] Samarati P., Vimercati S., Access control: policies models, and mechanisms, Foundations of Security Analysis and Design (2000).
[22] RFC2560, X.509 Internet Public Key Infrastructure, Online Certificate Status Protocol – OCSP (1999).
[23] ETSI101903, XML Advanced Electronic Signatures (XAdES) (2004).
[24] Ng H. S., Sim M.L., Tan C. M., Security issues of wireless sensor networks in healthcare applications h s. BT Technology Journal 24(2) (2006): 138–144.

[25] Thuraisingham B., Security and privacy for multimedia database management systems, Multimedia Tools Application 33 (2007): 13–29.