Workflow Based Security Incident Management

Meletis A. Belsis[1], Alkis Simitsis[2], and Stefanos Gritzalis[1]

[1] University of the Aegean,
Department of Information and Communication Systems Engineering,
Samos, Greece
meletis_belsis@yahoo.com, sgritz@aegean.gr
[2] National Technical University of Athens,
Department of Electrical and Computer Engineering,
Athens, Greece
asimi@dblab.ntua.gr

Abstract. Security incident management is one of the critical areas that offers valuable information to security experts, but still lacks much development. Currently, several security incident database models have been proposed and used. The discrepancies of such databases entail that worldwide incident information is stored in different formats and places and, so, do not provide any means for Computer Security Incident Response Teams (CSIRTs) collaboration. This paper presents an architecture based on advance database techniques, able to collect incident related information from different sources. Our framework enhances the incident management process by allowing the law enforcement units to (a) collect the required evidence from incident data that are spread through a number of different incident management systems; (b) transform, clean, and homogenize them; and, finally, (c) load them to a central database management system. Such architecture can also be beneficial by minimizing the mean time between the appearance of a new incident and its publication to the worldwide community.

1 Introduction

Today's incidents sought the need for collaboration. This can be easily proven when one thinks of the number of steps, actors and internet sites that modern security incidents use. Today's hacking community posses a huge number of tools that redirect hackers' IP packets, the method is called proxying, from a number of Internet sites in order to hide, their true identity. Along with this, experience hackers usually gain unauthorized access to unprotected computer terminals (e.g. usually university terminals) and use those to forward their attacks to their final target.

One requirement regarding the usability of an incident structure is cooperation. Most of the time fighting against incidents includes the cooperation of many different Computer Security Incident Response Teams (CSIRTs) around the globe. Such cooperation enables the tracking of the path an incident followed. This assists CSIRTs in identifying the potential originating source of the attack and even helps them to prosecute the perpetrator. Providing a structure and an underline system that allows the sharing of incident information in a secure and controlled way is a vital requirement.

P. Bozanis and E.N. Houstis (Eds.): PCI 2005, LNCS 3746, pp. 684 – 694, 2005.

Unfortunately, operating such a system is not likely using today's approaches. Modern incident management systems use different approaches and mechanisms to store incident related information (i.e. databases, text files, log files). Enterprises usually deploy their own internal incident management system and most public available CSIRTs use their own models and languages to describe and store incident related data. The semantics of each database differ substantially. The discrepancies of such databases entails that worldwide incident information is stored in different formats and places and, so, does not provide any means for CSIRTs collaboration. Unfortunately, providing a common incident structure is not an easy task. Such an observation is based on the fact that CSIRTs are built for different purposes (i.e., enterprise, department, or country level) and follow different social and technical rules.

At the moment, incident data collection and exchange is usually performed in semi-automated ways; e.g., telephone, email or transport of text files. Such techniques delay the process and usually affect the integrity of the incident data. Deploying a system able to automatically gather, and correlate incident information from a number of places is vital for the security of many organizations.

In this paper, we present a novel centralized incident management system based on advance Extraction Transformation Loading (ETL) database techniques. The proposed system provides a framework for CSIRT collaboration, by shaping the *balkanized model* that current approaches use into *federated model* on which each different incident management teams may hold their own databases, but the system is able to collect and correlate information stored on them. This is performed by allowing the fully automated extraction of incident data from different CSIRTs and storing these in a central incident database. Moreover, while the incident information is stored in a common format, it is possible to further manipulate it, in order to produce full incident descriptions.

In brief, the main contributions of this paper are as follows.

- We employ advance database techniques to tackle the problem of designing a centralized incident database management system.
- We identify the main problems that are underlying the population of a central incident database.
- We propose a method based on ETL workflows for the incremental maintenance of such a centralized database.
- We present a framework for incident correlation in order to keep track of a full attack that its component incidents are stored in different databases.

Outline. Section 2 presents the state of the art concerning incident databases. In section 3, we describe a method for the collection of incidents for different databases and their propagation to a central one. Section 4 presents the technology of Extraction-Transformation-Loading workflows. In Section 5, we describe the system architecture for the incident management. Finally, in section 6, we conclude our discussion with a prospect to the future.

2 Related Work

Currently, there are a number of research efforts that propose data models able to store information related to a security incident. These efforts have been developed and

used either for internal use by specific organizations or as a common centralized incident database solution. Example of the former efforts include the IBM's VULDA [1] incident database that has been developed and used exclusively by the IBM's Global Security Analysis Lab (GSAL), the IDB from the Ohio State University (OSU) [10] which stores only high level incident information that are based on the TCP/IP protocol, and the numerous databases that are developed to support the CSIRTs around the world (i.e. CERT/CC).

Along with the previous models there is a number of proposals that have been developed as worldwide centralized incident database solutions. The European project S2003 proposed a simple data model that can be used to build a library of security incidents [9]. This model can be used by European Computer Security Incident Response Teams (CSIRTs) as the means of storing data collected from security incidents.

The Incident Data Model [4] provides a centralized incident model that allows both managerial and technical incident related information to be stored while considering an incident a collection of steps.

Another paradigm of a vulnerability database is the Internet – Categorization of Attacks Toolkit (ICAT) developed by NIST [19]. The ICAT database is a Microsoft Access database that can be found on the Internet and includes a number of vulnerability descriptions.

Based on the second category the CERIAS Incident Response Database (CIRDB) [22] from Purdue University provides a database that can be either access on-line or be downloaded and installed for enterprise internal use.

One of the most interesting efforts is the Incident Object Description and Exchange Format (IODEF) [6] developed by the Incident Taxonomy and Description Working Group (TF-CSIRT) which is based on the Intrusion Detection Exchange Format Data Model (IDEFDM). The model was created to assist CSIRTs exchange incident data presented in an XML form. The work on this model has now been concluded and the results have been superseded by the FINE (Format of Incident Report Exchange) model, which is developed by the IETF INCH (Extended Incident Handling) working group [7, 5].

The Open Vulnerability and Assessment Language (OVAL) [21] provides a common ground on which vulnerability assessment tools can be developed. OVAL comprises three different XML schemas: (a) the System Characteristics Schema records the characteristics of a system; (b) the Definition Schema stores specific vulnerabilities, patches and compliance definitions; and (c) the Results Schema keeps track of the information produced by the vulnerability analysis. The OVAL community has already produced a Definition Interpreter which gathers the specific characteristics of a system and compares them against an OVAL Definition in order to produce a Result File that contains specific vulnerabilities of the aforementioned system. OVAL results are used to allow the exchange of the security related information among experts and system developers. Such a common understanding of system and vulnerability specifics enhance the sharing of security related information and accelerate the incident response process.

Unfortunately, the use of current designs, with possibly the exception of the IODEF and the OVAL, leaves little, if any opportunities for collaboration among different incident management teams. Clearly, the current luck of cooperation greatly

delays the process of identifying the perpetrators that organized an attack. The existent ways of performing such collaboration is using manual procedures, which inherit many problems. These are associated with the facts that not all people are native English speaking and that humans usually use different expressions to express the same incident. The integrity of an incident record could be greatly jeopardized by the previous facts.

3 Incident Collection

Collecting incident information from different databases includes providing solutions to a number of challenges. These challenges relate with several technicalities of a database system; e.g., the type and size of database fields, as well as, with further logic on deciding the incidents that can be grouped as steps in a common incident record.

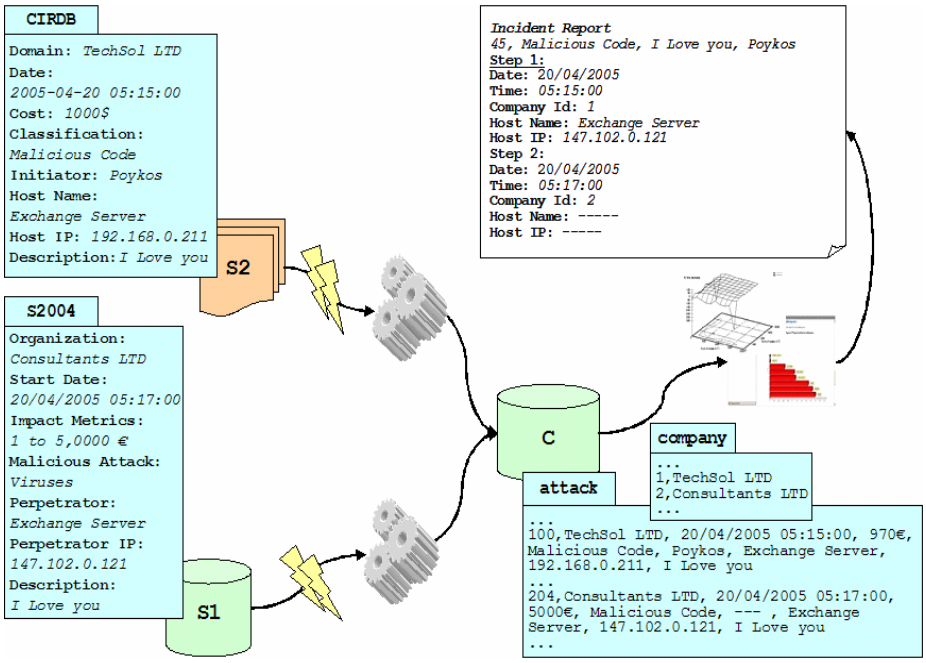

Fig. 1. Collecting and correlating incident information

To motivate our discussion consider the case depicted in Fig. 1 that involves incident records of two different incident management systems `S1` and `S2`, and a centralized incident database `C`. The first record stored in an incident database `S1`, is presented as it is recorded in the 'European S2004 Incident Database'. The second one stored in a flat file `S2`, is presented as it is recorded in the 'CIRDB' database. This scenario concerns the population of `C` by collecting data from both `S1` and `S2`.

The centralized database C contains homogenized incident information. Thus, it is eligible to apply further business analysis with data mining techniques in order to discover similarities and relations between the incident records.

In summary, by collecting, cleaning, transforming, and homogenizing distinct incident records from different incident sources, the system is able to identify that some of these are actually steps of the same incident. For instance, in Fig. 1 the correlation of the two source incident records results to the existence of an instance that involves a hacker nickname Poykos which emails an infected attachment with the I Love you virus to a company named TechSol LTD and more specific to the company's Exchange server with IP address 192.168.0.211. Then, this server retransmits the infected email two minutes later to a company named Consultants LTD. The overall cost of the incident is estimated up to 5970€.

A careful examination of such scenarios results in a list of problems that we have to deal with, in order to populate a centralized incident database. In terms of the transformation tasks, we can categorize the problem in two main classes of problems [17]: conflicts and problems at the schema level, and, data level transformations (i.e., at the instance level). In more detail, we have to tackle the following problems.

1. *Naming conflicts*. The same name is used for different objects (homonyms) or different names are used for the same object (synonyms). Observe in Fig. 1, that attributes Impact Metrics (S2004) and Cost (CIRDB) represent the same real world entity that describes the cost of a certain incident.
2. *Structural conflicts.* One must deal with different representations of the same object in different sources.
3. *Data formatting*. Equivalent data values are stored in different format into the incident databases. Observe in Fig. 1 the cost values: S2004 incident database stores costs as a range of values in Euro currency, while CIRDB stores costs as single values in Dollars.
4. *String Problems*. A major challenge is the cleaning and the homogenization of string data, e.g., data that stands for addresses, acronyms, names etc. Usually, the approaches for the solution of this problem include the application of regular expressions (e.g., using Perl programs) for the normalization of string data to a set of 'reference' values. For instance, consider the cases of 'Hewlett Packard', 'HP' or even 'Hioulet Pakard' where their respective value in the centralized database for all three cases can be 'HP'.

In addition, there are a lot of variations of data-level conflicts across sources: duplicated or contradicting records, different value representations, different interpretation of the values (e.g., measurement units Dollar vs. Euro), different aggregation levels (e.g., attacks per company department vs. attacks per company) or reference to different points in time (e.g. current attacks as of yesterday for source 1 vs. as of last week for source 2). The list is enriched by low-level technical problems like data type conversions, applying format masks, assigning fields to a sequence number, substituting constants, setting values to NULL or DEFAULT based on a condition, or using simple SQL operators; e.g., UPPER, TRUNC, and SUBSTR.

Moreover, a crucial issue is that the population of the centralized database should be executed in an incremental fashion. Obviously, the time window for the population of the centralized database is rather too small to repeat the same job more than once.

Thus, instead of extracting, transforming, and loading the same data, we choose to employ this procedure only on those incident records that have been changed during the last execution of the process. So, this means that we are interested only to the (a) newly inserted, (b) updated, and (c) deleted incident data.

In order to deal with such problems, we use advance database techniques. The Extraction-Transformation-Loading (ETL) workflows can be used to facilitate the population of a centralized incident database from several different incident databases. Thus, before proceeding to the presentation of system architecture that concretely deals with the aforementioned problems, we present in brief the technology of the ETL workflows.

4 Extraction – Transformation – Loading (ETL) Workflows

The integration of data from several sources to a centralized database management system is a well-studied subject in the field of databases [17]. The practice in real-world environments has shown that the integration problem is more complicated and involves complex operational processes, in order to clean, homogenize and customize the data as the management requirements demand. Recently, both researchers [11, 24, 25, 28] and practitioners [16] have started to study the problem of the collection of data from several sources (e.g., databases, flat files, web), their transformation and cleaning, and finally, their loading to a central database called *Data Warehouse* (DW) in order to facilitate business analysis in large organizations.

These operational processes normally compose a labor intensive workflow and constitute an integral part of the back-stage of data warehouse architectures, where the collection, extraction, cleaning, transformation, and transport of data takes place, in order to populate the central database. To deal with this workflow specialized tools are already available in the market, under the general title ETL tools [13, 14, 18, 20].

Extraction-Transformation-Loading (ETL) tools are pieces of software responsible for the extraction of data from several sources, their cleansing, their customization, their transformation in order to fit business needs, and finally, their loading into a central database. To give a general idea of the functionality of these tools we mention their most prominent tasks, which include: (a) the *identification* of relevant information at the source side; (b) the *extraction* of this information; (c) the *transportation* of this information to the Data Staging Area (DSA), where all the transformations take place; (d) the *transformation*, (i.e., customization and integration) of the information coming from multiple sources into a common format; (e) the *cleaning* of the resulting data set, on the basis of database and business rules; and (f) the *propagation* and *loading* of the data to the central data warehouse.

In Fig.2, we abstractly describe the general framework for ETL workflows. In the left side, we can observe the original data stores (Sources) that are involved in the overall process. Typically, data sources are relational databases and files. The data from these sources are extracted by specialized routines or tools, which provide either complete snapshots or differentials of the data sources. Then, these data are propagated to the data staging area (DSA) where they are transformed and cleaned before being loaded into the data warehouse. Intermediate results, again in the form of (mostly) files or relational tables are part of the data staging area. The central database

DW is depicted in the right part of Fig. 2 and comprises the target data stores. The loading of the central warehouse is performed from the loading activities depicted in the right side before the DW data store.

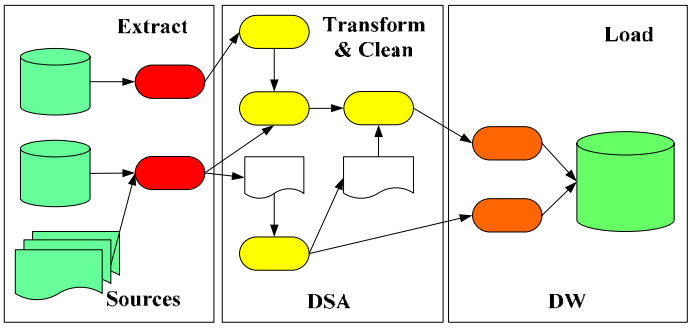

Fig. 2. The environment of Extraction-Transformation-Loading workflows

Several research efforts stress the fact that ETL workflows can be used to an area broader than just the data warehousing. Specifically, in [25, 26, 27] the authors describe ETL workflows as a generic framework, called ARKTOS II, capable to represent any process that can be modeled as an ETL workflow, through a customizable and extensible model that comprises a design tool, template transformations, a metadata repository, and an optimizer. In this work, we built upon the state of the art concerning ETL workflows. We reuse the modeling constructs of ARKTOS II upon which we subsequently proceed to build our contribution.

5 System Architecture

The management system proposed is based on the Common Object Request Broker Architecture (CORBA) architecture. Currently a number of research publications have proposed the use of CORBA to access databases [3, 2, 8]. Using CORBA the system allows for a number of client applications to access the incident database. The ability of adding new services and allow clients to dynamically discover them using the CORBA's DII is crucial.

A vital requirement for any such incident management system is security [12]. Such incident information could become a valuable tool for adversaries all over the world. For the current stage of this research, it has been agreed that only law enforcement units will be able to access the incident database. In later stages, the system will became available to the public in a way that does not sacrifice the confidentiality or the integrity of the incident data.

The system provides access control and encryption using the SSL/TLS protocol. TLS is selected because it is an already widely deployed security protocol and the computing community already has a number of implementations. The protocol in conjunction with client and server side certificates allows for strong access control,

confidentiality and integrity to be applied. Along with this there are a number of the CORBA Security implementations based on the SSL/TLS protocol.

Every CSIRT wishing to participate must obtain a digital certificate. Both client and server side certificate are exchanged to force authentication and access control. The CSIRTs certificates (Fig.3) are used to ensure that data are collected only from trusted sources and thus the integrity and non repudiation properties are ensured. During the initiation of the ETL process a handshake agreement is performed. The handshake is performed using the digital certificates enabling the authentication and key exchange process work. The incident data are then encrypted/decrypted using a session key (Skey). The X.509 certificates include the access rights that the holder posses on the incident data.

Registered law enforcement organisations can contact the DBMS system to access full incident records stored in the database, by sending their digital certificates (Fig.3). This can be performed through a WEB based interface or through custom developed clients. To enable access through the WEB, law enforcement units need to download a java client from the system's WEB site. The client will then perform CORBA's structured requests to the Incident management system. The process of accessing the system is described below:

1. *Web Browser downloads HTML page*. In this case, the page includes references to embedded java applets.
2. *Web browser retrieves Java applet from HTTP server*. The Http server retrieves the applet and downloads it to the browser in the form of byte codes.
3. *Web Browser loads applet*. The applet is first run through the java run – time security engine (i.e. checks the applet for suspicious code) and then loaded into memory.
4. *Applet invokes CORBA server Objects*. The Java applet can include IDL-generated client stubs, which let it invoke objects on the ORB server. Alternatively, the applet can use the CORBA DII to generate server requests "on-the-fly". The session between the Java applet and the CORBA server objects will persist until either side decides to disconnect

The Certification Server is responsible for creating, signing and revoking the certificates. To revoke certificates the pull model is used. Each client and the server have to pull the Certification Revocation List (CRL) from the server.

We model the collection of data from a number of incident databases and their propagation to a centralized incident database as an ETL workflow.

It is possible to determine typical tasks that take place during the transformation and cleaning phase of the population of a central database. We adopt the approach of [23] and we further detail this phase in the following tasks: (a) incident data analysis; (b) definition of transformation workflow and mapping rules; (c) verification; (d) transformation; and (e) backflow of cleaned incident data.

After, collecting the incident data from the databases and store them in a common format, the system has to decide if and which of these belong to a specific incident, and which aren't. This decision is based on the fact that attacks belonging to the same incident have common fields, especially in the section of the target and source IP addresses. To facilitate this procedure, further business analysis with the application of data mining techniques can be applied.

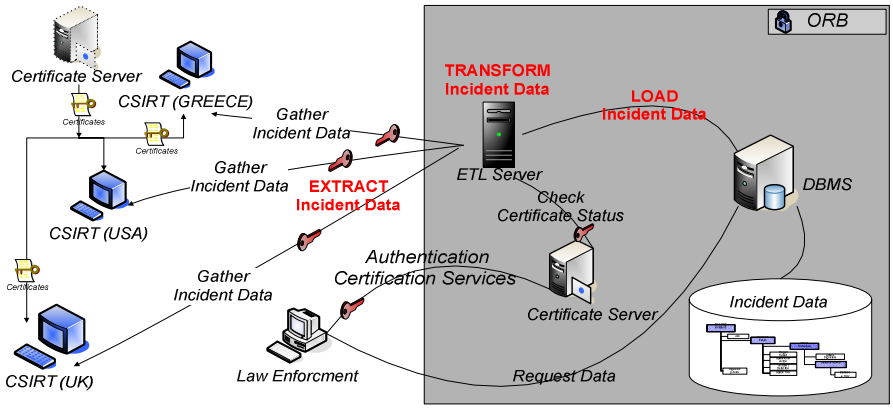

Fig. 3. System architecture for incident correlation

The system runs the ETL process on a daily bases to collect the required incident information. Usually, such procedures are executed during the night, to avoid overload the incident databases with an extra operational cost.

The structure of the centralized incident database is orthogonal to the usage of the ETL workflow. However, we propose the model described in [4] as the model of the centralized incident database due to the fact that it is developed in order to be a general incident database and thus can store a wide number of incident managerial and technical data. Along with that, this model considers an incident not as a one step process. The model is able to record all data associated with every step an incident include. Such consideration and structure will allow us to store incident records collected from different CSIRTs as steps of the same incident. For lack of space, we refer the interested user to [4] for a complete analysis of the incident model.

6 Conclusions and Future Work

In this work, we have provided a framework for automated incident management. We have presented an architecture based in advance database techniques able to collect and correlate incident related information from different sources. Our framework enhances the incident management process by allowing the law enforcement units to (a) collect the required evidence from incident data that are spread through a number of different incident management systems; (b) transform, clean, and homogenize them; and, finally, (c) load them to a central database management system. In that way, we provide a common representation of security incidents and thus, we minimize the mean time between the appearance of a new incident and its publication to the worldwide community.

As far as future work is concerned, we are currently working on the optimization of incident correlation procedure by optimizing the ETL process for the population of the centralized incident database. Also, we are interested in finding optimized methods for the correlation of incident data in the central database through the usage of data mining techniques. Finally, a challenge will be the personalized access of the

central incident database, in the sense of [15], for different levels of authorization allowing not just the law enforcement units, but, also, a number of different user types to access part of the stored incident data.

Acknowledgments

This work is co-funded by the European Social Fund (75%) and National Resources (25%) - Operational Program for Educational and Vocational Training II (EPEAEK II) and particularly the Program PYTHAGORAS.

References

[1] D. Alessandri, M. Dacier. VulDa. A Vulnerability Database. In Proceedings of the 2nd Workshop on Research with Security Vulnerability Databases, 1999.

[2] B. Athman, B. Benatallah, M. Ouzzani, H. Lily. Using Java and CORBA for Implementing Internet Databases. In Proceedings of the 15th International Conference on Data Engineering (ICDE'99), pp.218-227, 1999.

[3] B. Athman, B. Benatallah, L. Hendra, J. Beard, S. Kevin, O. Mourad. World Wide Database-Integrating the Web, CORBA and Databases. In Proceedings of the ACM SIGMOD Conference (SIGMOD'99), pp. 594-596, 1999.

[4] M. Belsis, L. Smalov. Building an Enterprise IT Security Management System. In Proceedings of the 18th IFIP International Conference on Information Security, Athens, Greece, 2003.

[5] D. Corner. IDMEF-"Lingua Franca" for Security Incident Management. SANS GIAC Security Certification, V.1.4b, 2003.

[6] Y. Demechenko. Incident Object Description and Exchange Format Data Model and Extensible Markup Language (XML). Internet Draft, 2001.

[7] Y. Demchenko, H. Ohno, G. Keeni. Requirements for Format for Incident Report Exchange (FINE) (draft-ietf-inch-requirements-00.txt), 2003.

[8] K. Ebru, O. Gokhan, D. Cevdet, K. Nihan, K. Pinal, D. Asuman. Experiences in Using CORBA for a Multidatabase Implementation. In Proceedings of the 6th International Workshop on Database and Expert System Applications, London, 1995.

[9] Commission of the European Communities Security Investigations Projects. Project S2003-Incident Reporting a European Structure "Final Feasibility and Strategy Report". Report No19733, version 1.0., 1992.

[10] M. Fullmer, S. Romig. The OSU Flow-Tools Package and Cisco NetFlow Logs. In Proceedings of the 14th Systems Administration Conference, pp. 291-303, 2000.

[11] H. Galhardas, D. Florescu, D. Shasha and E. Simon. Ajax: An Extensible Data Cleaning Tool. In Proceedings of ACM SIGMOD (SIGMOD'00), pp. 590, Texas, USA, 2000.

[12] S. Gritzalis, D. Spinellis. Addressing Threats and Security Issues in World Wide Web Technology. In Proceedings of CMS '97 3rd IFIP TC6/TC11 International Joint Working Conference on Communications & Multimedia Security, IFIP, Chapman & Hall, pp.33-46, 1997.

[13] IBM. IBM Data Warehouse Manager. Available at: www-3.ibm.com/software/data/db2/datawarehouse

[14] Informatica. PowerCenter. Available at: www.informatica.com/products/data+integration/power-center/default.htm

[15] G. Koutrika, Y. Ioannidis. Personalization of Queries in Database Systems. In Proceedings of the 20th IEEE International Conference on Data Engineering (ICDE'04), pp. 597-608, Boston, USA, 2004.
[16] R. Kimball, L Reeves, M. Ross, W. Thornthwaite. The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing, and Deploying Data Warehouses. John Wiley & Sons, New York, 1998.
[17] M. Lenzerini. Data Integration: A Theoretical Perspective. In Proceedings of 21st Symposium on Principles of Database Systems (PODS), pp. 233-246, Wisconsin, USA, 2002.
[18] Microsoft. Data Transformation Services. Available at www.microsoft.com
[19] NIST, New tool for identifying Vulnerabilities Up and Running. Journal on Research of the Nat. Institute of Standards and Technologies, 2001.
[20] Oracle Corp. Oracle9i™ Warehouse Builder User's Guide, Release 9.0.2, 2001. Available at: http://otn.oracle.com/products/warehouse/content.html
[21] MITRE. Oval Web Site. 2005. Available at: http://oval.mitre.org/
[22] PURDUE University. 2005. Available at: https://cirdb.cerias.purdue.edu/website/
[23] E. Rahm, H.Hai Do. Data Cleaning: Problems and Current Approaches. Bulletin of the Technical Committee on Data Engineering, 23(4), 2000.
[24] V. Raman, J. Hellerstein. Potter's Wheel: An Interactive Data Cleaning System. VLDB'01, pp. 381-390, Roma, Italy, 2001.
[25] A. Simitsis. Modeling and Optimization of Extraction-Transformation-Loading (ETL) Processes in Data Warehouse Environments. National Technical University of Athens: PhD Thesis, Athens, Greece, 2004.
[26] A. Simitsis, P. Vassiliadis, T. Sellis. Optimizing ETL Processes in Data Warehouse Environments. In Proceedings of the 21st IEEE International Conference on Data Engineering (ICDE'05), Tokyo, Japan, 2005.
[27] A. Simitsis, P. Vassiliadis, T. Sellis. State-Space Optimization of ETL Workflows. Accepted in Journal of IEEE Transactions on Knowledge and Data Engineering (TKDE).
[28] P. Vassiliadis, A. Simitsis, P. Georgantas, M. Terrovitis, S. Skiadopoulos. A Generic and Customizable Framework for the Design of ETL Scenarios. Accepted in Journal of Information Systems.