Quantifying Integrity Impacts in Security Risk Scoring Models

Eli Weintraub
School of Industrial Engineering and Management
Afeka Academic College
Tel Aviv, Israel

***Abstract*—Organizations are attacked daily by criminal hackers. Managers need to know what kinds of cyber-attacks they are exposed to, for taking defense activities. Attackers may cause several kinds of damages according to the knowledge they have on organizations' configuration and of systems' vulnerabilities. One possible result of attacks is damaging the database. Estimations of attacks' impacts on database integrity are not found in literature, besides intuitive managers' assessments. The aim of this work is defining a quantitative measure, which takes into consideration the known vulnerabilities threatening on database integrity and proving its feasibility. In this work a quantitative integrity impact measure is defined, formulated, illustrated and evaluated. The proposed measure is based on the real database configuration. The superiority of the measure over current practices is illustrated.**

Keywords*—Security; cyber-attack; risk score; vulnerability; database integrity

I. Introduction

Hackers attack organizations and cause various kinds of damages. Damages may span from stealing data, changing their software or paralyzing their website [1]. Such attacks expose organizations' computers for long periods of times, sometimes for weeks. The long periods start the moment, the vulnerability has been detected until the time a patch is prepared and loaded in organizations' network. According to [2] there is a need for a solution that can rapidly evaluate system vulnerabilities' potential damages needed to decide what risk mitigation activities the organization has to take.

Information systems contain large amounts of software vulnerable components stemming from improper design plans or programming bugs. Attacks are conducted by exploiting software vulnerabilities existing in the target system. When an organizations' network is exploited the organization might be exposed to three kinds of damages: degradation of confidentiality, breaking integrity of the database or downgrading networks' availability to users. In order to decide on defense actions the organization needs to take, the organization should have accurate knowledge of its network, focusing on systems' vulnerabilities.

Data quality is defined as fitness of the data for organizations' purpose [3]. High quality data enables decision making and planning. In healthcare industry, poor data quality could lead to increase in mortality [4]. A major improvement in dealing with data was the invention of the relational databases introduced by Codd [5]. Since then the Structures Query Language (SQL) is considered the industry standard for data management [6]. However, there exist several functionalities that distinguish SQL from the relational model [7] for instance dealing with null values. The implications of those differences are not well understood yet. By exploiting inference and reasoning, integrity constraints constitute a principled approach for detecting inconsistency and improving data quality [6]. Lack of completeness makes data in the database dirty, a situation which might cause fatal consequences to organizations [8]. Techniques for keeping database integrity are complicated to implement and time consuming [9]. Several methods have been developed but none are sufficiently general for providing a complete practical impact solution. The common practice in database applications is still based on ad hoc techniques. The main approaches are database triggers and hand-coding at the application level. Both methods have major disadvantages, as they are prone to programming errors. So, the assumption underlying in this research is that database integrity is generally insufficient, especially after cyber-attacks when integrity might be massively broken. This work focuses on assessing organizational risks by using real updated contents of their information systems and databases. Organizations decide on defense activities according to the potential damage and to vulnerability characteristics [10]. Damage evaluation activities are performed in two ways. First, directly by searching the specific damages cause by the attack to the technological environment, which might be a complicated task to perform due to destroyed by the attacker. Second, indirectly by comparing after-attack systems' integrity to before-attack integrity. This work uses the second way, focusing on estimating the integrity impacts which is a metric used by security scoring models. This metric measures the impact to integrity of a successfully exploited vulnerability. Integrity refers to the trustworthiness and guaranteed accuracy of information. We focus on estimating the integrity impact factor which is used by security assessment models, a metric that measures the degradation of integrity to the attacked system when exploited. It is an environment-specific indicator used to measure the potential damage to the integrity that could be affected by an attack. The greater the proportion of damaged components to systems' integrity is, the higher is the score. This work proposes a formula and model which enables assessing the actual environment attacked instead of using subjective rough assessments of the users. The proposed model is based on the real-time information on systems' configuration, as proposed by [11]. This work shows the

advantages quantitative exact risk scoring based on organizations' real environment instead of subjective users' assessment.

Organizations use various kinds of software tools to defend themselves from cyber-attacks; few of them are anti-virus, anti-span, firewalls, network and intrusion detection systems and prevention systems. Continuous Monitoring Systems (CMS) monitor computer systems in real time to detect vulnerabilities and notify as early as possible to limit organizations' exposure to attacks. Advanced monitoring tools use vulnerabilities databases which are continually updated whenever new vulnerabilities are published thus minimizing exposure time frame. CMS use scoring algorithm algorithms which assesses potential business damages in correlation to vulnerabilities database. This work proposes a CMS framework which evaluates risk scores relating to the actual configuration, focusing on an algorithm which improved accuracy of one out of several parameters used by scoring algorithms for security risk scoring computations. The specific parameter is the database integrity impacts parameter which measures the impacts of an attack on database integrity.

Evaluating integrity impacts are usually performed in two cases; first, on a regular basis for evaluating organizational security defense status, and second, after the system is attacked, for making urgent management decisions relating to recovery alternatives. Accurate risk scoring is critical in both cases for organizations' risk mitigation activities. Current security risk scores rely on qualitative estimates, thus might cause organizations under-mitigation of major risks and over-mitigation of minor risks. By using actual network configuration, organizations will be able to build IT systems in proportion to risks. According to [12] management of information systems under the conditions of frequent changes is a complex recognized problem, but the common solution is still absent. This work defines a new algorithm for assessing the integrity impact metric. The algorithm is based on two grounds: first, knowledge concerning actual database contents after the attack, and second, score evaluation based on quantitative rather than qualitative measures.

The rest of the paper is organized as follows. Section 2 describes current known existing solutions. Section 3 describes access control systems. Section 4 overviews database integrity constraints. Section 5 presents the proposed framework. Section 6 presents the integrity impact metric computation. Section 7 concludes and suggests future research directions.

II. Existing Solutions

Security monitoring tools intended to produce up to date risk scores should include the updated information managed by the external vulnerabilities databases. There exist several external vulnerabilities databases, some are public other proprietary for usage by their owners only [2]. Examples are The Sans Internet Storm Center services and The National Vulnerability Database (NVD). This work uses NVD vulnerabilities database for illustration of the proposed model.

Risk scoring systems make use of several parameters for evaluation organizations' risk scores in coordination with the vulnerabilities' that threaten the target system. The Common Vulnerability Scoring System (CVSS) is a scoring system which enables characterizing vulnerabilities and predicting risks [1]. CVSS uses three groups of parameters. Each group is represented by score compound parameters used for scoring computations. The basic parameters represent the intrinsic characteristics of the vulnerability such as the access vector used for exploits, access complexity, and authentication level needed for the attacker. The temporal parameters represent the vulnerabilities' specifications that might change over time due to defense activities taken such as patches published or updated new knowledge regarding vulnerabilities' actual damages to organizations. The environmental parameters represent the characteristics of vulnerabilities as configured by the specific organization, such as operating systems used by the organization, database management software and applications in use.

Organizations using CVSS scoring system may gain a standard scale for characterizing vulnerabilities, scoring risks and normalizing vulnerabilities according to their IT environment.

In order to compute the risk score, the CVSS algorithm performs manipulations on the three parameter groups, which must be specified prior to scoring algorithm execution. Basic and temporal parameters are specified by products' vendors who have the best knowledge of their product, its structure, design and flow of logic. Those parameters relate to the general characteristics of the vulnerability and to an anonymous attacked organization, but does not relate specifically to organizations' configuration. Environmental parameters are specified by the users who have the best knowledge of their environments and its impacts on their organization. This paper focuses on environmental metrics.

Environmental parameters are of three groups:

1) Collateral Damage Potential (CDP) which measure the potential damage caused by vulnerability.

2) Target Distribution (TD) which indicates the percentage of vulnerable components in organizations' network.

3) Security Requirements (CR, IR, AR).

Three parameters indicating the security requirements indicating the importance of their security objectives: Confidentiality Requirement (CR), Integrity Requirements (IR), and Availability Requirements (AR).

In this work we focus on the integrity impact metric which refers to the trustworthiness of information included in the database. This metric is calculated basing on two parameters, a basic and an environmental parameter. Parameter (I), is a basic parameter which measures the impact to integrity of a successfully exploited vulnerability. The possible values for this metric are: None, Partial and Complete, representing states of No impact, Partial, and Complete compromise of system integrity. Increased integrity impact increases the vulnerability score.

The environmental metric Integrity Requirement (IR) enable to customize the CVSS score depending on the

importance of the affected IT asset to a user's organization, measured in terms of integrity. IR metric has three possible values: "Low", "Medium" or "High."

The full effect on the environmental integrity impact score is determined by the corresponding base impact metric that modifies the environmental score by reweighting the (base) integrity score and environmental impact metric.

According to Federal Information Processing Standards (FIPS) 1995 [13], organizations assign their IT resources security importance measures based on component location in the environment, business function using it, and potential losses in case the component is damaged. U.S. government assigns every IT asset to a group of assets called a system. Every system is assigned three importance ratings according to three security objectives: confidentiality, integrity, and availability, to represent the potential impact on the organization in case the system is compromised. According [14] CVSS follows this general model, but does not require organizations to use a particular system for assigning the impact ratings. According to NIST, organizations should define the specifications of security risks of their specific environment. however, [14] states that NIST does not define the ways organizations have to specify that information. The Department of State has implemented a scoring program called iPost that provides continuous monitoring capabilities of security risks for IT configuration. According to [15] the iPOST scoring model does not define the base scores of CVSS to reflect the characteristics of its specific environment. This work presents a model aimed to cope with this issue.

Quantification of the environmental parameters in CVSS system has been suggested in a research demonstrating improvements in accuracy of risk scores by using the actual IT configuration [16]. Those researchers suggest assessing damage potential using a directed graph which represents the interrelationships between systems' components representing message passing probabilities between components. In [11], author suggests quantifying risks based on a configuration management database which manages information on systems' components such as data tables, data items and security specifications. This paper continues this research direction which is aimed at improving risk scoring accuracy by quantification of the integrity impacts metric, basing evaluations on the specific organizations' environment.

III. Access Control

Access Control refers to control how Information Technology resources are accessed so that they are protected from unauthorized modifications or disclosure [17]. Access controls give organizations the ability to control, restrict, monitor and protect resource availability, integrity and confidentiality. A decision weather a user may access a specific resource is a process comprising two steps: authentication and authorization. Authentication is the process of identifying the user, and authorization is the decision of allowing him to access particular resources. Authorization is a core component of every operating system, using access criteria rules to enable its decisions. Access tables manage the information weather a user has the permissions to perform varied operations on resources. Granting access rights to users should be based on the users' need-to-know in order to perform his work. The different access criteria can be enforced by roles, groups, location, time, and transaction type. Roles are based on organizational job assignments. Groups are a way of assigning access control rights. A group represents a couple of users who require the same types of access to information. Using groups is easier to manage then assigning rights and permissions to each user. The need-to-know principle is based on the concept that users should be given access only to the information they require to perform their duties. Giving any more rights to a user raises the possibility of that user to abuse the permissions assigned to him, thus raising the risks of illegal usage. An Access Control Model is a framework that dictates how users access resources using mechanisms to enforce the rules of the model. Access control systems enable organizations to detect illegal access to their IT systems weather by external hostiles or inner unauthorized employees.

IV. Database Integrity Constraints

Database structure is defined using a data definition language (DDL) which is also used to specify properties of database contents. Definition of storage structure and access methods is done using special DDL statements. The data stored in the database must satisfy certain consistency constraints. The DBMS checks these constraints every time the database is updated. According to [18] integrity constraints can be categorized to five types. The constraints are presented using SQL notation.

1) Domain Constraints: A domain of possible values must be associated with every data item, for example integer types or range of real numbers. Those are the most elementary form of integrity constraints. Those constraints are tested by the DBMS each time a new data item is entered into the database.

2) Integrity Constraints: Constraints aimed to assure that changes made to the database do not result in loss of consistency. Such constraints guard against accidental damages to the database and can be implemented using arbitrary predicates pertaining to the database. Some forms of integrity constraints are: Not Null, Unique, and Check (<predicate>). The Not Null constraint prohibits the insertion of null values for specified attributes. The Unique constraint says that a list of attributes form a candidate key, that is, no two tuples in the table can be equal on all the primary-key attributes. The check Clause when applied to a table specifies that a predicate must be satisfied by every tuple in the table.

3) Referential Integrity Constraints: A value that appears in one table for a specific attribute also appears for a certain set of attributes in another table. Declaration of a foreign key in one table specifies that their must exist in another table as a primary key. Database modification might cause violation of referential integrity.

4) Assertions: Any condition that the database must always satisfy. Domain constraints and referential integrity constraints are special forms of assertions, however there are many constraints that cannot be expressed using those simple forms. Modifications to the database are allowed only if they do not violate that assertion.

5) Authorization: Differentiation among users according to type of access they are permitted on various data values in the database. In this work we focus on database contents, so we will not deal with authorizations.

The model presented in this work focuses on quantification of the proportion of integrity rules which might be damaged during a cyber-attack. The impacts of attacks cause breaking integrity rules, in addition to damages to database contents. The model includes a description of the CMS structure and components, then a presentation of the new impact factor formula and algorithm. Then the detailed algorithms' computations of the impact score after an attack has been conducted on a sample database which is governed by integrity rules. We illustrate the feasibility of computing a quantitative measure which is based on the actual organizational database, relating to specific vulnerabilities. We show the advantages of the proposed measure compared to current practices.

The proposed model may be useful in two cases: First, cases in which an organization has been attacked, forcing management to make quick decisions relating to recovery alternatives. Secondly, for assessing the potential damages to database integrity, in cases of future potential attacks.

V. Proposed Framework

The proposed framework is based on two principles (see Fig. 1). First, the environmental parameters are based on systems' configuration as opposed to current practices. The configuration specification is specified in a systems' Configuration Management Data Base (CMDB) [15]. This capability enables basing the scoring models on the actual IT configuration rather than relying on user's estimates as proposed by [16]. According to [19] it is impossible for organizations to make precise estimates of the economic damages caused by an attack without having full knowledge of users' IT environment. In [11], [20] authors state that network configuration should be monitored on a continuous basis, and vulnerabilities must be analyzed to provide the necessary appropriate security level. Secondly, the computed integrity impact is a real number which enables to put on one uniform scale for comparison reasons various states rather than the current ordinal three-grades scale, naming low, medium and high. This enables increasing accuracy of estimates, representing higher resolution risk assessment.

According to the proposed framework the system scans continually the NVD database of published vulnerabilities, searching for threats to actual organizational technological assets, and computing organizational risk scores. The proposed architecture and components are the following:

- Continuous Monitoring System (CMS)

The system scans the NVD searching for threats systems components producing updated risk scores of components which have been found to be threatened by relevant vulnerabilities. Computations of integrity impacts impact scores are performed in three cases:

1) When a new vulnerability is published, or its status changed by NVD.

2) When a change made to a systems' component: Changes may expose the component to new vulnerabilities or otherwise prevent risky states. Changes to the structure of the database or its contents may generate new exposures or changes of database integrity.

3) When the access control system signals that a certain systems' component was exploited by a human or program activity.

- Vulnerabilities database (NVD)

Vulnerabilities database includes all known vulnerabilities as published by database owners. Examples of vulnerability specifications used by NVD are: vulnerability category, vendor name, product name, published vulnerability start and end dates, vulnerability update dates, vulnerability severity, access vector, and access complexity [13].

- The Common Vulnerability Scoring system (CVSS)

The risk scoring system (CVSS) is the algorithm this research uses for illustration of the proposed model.

- Configuration Management Database (CMDB)

CMDB is a meta-database which includes all database components: data tables, columns, data items and all integrity constraints of the target system. The CMDB includes also security requirements (CR, AR, IR).

- Access Control System

Illegal access to systems' components are caused by hackers, illegal user or software flaws. Hostiles look for vulnerabilities or backdoors which let them bypass the access control system or change systems' logic, thus reaching illegally data or software components. This module monitors all systems' components including hardware devices, software components, databases, database integrity constraints, stored procedures, data tables, table columns or database data item. When the module recognizes an illegal access to a certain component it alerts operators or terminates processes.

- Organizational Database

Activation of CMS triggers a process of computing IR metrices relating to the specific attack and the specific environment the moment the attack was conducted. Computing IR involves a scan of all integrity constraints to check weather each constraint is still consistent after the system has been attacked. Attacks on database items which change or delete database, its contents might cause damages to the integrity of the database by breaking the integrity rules so that they are no more consistent. Checking the validity of the integrity constraints needs validation of the constraints to database contents to compute the amount of degrading in integrity rules. After the IR is computed it is written to the environment metrics database together with an indication of the environment the moment the system has been exploited.

- Environment metrics database

Each activation of CMS system, CVSS computes the new IR metric and writes it on the environment metrics database in addition to an equivalent environmental indicator. The environmental indicator includes a version number and a list of configuration components in accordance with the computed IR.

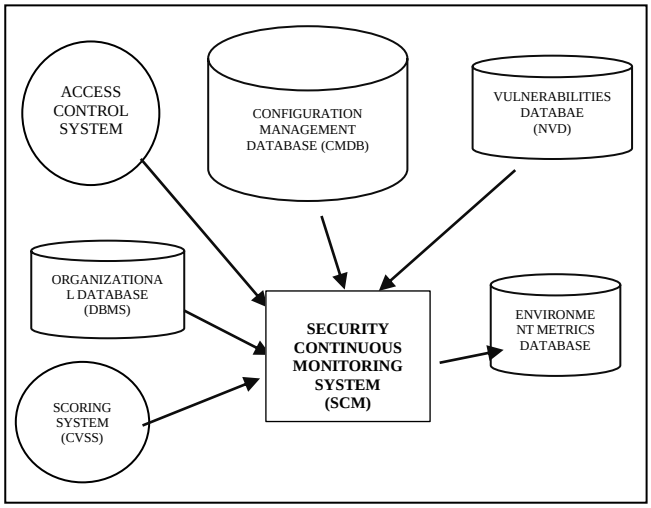

Fig. 1. Security continuous monitoring system framework.

VI. Integrity Impact Metric Computation

After a cyber-attack on the system, CMS executes the IR-Algorithm which checks weather the consistency the database is kept following the integrity constraints. IR-Algorithm scans all integrity constraints (written in the DBMS) and reads database contents calculating the amount of actual in-consistencies in the database, producing the IR environmental metric. IR-metric is an input to the CVSS scoring system. Below in Fig. 2 is outlined the IR-Algorithm pseudo-code.

For all integrity constraints written in CMDB compute inconsistency proportion:

; for Domain Constraints
The number of data items inconsistent with domain constraints / number of data items controlled by column domain constraints.

; for Integrity Constraints
The number of data items inconsistent with Integrity Constraints / number of data items controlled by relation or column domain constraints.

; for Referential Integrity Constraints
The number of Foreign-Keys inconsistent with Referential Integrity Constraints (meaning that there is no data item defined as Primary-Key of another table which is equal to the value of the Foreign-Key) / number of data items Foreign-Keys controlled by Referential Integrity Constraints.

;for Assertion Constraints
The number of data items inconsistent with Assertion Constraints / number of data items controlled by all database Assertion Constraints.

End For all
IR metric = Sum of four computed Integrity inconsistencies / total number of constraints.

Fig. 2. IR-Algorithm pseudo-code.

After computing four types if integrity inconsistencies we compute the IR-metric. IR-metric is a real number in the range [0..1].

0 <= IR <=1 since for the following argumentations:

The minimal IR value is assigned to zero in case there are no constraints on the database. The maximal IR value is 1 in case each data item which is controlled by a constraint is inconsistent with all the constraints.

Hence, the IR metric gets real value on [0..1] scale representing the proportion of the exact number of inconsistent data items out of all database constrained data items. Moreover, the metric represents the proportion of inconsistencies which are effective to the constraints, meaning that data items that are not constraint-controlled are not considered in IR computations.

We illustrate now the advantages of the new IR metric compared to the current possible values L, M, H.

- Illustration

The database contains two tables: bank_accounts and branch_names.

Following the SQL-like notation definitions of the database:

Create table bank_accounts

(acc_num numeric (14),

branch_code references table branch_names (branch_code),

acc_balance numeric (14,2),

primary key (acc_num))

Create table branch_names

(branch_code numeric (3),

branch_name char (20),

branch_addr char (30),

Primary key branch_code))

Following Tables I and II describe the database structure and contents.

TABLE I. Bank Accounts

Acc_num	***Branch-code***	***Acc_balance***
10	100	1000
20	200	2000
30	Abc	3000
40	100	Bal2

TABLE II. Branch Names

Branch_code	***Branch-name***	***Branh_addr***
100	Branch one	Addr one
200	Branch two	Addr two
500	Branch five	Addr five
500	Branch six	Null
700	Branch seven	Addr seven

Following are the six database integrity constraints:

Domain Constraints:

1) Create domain acc_balance numeric
2) Create domain acc_balance (branch_code) numeric

Integrity Constraints:

3) Constraint branch_names (branch_addr) not null
4) Constraint branch_names (branch_code) unique

Referential Integrity Constraint:

5) See referential integrity definition of branch_code in bank accounts table

Assertion Constraint:

6) Create assertion as check
(select (sum (acc_balance)

From bank_accounts

< 5000))

Hence, six integrity constraints:

Following are the IR computations:

Constraint 1:

One acc_balance out of 4 data items is inconsistent with constraint, hence domain inconsistency = 1/4

Constraint 2:

All 5 branch_codes are numeric according to the constraint, hence inconsistency = 0

Constraint 3.

One branch address is not null out of 5, hence integrity inconsistency = 1/5

Constraint 4.

Two branch codes are not unique out of 5 hence integrity inconsistency = 2/5

Constraint 5.

Referential integrity inconsistency by account number 30 since foreign key branch_code does not exist in branch names table, hence referential inconsistency of 1 out of 4 foreign keys = 1/4

Constraint 6.

One Assertion constraint inconsistency since sum of account balances is greater than 5000, assuming that assertion was broken when one of the accounts was inserted into the table hence assertion inconsistency = 1/4

We can compute now:

IR-metric = (1/4 + 0 + 1/5 + 2/5 + 1/4 + 1/4) / 6 = **0.225**

As illustrated, this metric is based on the real contents of the data base. The metric is calculated by computing the proportion of inconsistencies of data item to integrity constraints.

According current scoring algorithms the user assesses the impact of a possible cyber-attack on database integrity, using an ordinal scale: Low. Medium, High. The user has no quantitative basis and no algorithm assisting him assessing integrity impacts.

Let us now compute the change in the computed impact score, in the case of adding one more inconsistency to the database.

Suppose account balance of account number 30 is changed by a hostile attacker and changed to aaaa hence breaking constraint number 1 which forbids not-numeric values in a column.

Computing inconsistency produces a new IR-metric which is higher than the previous score.

IR-metric = (2/4 + 0 + 1/5 + 2/5 + 1/4 + 1/4) / 6 = 0.267.

Hence, 0.267 > 0.225 which is logical.

Performance of the algorithm involves executing one loop on all constraints and additional inner loops for each constraint, according to constraints' logic. The total performance time depends also on number of items in the specified columns and on the amount of computations according to constraints' logic. So, approximating performance runtime is not trivial. Nevertheless, organizations facing database integrity risks are facing legal and privacy issues which reasonably undermine performance issues.

VII. Conclusions

This work presents a framework of a Security Continuous Monitoring System structure and mechanism, aimed to evaluate security risk scores. The CMS uses CVSS scoring model for risk scoring. This research has proved the advantages of the proposed algorithm over current practices in three areas: 1) The feasibility of defining a quantitative measure, in contrast to current qualitative measures. 2) The feasibility of basing a risk scoring algorithm on the real database structure and contents. 3) The algorithm computes the exact proportion of damages caused to database integrity.

The algorithm may be used in two ways: first, after attacks are conducted on the organization. At such instances management must decide urgently what damages the organization suffered and what is the right moment the organization may go on-air again. At such occasions the uncertainties are large, complicating even more making decisions. The proposed impact factor helps management understand the quantitative damages to the database, thus, decisions can be based on accurate actual configuration. Second, at each occasion of a change made to the database or to the published vulnerabilities database, a risk scoring computation is performed, using the history of previous integrity metrices, predicting the future impact metrices and new risk scores. Such predictions assist management in managing their security risks in accordance to the predicted damages.

Future research may span to several directions: looking for ways to shorten performance times, mathematical formalization of the presented algorithm, evaluating

algorithms' accuracy and usefulness using real cases. More research direction are: using data mining models for risk scoring predictions; Developing integrity impact prediction models which will be based on database rules instead of long database scans, thus minimizing runtime; Incorporating data security requirements relevant to each data item into the scoring formula, which may add a new values and higher resolution of parameters used by risk scoring models.

REFERENCES

[1] P. Mell, K. Scarfone, and S. Romanosky, "CVSS – A complete guide to the common vulnerability scoring system, version 2.0", 2007.

[2] Y. F. Nũez, "Maximizing an organizations' security posture by distributedly assessing and remeding system vulnerabilities", IEEE – International Conference on Networking, Sensing and Control, China, April 6-8, 2008.

[3] M. Gamble, & C. Goble, "Quality, Trust, and Utility of Scientific Data on the Web: Towards a joint model," Paper presented at the Third International Web Science Conference, 2011.

[4] C. Liu., A. Talaei-Khoei, D. Zowghi, and J. Daniel, "Data Completeness in Healthcare: A Literature Survey," Pacific Asia Journal of the Association for Information Systems, 9(2), pp. 75-100, 2017.

[5] E. F. Codd, "A relational model of data for large shared data banks, Communications of the ACM", 13, 6, 377-387, 1970.

[6] S. Link and M. Memari, "Static Analysis of Partial Referential Integrity for Better Quality SQL Data", Proceedings of the Nineteenth Americas Conference on Information Systems, Chicago, Illinois, August 15-17, 2013.

[7] S. Hartmann, and S. Link, "The implication problem of data dependencies over SQL table definitions: Axiomatic, algorithmic and logical characterizations", ACM Transactions on Database Systems, 37, 2, 13.1-13.52, 2012.

[8] W. Fan, and f. Geerts, "Foundations of Data Quality Management, Synthesis Lectures on Data Management", 4, 5, 1-217, 2012.

[9] H. Christiansen and D. Martinenghi, "On Simplification of Database Integrity Constraints", Fundamenta Informaticae 71, 371–417, 2006.

[10] S. Tom and D. Berrett, "Recommended practice for patch management of control systems", DHS National Cyber Security Division Control Systems Security Program, 2008.

[11] E. Weintraub, "Security Risk Scoring Incorporating Computers' Environment", (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 7, No. 4, April 2016.

[12] B. Carpenter., The Internet Engineering Task Force. Overview, Activities, Priorities, IETF Report to ISOC BoT, Oct. 2006.

[13] K. Dempsey, N. S. Chawia, A. Johnson, R. Johnson, A. C. Jones, A. Orebaugh, M. Scholl and K. Stine, "Information security continuous monitoring (ISCM) for federal information systems and organizations", NIST, 2011.

[14] R. Sandhu, D. Ferraiolo and R. Kuhn, "The NIST Model for Role-Based Access Control: Towards A Unified Standard", George Mason Univ., 1999.

[15] A. Keller and S. Subramanianm, "Best practices for deploying a CMDB in large-scale environments", Proceedings of the IFIP/IEEE International conference and Symposium on Integrated Network Management, pages 732-745, NJ, IEEE Press Piscataway, 2009.

[16] E. Weintraub, "Evaluating Damage Potential in Security Risk Scoring Models" International Journal of Advanced Computer Science and Applications (IJACSA), 7(5), 2016.

[17] S. Harris, All in one CISSP Exam Guide. 6Th Ed. McGraw Hill Education, 2013.

[18] A. Silberschatz, H. E. Korth, S. Sudarshan., Database System Concepts, 5th Ed. McRAW-HILL, 2006.

[19] M. R. Grimalia, L. W. Fortson and J. L. Sutton, "Design considerations for a cyber incident mission impact assessment process", Proceedings of the Intrnational Conference on Security and Management (SAM09), Las Vegas, 2009.

[20] I. Kotenko and A. Chechulin, "Fast network attack modeling and security evaluation based on attack graphs", Journal of Cyber Security and Mobility Vol. 3 No. 1 pp 27-46, 2014.