International Journal of Electrical and Computer Engineering (IJECE)
Vol. 9, No. 6, December 2019, pp. 5570~5585
ISSN: 2088-8708, DOI: 10.11591/ijece.v9i6.pp5570-5585

Security assessment framework for educational ERP systems

Hafsa Ashraf[1], Mamdouh Alenezi[2], Muhammad Nadeem[3], Yasir Javid[4]
[1,3]Faculty of Information and Communication Technology, BUITEMS, Pakistan
[2,4]Department of Computer Science, Prince Sultan University, Saudi Arabia

Article Info

Article history:

Received Feb 10, 2019
Revised Jul 17, 2019
Accepted Jul 28, 2019

Keywords:

Educational ERP
Security assessment
Software security

ABSTRACT

The educational ERP systems have vulnerabilities at the different layers such as version-specific vulnerabilities, configuration level vulnerabilities and vulnerabilities of the underlying infrastructure. This research has identified security vulnerabilities in an educational ERP system with the help of automated tools; penetration testing tool and public vulnerability repositories (CVE, CCE) at all layers. The identified vulnerabilities are analyzed for any false positives and then clustered with mitigation techniques, available publicly in security vulnerability solution repository like CCE and CWE. These mitigation techniques are mapped over reported vulnerabilities using mapping algorithms. Security vulnerabilities are then prioritized based on the Common Vulnerability Scoring System (CVSS). Finally, open standards-based vulnerability mitigation recommendations are discussed.

Corresponding Author:

Mamdouh Alenezi,
Department of Computer Science,
Prince Sultan University,
P.O.Box No. 66833 Rafha Street, Riyadh 11586 Saudi Arabia.
Email: malenezi@psu.edu.sa

1. INTRODUCTION

Enterprise Resource Planning (ERP) is the technology that provides the unified business function to the organization by integrating the core processes. Mainly it is related as a software solution; integrating information and business processes to enable sharing throughout an organization of information entered once in a database. The range of functionality of ERP systems has further expanded in recent years to include more applications, such as grants management, marketing automation, electronic commerce, student information systems, strategic planning, human resources management, customer relationship management, and supply chain systems.

The Educational ERP systems may have a number of vulnerabilities, which may be related to a specific version, generic, or due to a misconfiguration of the system. The public vulnerability repositories such as CWE, CCE, and CVE host useful information, which can be used to fix these vulnerabilities. However, there is a need to utilize the knowledge in these public repositories to address the educational ERP security problem.

Several characteristics make educational ERP systems to be more susceptible to security issues. These characteristics are [1]:

- Customizable: Logic of ERP system is altogether different from conventional software. Likewise, its deployment cannot be done in the same manner. In sagacity, ERP is the framework for software, not just software. You have a large amount of customized code according to organizations business logic.
- Complex: ERPs are huge and complex due to a number of components at one place like; database systems, application service providers, front-end programs, deployment may be on various operating systems, and large, complex integration mechanism. Often, security is removed due to complexity.

- Risky: Patches and bundle upgrade come with extended risk as they need deep understanding; you must consider and accept certain levels of risk beforehand. Not all ERP administrations can accept this risk, timely upgrades and patch application is also recommended.
- Cost of SDLC: Due to the complexity of ERP system security perspective is a hassle. During deployment of limited resources, this hassle ultimately excludes the quality of security features which cannot be demonstrated to clients in the sales process, as sales companies require rapid deployment figures [2].
- SSDLC: Secure software development life cycle software industry is still lagging to adopt this life cycle.
- Integration: The ERP program needs to be integrated with other processes and applications, and if the collaborative process is not completely secured then the ERP is prone to vulnerability.
- Authentication integration: Verifying authenticity is mandatory and be achieved using the single sign-on feature provided by maintaining Active Directory or configuring LDAP, and Kerberos, over the time this verification method has become legacy and has been exploiting by hackers [3].
- Lack of processes: Sometimes ERP exploitation is due to a number of internal sources, lack of security SOPs compliance, the least skilled sources (non-IT personnel) [4].

The process of identifying, quantifying, and prioritizing (or ranking) the vulnerabilities in a system is defined as vulnerability assessment. A significant part of security management is vulnerability scans, vulnerability assessment, and vulnerability mitigation [5]. There are various vulnerability scanners, which can easily scan systems to detect vulnerabilities. The output of these scanners may include flaws in software design, configuration issues, and network flaws. The following public repositories constitute the knowledge base for this research.

- CVE – Common Vulnerabilities and Exposures: This is vulnerability reported repository that provides detailed information on safety deficiencies and security disclosures this is functional by MITER, funded by the US Department of National Cyber Security Protection Division.
- CWE – Common Weakness Enumerations: Software community targeted at vulnerabilities and software flaws. The goal is to provide enhanced knowledge related to the weaknesses of the software. It serves as a common language, a measuring stick for software security tools, and as a baseline for weakness identification, mitigation, and prevention efforts.
- CCE – Common Configuration Enumeration: The CCE list provides a unique introduction to security-related configuration issues to improve workplace quickly and accurately connect data processing data across multiple sources and information tools.
- CVSS – Common Vulnerability Scoring System: CVSS is an open framework for communicating the characteristics and severity level of software vulnerabilities. CVSS consists of three metric groups.

This research includes multiple toolkits, which will be used in different phases of the framework.

2. RESEARCH METHOD

Security becomes an essential part of ERP systems, as they are used by various businesses or organizations like Health care, Military defense systems, Educational systems, and banking systems. The primary goal is to ensure the security of ERPs by employing stable security policies. ERPs are the complex cohesion of different modules, which are directly or indirectly dependent on each other. In this case, it becomes very difficult for policymakers to prioritize system policies that can make a system more secure [6].

Assessment of security threat in ERPs has been a daunting task for the security community due to the interdependencies of business modules. ERP systems must be able to process a wide array of business transactions and implement a complex security mechanism that provides deep access to users. The inherent complexity of ERP systems increases the complexity of security configuration and lead to potential security weaknesses [7].

During the implementation phase information security-related issues are commonly ignored, that may lead to poorly designed controls. Overlooking security issues during an ERP implementation process also results in ongoing security problems as it is very difficult and expensive to implement controls once an ERP system has been implemented [8]. Controls are essential to ascertain that tasks are performed completely and accurately and that unauthorized changes to the input do not occur. As controls act as defense mechanisms to prevent accidental hazards, discourage intentional acts, speedily detect problems, improve damage recovery and rectify errors [9, 10]. The implementation of an ERP system is also an opportunity for an organization to implement improved controls and security.

Due to insufficient effort on security issues during implementation, ERP security is ignored, which results in post-implementation problems and patching that can be expensive and ultimately lead to failure of the project [7]. Version specific vulnerabilities are related to a specific release of ERP to mitigate reported weaknesses. To overcome such weaknesses ERP developers supply patches to address specific problems. Applying continuous patches may increase the complexity and dependency of the ERP.

Patch management can also support these processes by assisting with the deployment of updates, by minimizing the risk of the change and by reducing the frequency of updates that are required [11]. Patches serve other purposes than just fixing software flaws; they can also add new features to software and firmware, including security capabilities. One key aspect of better and more secure software is the timely release of patches by vendors for the vulnerabilities in their products [12].

Open source vulnerability repositories contain information about vulnerabilities, possible mitigations, examples of demonstration, consequences, etc. These deposits are maintained and are available for public use. Provide unified, effective and standard information on security vulnerabilities [13]. According to the National Vulnerability Database maximum of eighteen vulnerabilities are published every day. This makes the job of the administrators quite bothersome. In fact, to maintain their systems secure and fully operational, they need to spend most of their time consulting security advisories in order to identify which of these vulnerabilities really represent a threat for their systems and to determine which countermeasures (e.g., installing a patch, modifying a firewall rule, etc.) must be applied [14].

Information about security vulnerabilities can be gathered from various sources namely vulnerability databases. Some of them are publicly accessible (like OSVDB, NVD, CVE, etc.), and free others can be consulted after payment of a certain amount (like SecureBase by SPI Dynamics). One of the key limitations of existing vulnerability databases is that no one provides all the necessary information needed to accurately identify the presence, exploitation, and effect of weakness. Thus, this information can only be collected by consulting more than one database. Fortunately, most databases provide information to run referrals to other databases [14].

At present there is no rule of thumb for vulnerability assessment, as every scanning procedure is different from another, each scanner vendor has its own formulation to assess vulnerabilities. Therefore, a single vulnerability may be assigned different security risk levels from different vulnerability assessment model [15]. Likewise, an architecture was proposed in [6]; it consists of vulnerability scanning module, vulnerability classification module, and deduction engine module. The vulnerability scan engine scans the host on the network. The Vulnerability Classification module classifies the vulnerabilities found in the scan report into the vulnerability of application and misconfiguration. The classified vulnerability information is stored as fact files in the deduction engine. The Deduction Engine module generates atomic attacks and attacks graphics [16]. Another approach for vulnerability assessment is Quantitative modeling of vulnerability discovery process based on shared source code measurements among multi-version software systems. Such a modeling approach can be used for assessing security risk both before and after the release of a version [17].

3. THE PROPOSED ASSESSMENT FRAMEWORK

In this proposed framework, a set of tools and algorithms are used for identification of vulnerability, clustering vulnerabilities with relevant articles, mapping articles with flagged vulnerabilities within the particular cluster, prioritizing vulnerabilities. These tasks were performed systematically with the help of the proposed framework. Figure 1 the identification of vulnerabilities of the target system is performed using automated tools, then these results are clustered using clustering algorithm and compared with the knowledgebase Common Configuration Enumeration (CCE) and Common Vulnerabilities and Exposures (CVE) to co-relate the reported vulnerabilities and to prioritize them with the help of Common Vulnerability Scoring System (CVSS).

Prioritization leads us to build our subset for highly ranked vulnerabilities, for which solution are segregated using Common Weakness Enumerations (CWE) repository and finally secure implementation plan is formulated. This proposed framework helps in analyzing and performing experiments throughout the study. The proposed framework can easily be divided into 3 phases on the bases of its working and outputs that are used in the next phase. The first phase focuses on public repositories and automated tools. The second phase defines the core working of this study in which the power of advanced algorithms are utilized for clustering and mapping the flagged vulnerabilities to the related articles. The third phase comprises of prioritizing vulnerabilities on bases of severity level that leads to finally suggesting implementation plan against the flagged vulnerabilities.

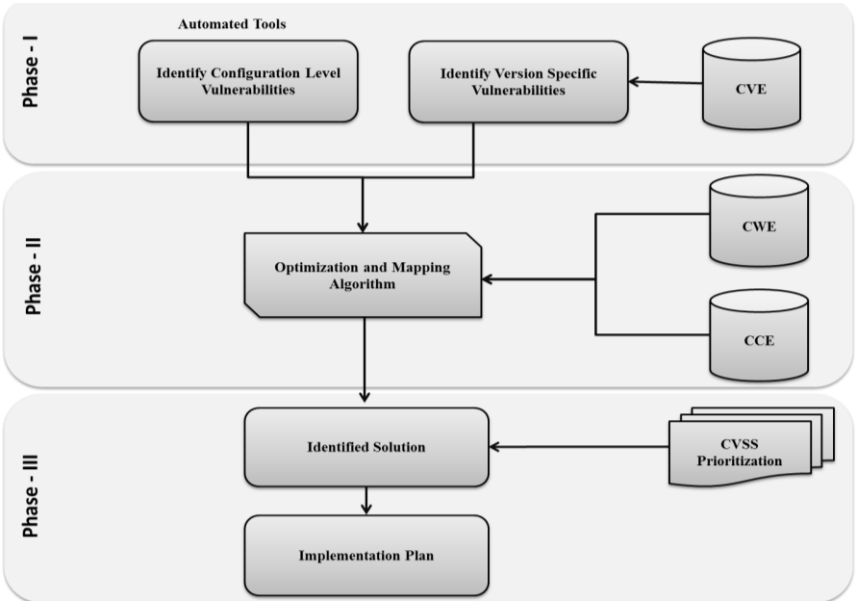

Figure 1. Proposed assessment framework

Phase 1: Public repositories and automated tools

Open source vulnerability repositories display informative details related to ERP security vulnerabilities [18]. This layer of framework addresses the importance of these public vulnerability repositories and the use of the automated tool for penetration testing. Penetration testing tools are used for vulnerability assessment in the target educational ERP system. Penetration testing tool is developed for standalone web application and as well as for enterprise solutions. Like any enterprise software, these software's are licensed or open-source. Phase 1 is depicted in Figure 2; this phase is the baseline for the formulation of the dataset that is used in phase 2.

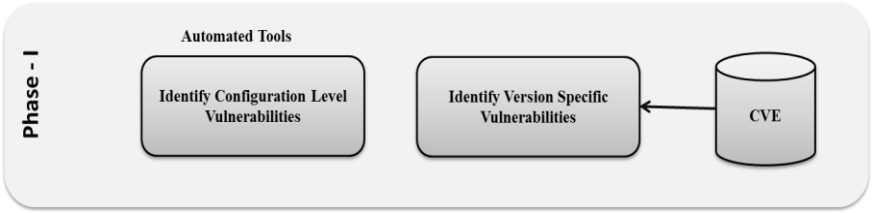

Figure 2. Proposed framework phase 1

Phase 2: Dataset optimization and use of mapping algorithm

Optimization is a technique that can make the most efficient use of recourses. This is the key quantitative tool for making decisions in an automated way. Data gathering in phase 1 is first optimized by using a clustering algorithm by which dataset is refined and then mapping algorithms are used to find best match top mitigation article with reported vulnerabilities. This step reduces the time and enhances the efficiency of mapping algorithms. Rather mapping all the flagged vulnerabilities over articles having mitigation strategies, only those clusters are used for mapping that contains matched articles against vulnerabilities.

The mapping algorithms are used to match the similarities between the documents present in the given cluster. In this study, we are calculating the similarity index within the description of articles and flagged vulnerabilities grouped in one cluster. Figure 3 illustrates that once vulnerabilities are identified of target system then using public vulnerability repositories articles are clustered, finally, the mapping is performed.

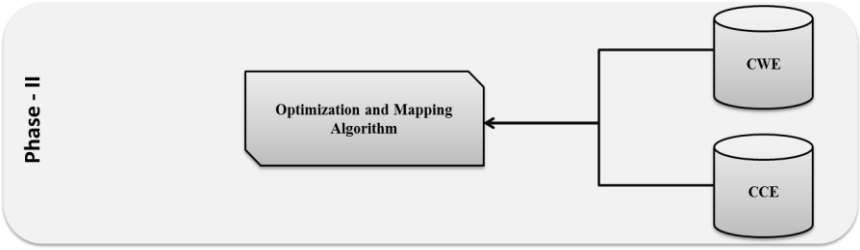

Figure 3. Proposed framework phase 2

Phase 3: Implementation plan

In this selection identified solutions are prioritized on the bases of severity level defined by CVSS - Common Vulnerability Scoring System repository. Using this prioritization, issues at high risk can easily be addressed first then medium level vulnerabilities and finally low ranked vulnerabilities will be resolved. As shown in Figure 4 once prioritization is made for the identified solutions then detail recommendations are suggested in the implementation plan section depending upon the issues reported.

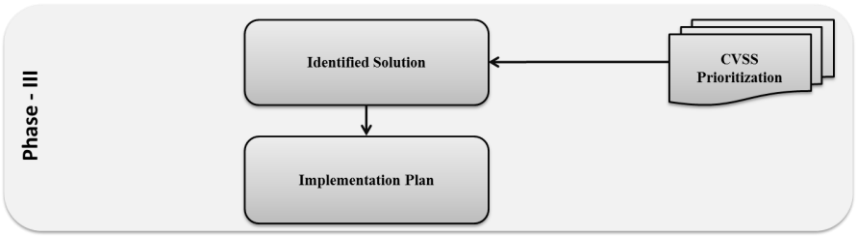

Figure 4. Proposed framework phase 3

Mainly public vulnerability repositories comprise; basic information of defects, the severity level of weakness, potential mitigation strategy, demonstration with examples, possible consequences. This widespread database is managed and publicly available for use. A key advantage of such repositories is they provide integrated and effective information on security risks. Common Weakness Enumeration (CWE), National Vulnerability Database (NVD), Common Vulnerabilities and Exposures (CVE) are highly used repositories depicted in Figure 5.

.

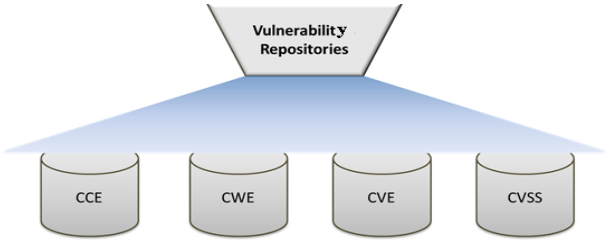

Figure 5. Public Vulnerability repositories

These repositories are extensively managed and state classification of each reported issue in terms of status, available solutions, and vulnerability type. Figure 6 shows highly reported vulnerabilities that may be categories as buffer overflow, cross-site scripting, denial of services, SQL injects and so on. The proposed framework leverages the power of public vulnerability repositories used for clustering and mapping the reported vulnerabilities to the articles. There is a number of communities working on information security at present. They use certain rules, policies, and procedure to check the security of software. Majority of information security communities use standards set by MITRE which is the US government-funded organization. In the next section, the most popular of these are discussed.

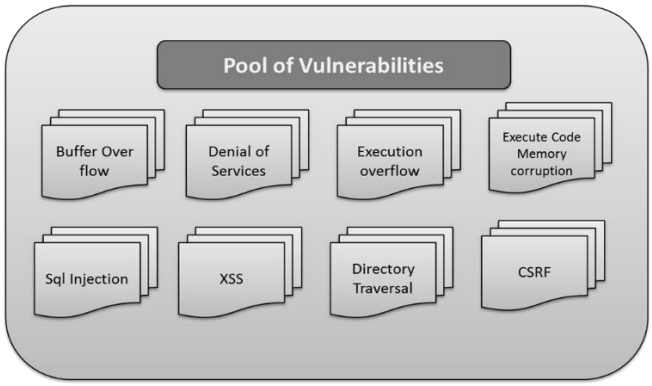

Figure 6. Type of Vulnerabilities

3.1. Common weakness enumerations (CWE)

CWE- common weakness enumeration can be defined as a classification of any software security flaw that can serve as a standard list of security weakness and provide information in unified language [19]. CWE is formed by different communities. Therefore it is known as a community project with the major goal of creating a catalog of software weaknesses and vulnerabilities that help in providing a better understanding of weaknesses in software [20]. It also serves as a standard yardstick for addressing vulnerabilities using security tools, providing techniques for identification of vulnerability, baseline mitigation strategies, and protection steps from known weaknesses.

MITRE's the founder organization of CWE; set objectives for this repository as to provide a platform that guides security assessment and software capabilities in terms of assurance and code maturity so that acquisition companies can adopt particular software [21].

At present, it is difficult to pinpoint high-quality tools that can identify the weaknesses and security flaws of software as they are new to this field and market. Few core tasks of CWE repository are listed below:

- Use the regular language to describe the weaknesses, which may present in architecture, design, or code of any software.
- Serve these defects as a standard measure for software security tools.
- Provides a communal guideline for weakness identification, its possible preclusion steps and effort required.

Most specifically finding what tool or services are best fit for which tasks are still an unanswerable question. CWE was exactly created to address such critical issues. 700 plus articles are present at CWE repository affirming CWE-ID, its title, short description, relationships common consequences, potential mitigations and so on against each CWE article, Figure 7 depicts an example of conventional CWE article.

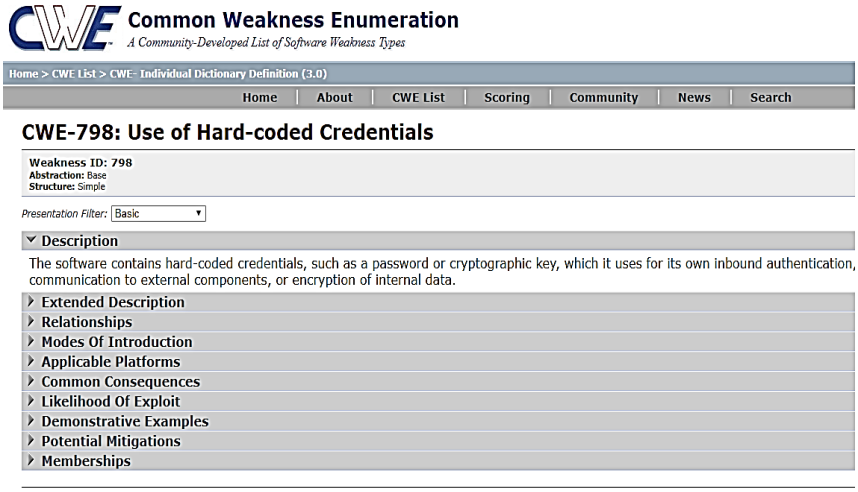

Figure 7. Example of CWE reported vulnerability [22]

3.2. Common vulnerabilities and exposures (CVE)

Early in 1999, MITER began to classify program weaknesses when CVE list was launched. As part of the CVE-MITER's CVE team, it has developed a basic rating and key classification of key risk types, attacks, errors, and other concepts that help define common weaknesses. For CVE vulnerability data, www.cvedetails.com provides the quick web-based interface. It is quite convenient to find vulnerability by selecting vendors, products, and versions. You can see statistics on vendors, product and product versions. CVE provides details on a single page in a compact view. The details of CVE interface shown in Figure 8; vulnerability details identified by CVE-ID (e.g., CVE-2006-0584) number followed by its description, reported date, impact factor, severity level, and its corresponding CWE article number.

CVE vulnerability repository is derived from the National Vulnerability Database (NVD) and in XML format, they acquire safety data from the National Institute of Standards and Technology (NIST). In addition to NVD CVE data, this repository also includes additional data from multiple resources like Metasloit modules, data supplied from vendors, potential exploits from www.exploit-db.com.

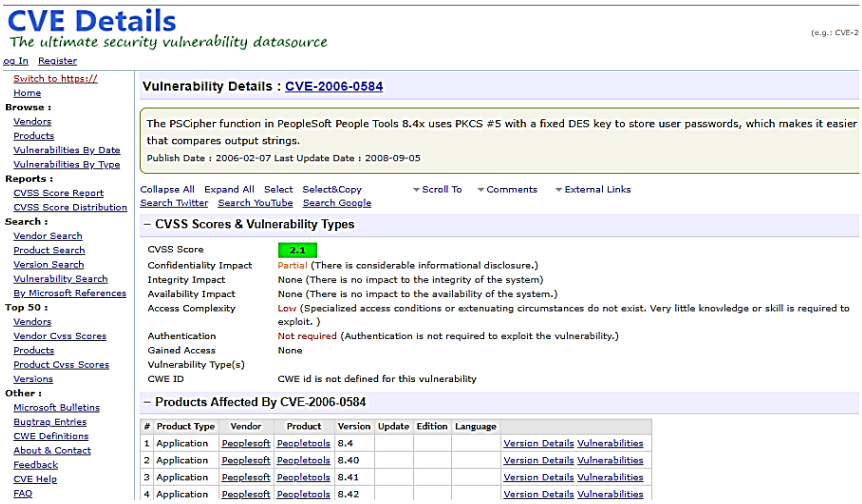

Figure 8. Example of CVE details [23]

3.3. Common configuration enumeration (CCE)

The CCE List provides unique identifiers to security-related system configuration issues to improve workflow by facilitating fast and accurate correlation of configuration data across multiple information sources and tools [24]. While working with multiple resources various factors are address; in order to use the stored information effectively; the data present on these repositories must maintain constant identification that can help in data correlation, interoperability feature may exist, the interactive capability of feeding automation and so on. Unique identifiers are assigned by CCE against each system setup problems so that it may be used to assist fast and accurate connection for configuration data through sourcing and various information tools.

Similar to CVE repository CCE also list configuration level security weakness in a unified manner, each CCE entry comprises of following few core attributes:

- CCE-ID: CCE identifier number e.g., CCE-2560-9
- CCE Description: Configuration flaws are listed in a human-readable format.
- Corrective parameters: Each system have a configuration panel and corresponding parameters are defined.
- Respective Corrective Mechanisms: Corrective measures can be more than one to get desired outputs by implementing them.

3.4. Common vulnerability scoring system (CVSS)

CVSS is an open standard and free platform for analyzing the intensity level of information system security weaknesses. The core task of CVSS repository is to assign a score to reported vulnerability on bases of its severity, define the priority of flaws to be addressed first which is set by responders.

CVSS has set score range that starts from 0 and ends at 10, 0 may be read as a lowest severe vulnerability that can harm to any system and 10 is the maximum point shows the highest level of impact on the system may cause by the weakness. Calculation formula of each reported vulnerability; so that it may be assigned a score is based on multiple factors that identify the exploitation factor and ease of exploitation to any course code or system [25]. CVSS has a different working structure, it has three scoring groups and each has a set of metrics shown in Figure 9

.

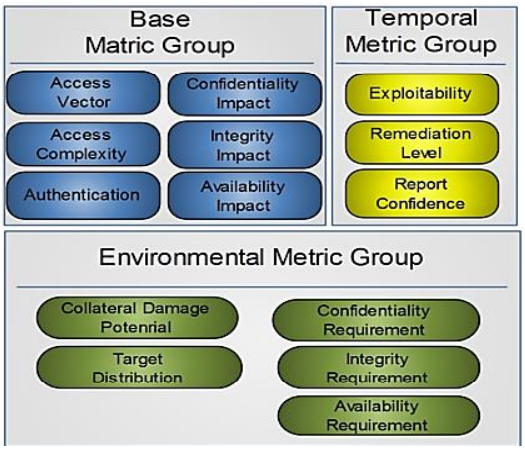

Figure 9. CVSS score metric

- The base metric group consists of those vulnerabilities which are not changed due to the effect of the environment or with the time. These vulnerabilities maintain their characteristics and show the inheritance with their parent vulnerability type.
- Temporal metric comprises of those vulnerabilities that are not consistent and change their impact and effect over time.
- Characteristics that combine to form Environmental metric group are unique because its formation is totally depended on those vulnerabilities which are based on the specific users' environment.

As a quick overview of the above described public security vulnerability repository following key points is listed:

- CWE deals purely with the flaws and their mitigation; it has nothing to do with the product type or version
- CVE gives an insight into the particular instance, its version, its vendor but not much interested in defining details of the underlying vulnerability.
- CCE focus on the specific configuration of any instance or product.
- CVSS is most important of all repositories as it indicates the severity level of vulnerability, irrespective of product or weakness type.

3.5. Used tools

In this section, the tools used in the proposed framework are going to be discussed in more details. This involves the use of different open-source, and custom-built tools and APIs.

3.6. Penetration testing tool

Tools designed for penetration testing can discover and exploit weak areas by simulation of attack scenarios. These security gaps may lead to misuse of login credentials, personal information related to health, property, identification, and credit card details. Data acquisition of this nature can have adverse business results [26]. Protective, harmful, lucid testing will help you decouple important business data in the future by reducing security gaps.

In our work, Netsparker was used as a pen-test tool as shown in Figure 10. It is a web-based vigorous scanner that helps in the identification of security flaws. It also suggests the best possible counteractive measures. This tool comes with both command line and the graphical user interface; it is scalable and provides integration few key features are listed in.

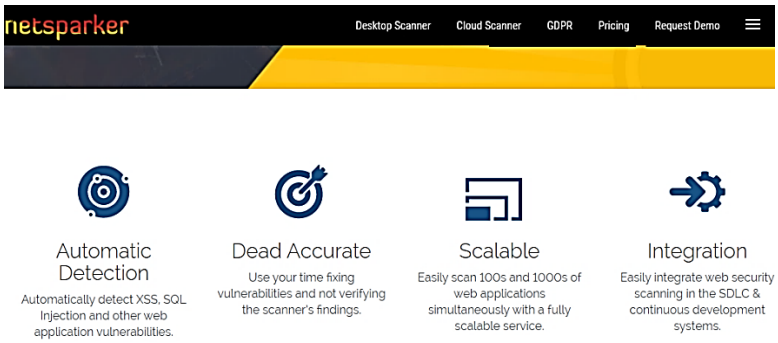

Figure 10. Features of netsparker

3.7. Clustering tool

In our study, the clustering of the dataset is a process that is used for optimization in the detection of structure or patterns in a set of data gathered from different sources. At present clustering is one of the most important text mining exploration directions for researchers. Although using this procedure might result in some loss of information but clustering technique simplifies the structure of dataset provided to be grouped in meaningful clusters and helps the user in the end to have a refined dataset for their further working. Commonly used algorithm for clustering techniques is K-means, SOM-Self organizing map, DBscan, hierarchal clustering, grid-based clustering.

The cluster performance depends on the interpretation that the related document belongs to the same cluster and share the same neighborhood, resulting to conclude it as; cluster techniques focus on the relevant contact element. Figure 11 depicts the clustering concept. In our study, we use the SPMF java-based tool for clustering.

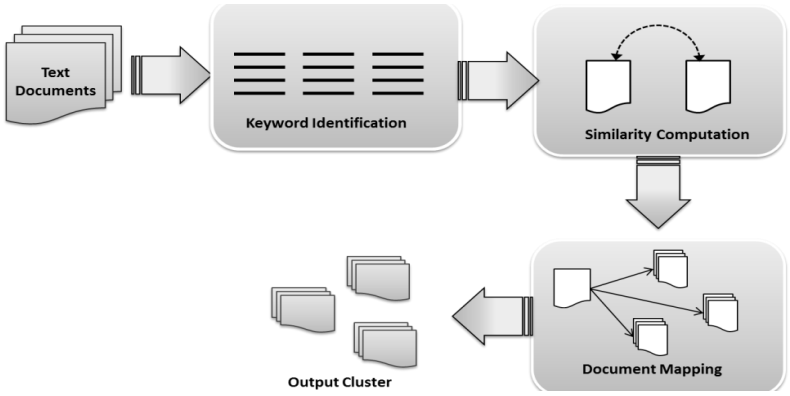

Figure 11. Cluster transformation

3.8. SPMF

It is an open-source data mining library, particularly used for pattern excavating its' a java-based tools and offers implementations of 150+ data mining algorithms [27]. These algorithms are used for following mining techniques; clustering and classification, time-series mining, sequential pattern, periodic pattern mining, high-utility pattern mining, itemset mining, episode mining sequential rule mining, association rule mining, and sequence prediction [28].

Our study uses one of its algorithms that are "TextClusterer" which takes the text file as input and produces the same as output. This algorithm-based tool work as follows

a. The input file is loaded and if required stop words and stem words are eliminated
b. tf*idf against each row of the input file is calculated
c. tf*idf value of each record is used to evaluate the similarity matrix.
d. Most similar records are marked as clusters initially.

e. By using transitive rule i.e., if a1, a5, a21 are most related and a5, a50 are most similar; then a1, a5, a21, and a50 are likely to be identical. This means that a1, a5, a21 and a50 are in the same cluster.
f. Then it will merge all the clusters based on the above rule for all the records.
g. Finally, the output file is generated containing text clusters.

This tool is very easy to install and use, it has no dependencies to other libraries. An updated version of SPMF is 2.30c released in March 2018 example shown in Figure 12.

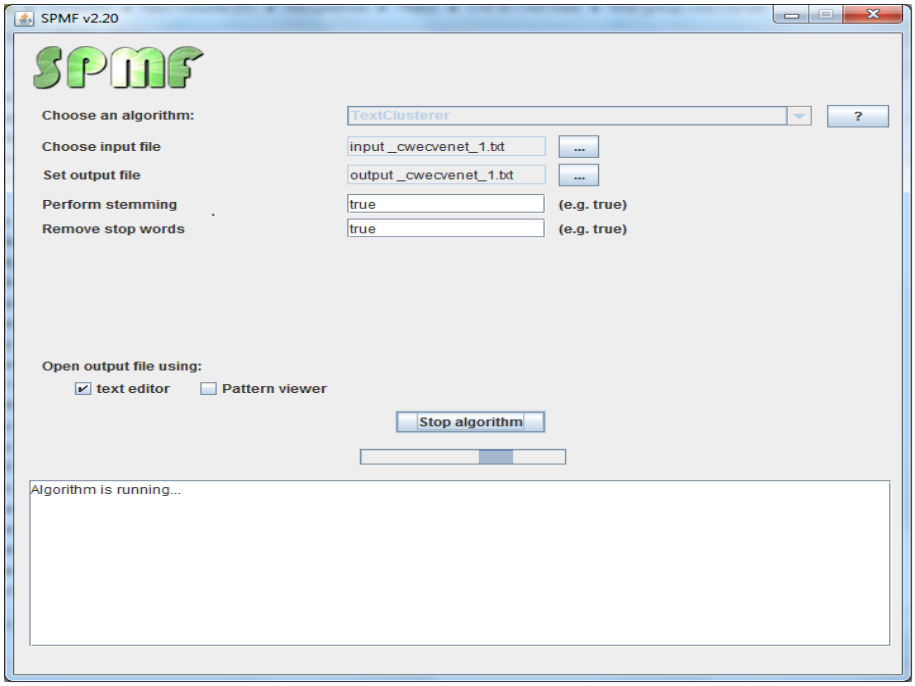

Figure 12. SPMF-clustering tool

3.9. Mapping tool

Mapping of data from one data model to another is the first step for data integration, and for data management and computation, it is essential to map data from sources with a particular destination. This mapping of data from source to destination represents the relationship of data, which further can be used for data transformation, data lineage analysis, or research work. In our study, we use mapping technique to consolidate multiple repositories in order to find best possible mitigation against maliciously reported issues and for this SimScore tool is used which is developed by students of the University of Mississippi.

SimScore is a tool developed on the .NET framework, its core functionality is to calculate the similarity index of provided folders using two mapping algorithms; if*idf and Jaccard coefficient. This tool helps us finding similarity between mitigation articles of public repositories and description of reported vulnerabilities [29].

TF*IDF is basically 2 terms TF and IDF. Term Frequency TF determines the occurrence of a particular term in the whole document, and IDF is inverse document frequency is the frequency of a particular word in the set of documents. Core working of these two terms is elaborated below:

TF: Term Frequency that is used to measure the frequency of a term with respect to its presence in a document, as all the documents are not of the same size there is a probability that term frequency varies with the change of document length. To normalize this concern TF is calculated by dividing a total number of terms in the document. TF (t) = (term t frequency/ total terms in a document).

IDF: Inverse Document Frequency is responsible to find the importance of a particular term in document whereas TF consider all terms equally important in a document. Stop words, parts of speech need to be controlled as they appear many times but do not have much significance in calculating IDF while rare terms need to measure. IDF (t) = log (Total documents / Number of documents with term t).

$$JSim\ (X_1, Y_1) = |X_1 \cap Y_1| / |X_1 \cup Y_1|$$

The working mechanism of the Jaccard coefficient or similarity can easily be seen in Figure 13. Jaccard similarity is the intersection of 2 sets X_1, Y_1 divided by the same sets.

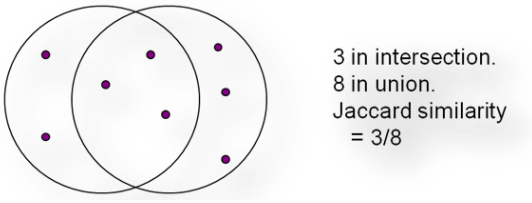

Figure 13. Jaccard index

3.10. Articles extraction tool

Open source article extractor tool is used to extract mitigation articles from public vulnerability repositories. Articles are extracted and saved in separate text files; file name is the same as that of the article identification number.

3.11. Case study

In this section, a case study is conducted where the proposed method is applied. The case study was done at a university with more than 9197 students and more than 600 faculty members. Multiple experiments are performed to provide evidence to support the proposed framework. The educational ERP under study is Oracle PeopleSoft Enterprise version 9.0 with Oracle PeopleSoft Enterprise PeopleTools version 8.53. The server is Weblogic server version 10.3.6.0 with the operating system as Oracle Linux version 6.0.

Phase 1: Public repositories and automated tools

Some data was gathered for the case study using penetration testing tool- Netsparker. The version number 4.7.1.12478 was used in this case study. Some configuration is needed before running the experiment. Figure 14 shows the steps of the penetration testing process.

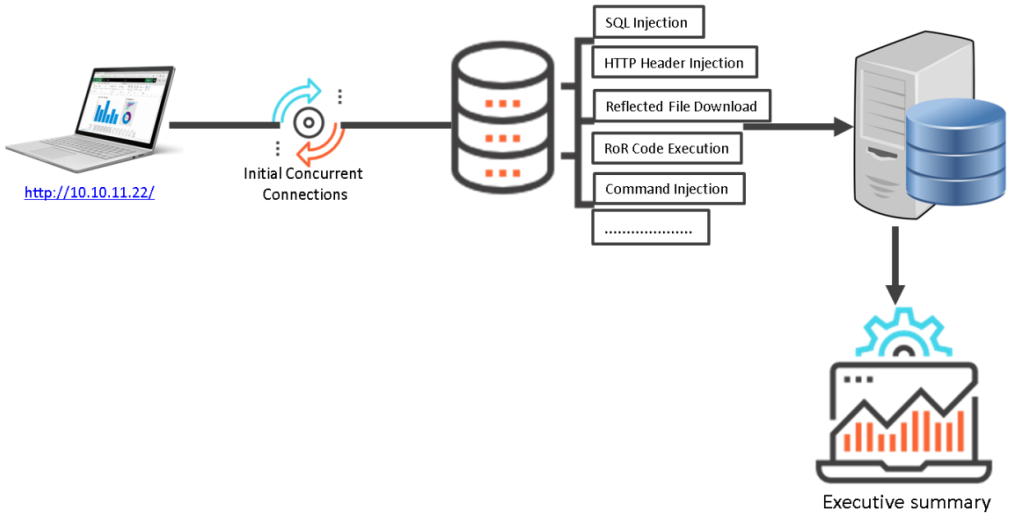

Figure 14. Flow of experiment 1.1

After completing the process of multiple output reports. The scanning took about two minutes where the tool sent 379 requests and found 10 issues. Figure 15 summarizes the result of the penetration testing. After that and in order to identify vulnerabilities against mentioned versions, all the reported issues for these versions were collected and stored in text files along with their CVE IDs and detail description. Table 1, depicts the total reported vulnerabilities found in CVE repository against the specified version.

93 vulnerabilities were found, application layer contains 7 issues, supported tools of the target ERP have 21 reported issues as shown in Figure 16, 15 problems lies at WebLogic layer and finally, 50 problems are related to the operating system.

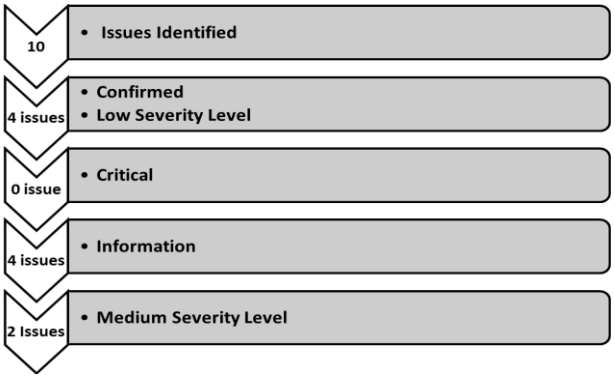

Figure 15. Netsparker result

Table 1. Version specific vulnerabilities

Product	Version	Vulnerabilities in CVE repository
Oracle PeopleSoft Enterprise	9.0	7
Oracle PeopleSoft Enterprise Peopletools	8.53	21
Weblogic Server	10.3.6.0	15
Oracle Linux	6.0	50

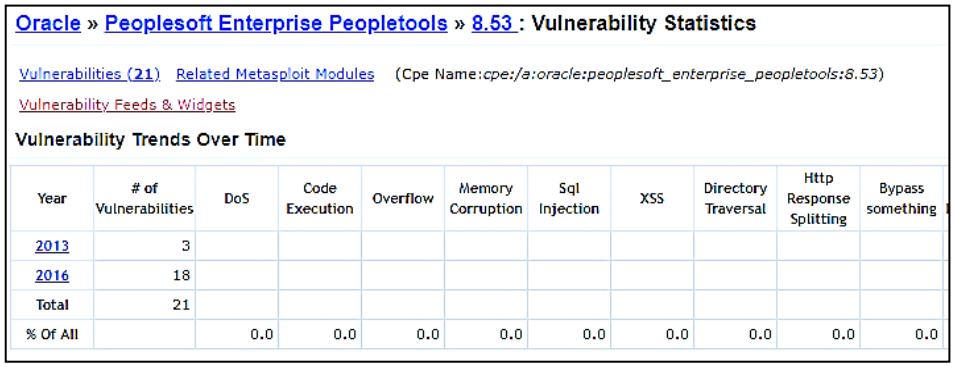

Figure 16. Version specific vulnerability statistic reported by CVE [30]

Phase 2: Dataset optimization and use of mapping algorithm

Configuration level vulnerabilities exist in enterprise solutions just like version-specific vulnerabilities. To identify these issues CCE repository is used. Detail of all these public vulnerability repositories is added in tools and knowledge base. Issues reported against this release are found in CCE repository, they are later stored in excel file. They include CCE-ID against each record, parameters details, their description, and navigation mechanisms. The Table 2 depicts total issues flagged against this release is 112. Results formulated from experiments conducted before are combined. They are collected from the Netsparker tool, CVE repository, and CCE repository.

Table 2. Configuration level reported vulnerabilities

Product	Version	Reported vulnerabilities in CCE repository
Oracle Weblogic Server 11g Release 1	10.3.1	112

Optimization and mapping are conducted using Clustering tool- SPMF and mapping tool- SimScore. Text clusters need to be generated as per the proposed framework and methodology, for clusters we need flagged vulnerabilities and mitigation articles. The output generated from previous experiments contain all the identified vulnerabilities.

Input file requires by SPMF tool contains 2 fields; 1st field represents row number and second store description of the corresponding field. In our case Row ID represent article number and vulnerability number, showing CWE ID articles and CVE / Netsparker vulnerability ID respectively. Now, this text file contains 100+ vulnerabilities and 700+ articles. The input parameters set for this experiment are shown in Figure 17.

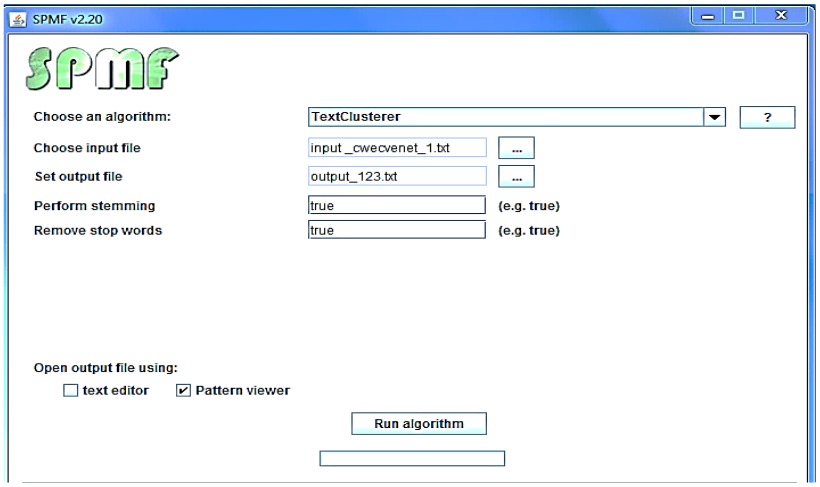

Figure 17. SPMF GUI interface

The run algorithm will generate the output file in the desired folder and opens in a text editor. The output file contains 2 columns; first display row ids and next shows cluster numbers that are formed on the bases of steps performs by the clustering. The Figure 18 depicts the findings of the clustering experiment. The output file was studied extensively and the following findings are produced:

- 100+ vulnerabilities plus 700+ articles in a file combing; 800+ rows were given as input to SPMFs' text clustering algorithm.
- 178 clusters are formed, 27 clusters are those in which flagged vulnerabilities and mitigation articles are combined.
- 48 most sensitive flagged vulnerabilities are able to find relevant mitigation articles in these 27 clusters.
- Most of the other reported vulnerabilities are addressed by Oracle security patches, bundle and version upgrades.
- These clusters are not only grouped on bases of similarity but also having the same vulnerability types (e.g., DoS, Memory overflow). All such articles plus vulnerabilities are grouped in one cluster respectively.

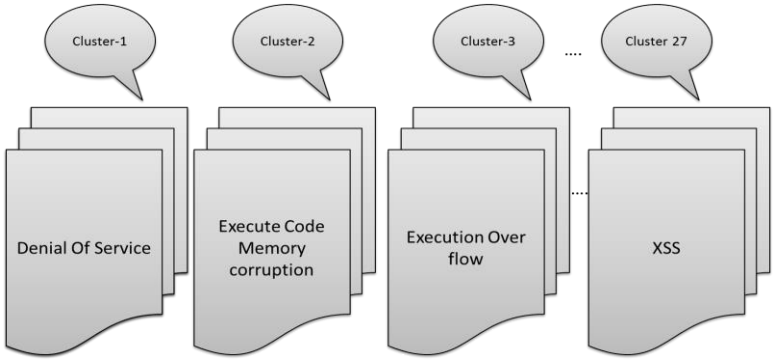

Figure 18. Clusters formed from reported vulnerabilities and related CWE articles

Optimization of a dataset is complete now the next step is to map reported vulnerabilities to mitigation articles of CWE. From now onwards Clusters are read as = Cx, Reported vulnerabilities as = Vi and Mitigation Articles as = Aj.

Phase 3: Implementation plan

In this experiment, mapping algorithms are used to map Aj with Vi grouped in Cx. As discussed in the proposed framework that this step will finally take us to suggest an implementation plan for that reported vulnerabilities can be fixed using this framework. 27 clusters from the previous output are used as input of this experiment. We need to use mapping algorithms for finding the best matching articles. SimScore tool is used for this task, it is based on TF*IDF and Jaccard index. As a proof-of-concept, we take C16, in which 7 reported vulnerabilities and 13 potential mitigation articles are present as shown in Figure 19.

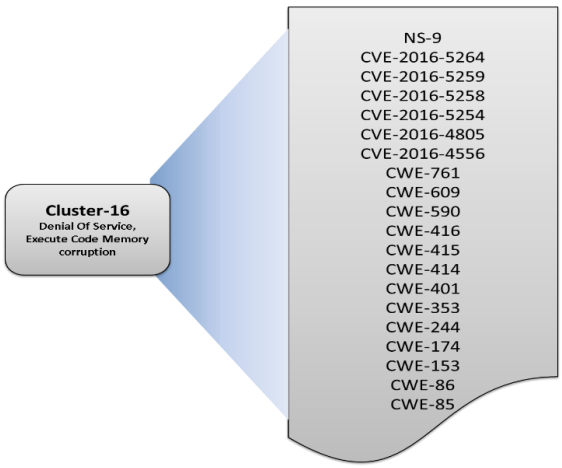

Figure 19. Cluster 16

All seven issues were stored separately in text files in which complete detail is mentioned, these files are then placed in a folder namely 'Source' shown in Figure 20.

- Repeat the same process with CWE articles of C16 and place them in the 'Destination' folder.
- Simscore tool is run by providing destination and source folder and finally output file "Similarity.txt" is updated.

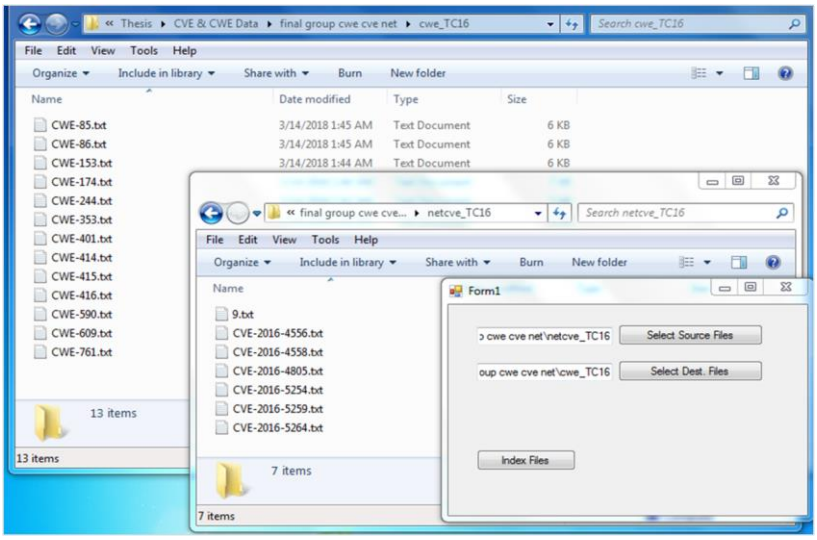

Figure 20. SimScore input screen

Mapping $V_i \rightarrow A_j$ is performed and the output file is generated which contains 4 fields; destination file name, source file name, TF*IDF similarity score, and Jaccard index score. C16 was used, that contains 7 vulnerabilities and 13 mitigation articles, by using mapping algorithm tool SimScore vulnerability files were mapped on CWE articles. As the output of this, we get TF*IDF score and Jaccard coefficient as similarity index of each file. We intend to elaborate the similarity index of reported vulnerabilities Vi of CX mapped with articles Aj, in which TF*IDF score and Jaccard coefficient are examined. This will present the accuracy and precision of these mapping algorithms which were responsible to find the most relevant articles from the knowledgebase.

4. CONCLUSION

Educational ERP applications are huge, complex, and consist of multiple components. These systems are built to build enterprise solutions such as Front-end application, Database Server, Web Server, Application server, compatible operating system, and other parts. The purpose of this study is to caution users to protect the security of entrepreneurial applications through a few experiments and providing guidelines and tools for the issue and security assessment of enterprise applications. This type of huge application controls money and resources and there may be a breach of security, our study help to control that perspective as well. By using this approach all issues in term of their solution are analyzed in a better way. No vulnerability left overlooked. Whereas this study will help implementers, client and vendors to adopt this solution framework while the deployment of educational ERP; while upgrading the version; integrating with any third party application and so on. Our future work will be an extension of this study by covering all those applications, which are integrated with educational ERP system at present, like learning management system - LMS, timetable scheduling system- Uni-Time, and online admission system.

REFERENCES

[1] A. Polyakov, "Practical pentesting of ERP's and business applications," *ERPScan* 2013.
[2] G. Tokdemir and N. E. Cagiltay, "Investigating the Relationship Between SLOC and Logical Database Measures to Improve the Early Estimation of Software Cost," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 29, pp. 401-413, 2019.
[3] O. PeopleSoft, "Securing your PeopleSoft application environment," *Oracle*, 2010.
[4] V. Kanchana and S. S. Ranjini, "Investigation and study of vital factors in selection, implementation and satisfaction of ERP in small and medium scale industries," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8(2), pp. 1150-1155, 2018.
[5] M. Alenezi and Y. Javed, "Open source web application security: A static analysis approach," *Proc. - 2016 Int. Conf. Eng. MIS, ICEMIS 2016*, 2016.
[6] Y. Wang and J. Xiao, "An intelligent model for vulnerability analysis using attack graph," *Proc. - 2009 Int. Forum Inf. Technol. Appl. IFITA 2009*, vol. 3, pp. 526-529, 2009.
[7] K. Hsu, et al., "Avoiding ERP pitfalls," *J. Corp. Account. Financ.*, vol. 17, pp. 67-74, 2006.
[8] M. L. Markus, "Technochange management: using IT to drive organizational change," *J. Inf. Technol.*, vol. 19, pp. 4-20, 2004.
[9] D. Chhillar and K. Sharma, "Proposed T-Model to cover 4S quality metrics based on empirical study of root cause of software failures," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9(2), pp. 1122-1130, 2019.
[10] F. Ferri, et al., "IBF: An integrated business framework for virtual communities," *J. Electron. Commer. Organ.*, vol. 10, pp. 1-13, 2012.
[11] J. Arnaud, *et al*., "Version Control and Patch Management of Protection and Automation Systems," pp. 12.80-12.80, 2014.
[12] Y. Javed and M. Alenezi, "Defectiveness Evolution in Open Source Software Systems," *Procedia Comput. Sci.*, vol. 82, pp. 107-114, 2016.
[13] M. Nadeem, *et al*., "A Method for Recommending Computer-Security Training for Software Developers: Leveraging the Power of Static Analysis Techniques and Vulnerability Repositories," *Proc. - 12th Int. Conf. Inf. Technol. New Gener. ITNG 2015*, pp. 534-539, 2015.
[14] P. Maggi, *et al*., "Vulnerability modelling for the analysis of network attacks," *Proc. Int. Conf. Dependability Comput. Syst. DepCoS - RELCOMEX 2008*, pp. 15-22, 2008.
[15] C. H. Lin, *et al*., "A study and implementation of vulnerability assessment and misconfiguration detection," *Proc. 3rd IEEE Asia-Pacific Serv. Comput. Conf. APSCC 2008*, 2008, pp. 1252-1257.
[16] F. Stouten, "Big data analytics attack detection for Critical Information Infrastructure Protection," Degree Project, Luleå University of Technology, Department of Computer Science, Electrical and Space Engineering 2016.
[17] J. Kim, *et al*., "Vulnerability discovery in multi-version software systems," *Proc. IEEE Int. Symp. High Assur. Syst. Eng.*, pp. 141-148, 2007.
[18] M. Alenezi and Y. Javed, "Open source web application security: A static analysis approach," *2016 Int. Conf. Eng. MIS*, pp. 1-5, 2016.

[19] K. Sivakumar and K. Garg, "Constructing a 'Common Cross Site Scripting Vulnerabilities Enumeration (CXE)' Using CWE and CVE," *Inf. Syst. Secur.*, pp. 277-291, 2007.
[20] https://cwe.mitre.org/about/faq.html
[21] M. Alenezi, *et al*., "Efficient Bug Triaging Using Text Mining," *Journal of Software*, vol. 8, pp. 2185-2190, 2013.
[22] http://cwe.mitre.org/data/definitions/798.html
[23] https://www.cvedetails.com/cve/CVE-2006-0584/
[24] S. Plansangket and J. Q. Gan, "A query suggestion method combining TF-IDF and Jaccard Coefficient for interactive web search," *Artif. Intell. Res.*, vol. 4, 2015.
[25] R. Wang, *et al*., "An improved CVSS-based vulnerability scoring mechanism," *Proc. - 3rd Int. Conf. Multimed. Inf. Netw. Secur. MINES 2011*, pp. 352-355, 2011.
[26] F. Caron, "Obtaining reasonable assurance on cyber resilience," *Manag. Audit. J.*, 2019.
[27] http://www.philippe-fournier-viger.com/spmf/index.php
[28] P. F. Viger, *et al*., "The SPMF open-source data mining library version 2," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9853 LNCS, pp. 36-40, 2016.
[29] S. Emani, "A Comparative Evaluation of Semantic Web Service Discovery: Algorithms and Engines," A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial, 2009.
[30] https://www.cvedetails.com/version/144760/OraclePeoplesoftEnterprisePeopletools8.53.html