WEB APPLICATION VULNERABILITY PREDICTION USING SOFTWARE METRICS AND TEXT MINING-A THREE-WAY DECISION BASED APPROACH

Shamal P K[1], Rahamathulla K[2], Ali Akbar[3]
[1]Assistant Professor in IT, College of Engineering Trikaripur, Kasaragod, Kerala
[2]Assistant Professor in CSE, Govt Engineering College Thrissur, Kerala
[3]Assistant Professor in CSE, Govt. Engineering College Wayanad, Kerala

Abstract
A number of software vulnerability prediction models are developed to forecast the vulnerabilities in the software. The errors in the prediction are the main problem of software vulnerability prediction model. This paper proposes on machine learning based approach that combines software metrics and text mining of source code methods. This approach is a two phase classification method. In the initial phase,a three-way decision classifier based on software metrics is used .The second phase consists of two components ,(i) A classifier based on text mining,(ii)A composer to combine the output of software metrics based classifier(first phase) and the output of the text mining based classifier. Usually classification errors occur for the data items near the classifiers boundary. This method handles this situation by using three way decision classifier. During the first stage, three way decision classifier classifies the data into any of the three classes; vulnerable, clean or deferment .The deferment class contains the data items near to the classification boundary. Thus it needs further examination for removing classification errors. So only these deferment modules are passed to second stage. The proposed system helps to improve the classification accuracy of the prediction model. The vulnerability status of the deferment class is predicted by combining the output of both software metrics based model and text mining based model. The proposed model improves the quality of software vulnerability prediction model. This method helps to reduce the time and effort required to build secure software.
Keywords: software vulnerability prediction, machine learning, text mining, software metrics.

I. INTRODUCTION

Software development is complex and time bounded activity. The errors in software will be detected and corrected various stages of software development cycle. Some of these errors may not be detected and that will lead to vulnerability. A software vulnerability can be seen as a flaw, weakness or even an error in the system that can be exploited by an attacker in order to alter the normal behavior of the system. Usually the goal of an attacker is to gain some privileges in the system to take control of it or to obtain valuable information for their own benefit. Most of the known vulnerabilities are due to improper handling of inputs. Software consists of number of components, such as programs, data GUI components etc .So it is impossible to scan all the components for vulnerability detection.The solution is software vulnerability prediction model . Software prediction model focuses on classifying software entities (components, classes, modules, etc.) as vulnerable or clean. Prediction models that identify vulnerability prone software components can be undergone through analysis for detecting and removing vulnerabilities. This would enable efficient allocation of testing resources and better informed decisions concerning release quality. Software prediction model help to reduce the time and effort required for the detection and removal vulnerability by

checking only files that are predicted as vulnerable by it.

The software vulnerability prediction model use different methods such as the software metrics as indicator of software vulnerability, tokens and it frequency in source code as indicators of vulnerability. Software metrics obtained from source code and development history are used to predict the vulnerability. The commonly used software metrics are lines of code, number of functions, cyclomatic complexity, Maximum nesting complexity, Halsteads volume ,Fan in, Fan-out, number of function call(external and internal).Predicting vulnerable software components is based on text mining the source code of the components. Each component is characterized as a series of terms contained in its source code, with the associated frequencies. These features are used to forecast whether each component is likely to contain vulnerabilities.

In normal binary classifier , classify the object in to two classes based on the probability value of the object and the threshold value of the classifier. For example if threshold value of a classifier id 0.5 then object whose probability value is greater than 0.5 is classified into one class and objects whose probability value less than 0.5 is classified into another class. Main miclassification errors are concentrated near to the 0.5.In three way decision based classifier, instaed of single value threshold is selected between two level. For example if threshold levels are 0.3 and 0.7 hen object with probability value greater than level 1,0.7 is classified as one class. And if the probability of object less than 0.3 is classified as another class. The objects with probability value in between 0.3 and 0.7 is classified as deferment

In software defect prediction ,a three way decision classifier based on software metrics classify the files in to three classes such as defect files ,defect free files ,and deferment files, such as it need further checking .The deferment files passed to second stage of classification. The second classifier is combination of classifier based on text mining and software metric.

II. LITERATURE REVIEW

Shin and Williams [1] used correlation between complexity, code churn and developer activity metrics with vulnerabilities. Three types of software metrics to build vulnerability prediction models: complexity, code churn, and developer activity (CCD) metrics. Complexity can make code difficult to understand and to test for security. Frequent or large amount of code change can introduce vulnerabilities. Poor developer communication can lead to insecure coding practices. This is based on hypothesis on different software metrics. Chowdhury and Zulkernine [3] proposed that code complexity, coupling and cohesion metrics could be used as indicator of the vulnerabilities. Complexity, coupling and cohesion metrics are used as early indicator of vulnerability. LOC, Number of function dened, cyclomatic complexity, Fan-in, Fan-out are used for build the model. If this prediction gives positive values then perform through inspection defective components for detecting and xing the vulnerabilities. Nguyen and Tran [2] proposed an approach based on dependency graphs to predict vulnerable components. These dependency graphs are based on the relationship among software elements (i.e., components, classes, functions, variables) which can be obtained from a static code analyzers (e.g., Doxygen) or from a detail design specication. Therefore, the model can be applied in both design phase and developing phase. The model is supported by an experiment on JavaScript Engine of Firefox. Riccardo Scandariato, James Walden, Aram Hovsepyan, and Wouter Joose[4]formulated an approach based on machine learning to predict which components of a software application contain security vulnerabilities. The approach is based on text mining the source code of the components. Namely, each component is characterized as a series of terms contained in its source code, with the associated frequencies. These features are used to forecast whether each component is likely to contain vulnerabilities. James Walden, Jeff Stuck- man and Riccardo Scandariato[5] compare the performance of prediction model based on software metrics with model based on text analysis, and found that text mining provided signicantly better recall performance. Yun Zhang, David Lo, Xin Xia, Bowen Xu, Jianling Sun and Shanping Li [6]combine software metrics and text features for vulnerable le prediction. They developed application VULPREDICTOR, predict vulnerability of web application Drupal,
PHP My Admin and Moo- dle effectively. Weiwei Li , Zhiqiu Huang a, and Qing L [7]developed a prediction model based on three way decision .It use two phased classier system. At the rst phase,les are classied in to vulnerable, defect free, and third class as either defect or

defect free ,that is it required a deferment. And in second phase is for classifying these les .

III. MOTIVATION

The existing software vulnerability prediction model based on either software metrics or text analysis method. Main problem in these method rate of error in prediction, the misclassification ,its level is not acceptable in software engineering even though it is common in all prediction model. If file is actually clean, but predicted as vulnerable , then it is simply wasting of time and resource for scaning the file for vulnerability. Similarly if a file is actually vulnerable, but predicted as clean then it will lead insecure software. It is observed that the errors in prediction is occurred at boundary, ie region near to the threshold value. So these region are classification error prone region so they undergo special treatment. So these components are passed to a second stage classifier. For this a three way classifier is introduced.it can handle the error near to the classification boundary Thus quality of prediction will improve.

IV. PROPOSED SYSTEM

The block diagram of proposed vulnerability prediction model is shown in fig.1.Major steps in the proposed system is given Algorithm 1.This is a two staged system.In first phase a three-way decision based system trained with software metrics and in second phase combination of software metrics and text based vulnerability prediction model.Major component of the proposed systems are(i) Three-way decision based classifier (ii) Tokenization (iii)Text mining based vulnerability prediction model(iv)Composer.

A. *Three-way decision based classifier*

This is a normal binary classifier. Normally binary classifier predicts whether the given software component is vulnerable or not. This is done based on the value calculated by the classifier and threshold value. If value is greater than the threshold value then classifier will return 1 indicate that it is vulnerable. If the value is less than the threshold it will return 0 indicate that it is clean. Consider if threshold is set to 0.5, then all the values above 0.5 are vulnerable and below 0.5 is clean. But it is noticed that errors in prediction is occur for the value near 0.5.That is value near the boundary has chance of erroneous prediction. To improve the quality of prediction, it is better to treat value near to threshold for further examination. To solve this problem a three way decision based classifier is used .This classifier classify vulnerable file or clean file or file its value near to the threshold is classified as a third class. This is called deferment. The deferment class need further examination.

Three-way decision based classifier is software vulnerability prediction model using software metrics. This is important to determine the lower and upper boundary of the deferment module (classifier's boundary width).Consider 'a' and 'b' are upper and lower boundary respectively. If the probability value of the classifier for test file ,Ps is greater than 'a' then it is classified as vulnerable .If the probability value of the classifier ,Ps is lower than 'b' then it is classified as clean. If the probability value of the classifier, Ps is between a and b then that file is classified as deferment. Initially the value of a and b are set to 0.4 and 0.6 .And these values are optimized based on accuracy. The deferment module is passed to second phase of the proposed system.

B. *Tokenization*

The source code of the deferment module is the input for the tokenization and out put is list of tokens and it's frequency. This term vector is act as input text based vulnerability prediction model. Tokens represent language keywords, punctuation, and other code constructs. A tokenizer program tokenize source file and output a list of tokens and their associated frequencies. This representation is known as a bag of words. The set of unique terms or tokens is the vocabulary of the developers. The tokenizer processes the set of tokens to eliminate unnecessary features. It ignores comments and whitespace. These tokens and it's frequencies of test file (defermnt) are passed as the input to the text mining based classifier.

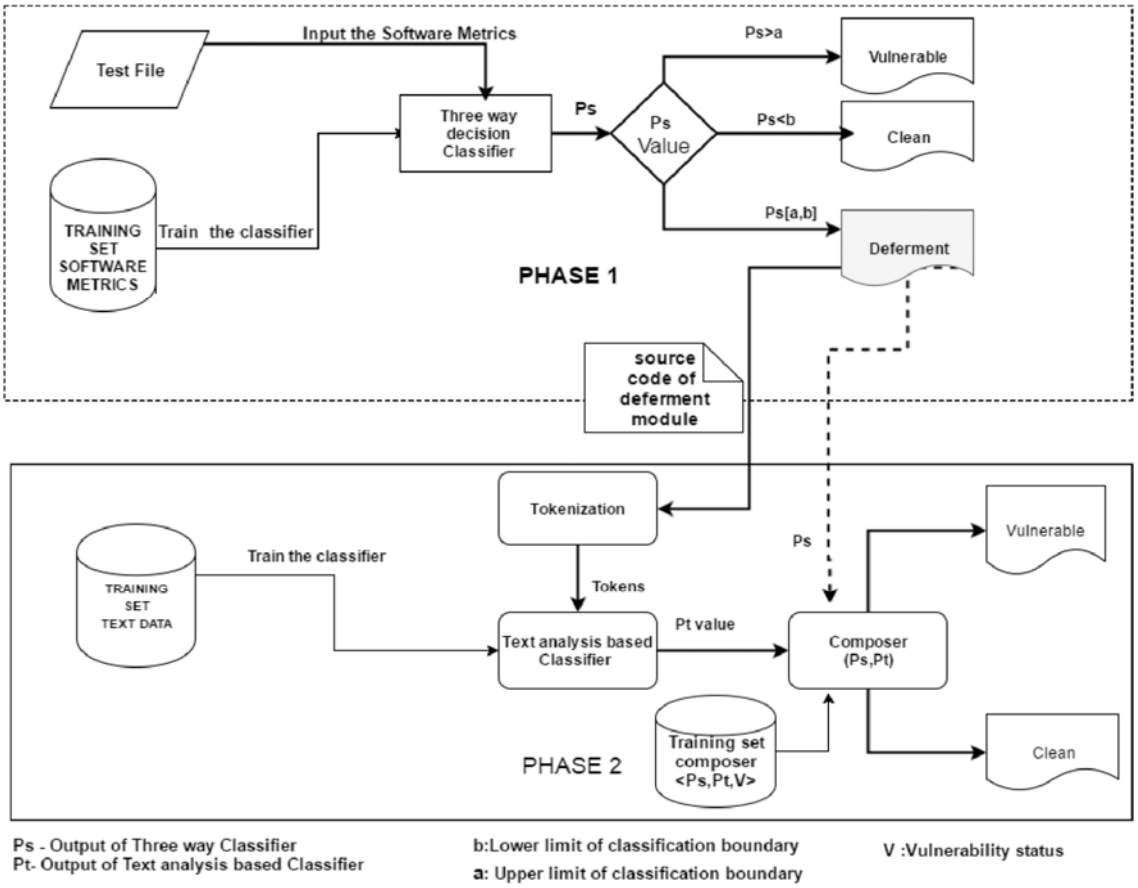

Fig. 1. Proposed system

c. *Text mining based vulnerability prediction model*

Text based vulnerability prediction model trained with tokens and frequencies of software components's source code.In text mining based model,each source file is represented as a term vector. Term vectors typically have a high dimensionality depending on the size of the application and the vocabulary used by the developers. These term vectors are then used as the predictors.Text mining based vulnerability prediction model produce probability value Pt for test file(deferment file),and it is pass to composer.

d. *Composer*

Composer is a meta classifier.This classifier is trained with probability values of software metric based classifier and text mining based classifier.Each file can be represented as < *Ps,Pt,V i* >.Ps is the probability value produced by three-way classifier for file i and Pt is the probability values produced by text mining based classifier for file i and Vi is the vulnerability status of the file i. This classifier combine the output of text based classifier and software metrics based classifier.It combine Ps and Pt value of test file (deferment file) and produce the vulnerability status such as clean or vulnerable

V. EXPERIMENT AND RESULTS

The experiments are conducted on dataset Drupal using R studio and Weka. Drupal is open software developed usinf PHP. Drupal is content management system. This dataset is the result of work done by James Walden et al[5].Drupal consists of total 202 files and in which 62 files are vulnerable.

Three-way decision based classifier is implemented by using the logistic regression .Random Forest classifier is used to implement text based classifier and composer.

Accuracy is the main evaluation measure and recall , precision and F score are used to compare the proposed system with other methods.

• Accuracy is the percentage of correct results.

Algorithm 1 Proposed system

1: Set upper boundary a and lower boundary b.
2: Software metrics based classifier generate probability value Ps for test file.
3: if *Ps* > *a* then
4: Print Vulnerable file
5: else if *Ps* < *b* then
6: Print Clean file
7: else
8: Classified as deferment file .
9: Tokenize the deferment module.
10: Pass the token and frequencies of deferment module to text based classifier and produce probability value Pt.
11: Pass Ps and Pt value as test data to composer. Composer produces output such as vulnerable or not.
12: end if

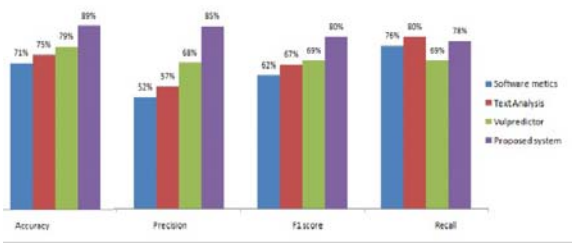

Fig. 2. Comparison with other methods

- Precision is the probability that a file classified as vulnerable is indeed vulnerable.
- Recall is the probability that a vulnerable file is classified as such.
- F score is the harmonic mean of precision and recall.

Table 1 show that 10-fold cross validation result of this prediction model. Result is the average of result obtained in each round. It is found that accuracy of the system is 89 %.Thus the error in prediction is less than 15 %.

The result is compared with exiting systems. This is done by result obtained by using exiting system with same dataset Drupal. The existing systems are software vulnerability prediction model using software metrics, software vulnerability prediction model using text mining, software vulnerability prediction model using combination of software metrics and text mining (Vulpredictor).It is observed that the precision accuracy recall and f measure of proposed system are better than that of exiting system. The comparison is shown in figure 2.

VI. CONCLUSION

Major problem of software vulnerability prediction model is classification error .Classification errors are concentrated at classifier's boundary region. Web application vulnerability prediction model using software metrics and text mining with three-way decision based approach. Quality of system can be improved by use of two stage classification based on three way decision classifier. Three way decision based classifier filtered out the deferment modules, so they are undergone further processing at second stage. Thus the classification errors can be reduced. This predcition model improves the quality of software development with less time and resources. This can handle the error at the decision boundaries through the use of three way decision based classifier.

Accuracy	89 %
Precision	0.85
Recall	0.78
F1-score	0.80

TABLE I. RESULT

REFERENCES

[1] Andrew Meneely, Laurie Williams,Yonghee Shin "Evaluating Complexity, Code Churn, and Developer Activity Metrics as Indicators of Software Vulnerabilities", IEEE Trans. Softw. Eng., vol. 37, no. 6 , pp.,772-787 Nov.Dec. 2011

[2]Le Minh Sang Tran,Viet Hung Nguyen, "Predicting vulnerable software components with dependency graphs",Proceeding International Workshop Security , 2010, p. 3.

[3]M.Chowdhury and Zulkernine,"Using complexity, coupling, and cohesion metrics as early indicators of vulnerabilities ",J. Syst. Archit. vol. 57, no. 3, 2011.

[4]A. Hovsepyan,J. Walden,R. Scandariato, and W. Joosen, "Predicting Vulnerable Software Components via Text Mining ", IEEE Transactions on Software Engineering, Oct. 2014.

[5]R. Scandariato and Walden, J. Stuckman, , ", Predicting Vulnerable Components: Software Metrics vs Text Mining, ",IEEE 25 th International Symposium on Software Reliability Engineering (ISSRE),2014,pp 23-33.

[6] Yun Zhang, David Lo, Xin Xia, Bowen Xu, Jianling Sun, Shanping Li," Combining Software Metrics and Text Features for Vulnerable File Prediction,"20th International Conference on Engineering of Complex Computer Systems,2015.

[7] Weiwei Li a,b, Zhiqiu Huang a, Qing Li" Three-way decisions based software defect prediction, "Knowledge-Based Systems (2016)