Fuzzy interference approach based prioritization of security requirements

S. Mithun brindha[1], Dr.G. Singaravel[2]

[1]*PG Scholar, M.Tech IT,* [2]*Head of the Department, KSR College of Engineering, Thiruchengode, Namakkal, Tamil Nadu, India*
mithunbrindhame@gmail.com[1], gsingarvel@gmail.com2

Abstract

Objective: The main objective goal of this work is to analysis and identifies the security requirements on behalf of the software development. Improve and prevent the software development process from the failure by ranking and prioritizing the security requirements based on risk level.
Methodology: The fuzzy interference based approach is introduced to prioritize the security requirements that are more vulnerable to the system. This methods work with the aims of prioritizing the tasks based on the security risk level. The threat with more risk level will be given more priority and that will be concentrated more at the tie software development.
Results: The experimental tests conducted were proves that the proposed methodology can provide the better result than the existing methodology in terms of improvement of well software development
Conclusion: From the findings, it can demonstrated that the proposed methodology is improved in terms of the accuracy

Keywords: Software failure, Security Requirements, Fuzzy Model, Prioritization

1. Introduction

Software engineering is essential in creation of the innovative products. The software creation can be done effectively by following the software development life cycle which consists of 5 phases for the software creation. The processes done in these phases are explained as follows:

1.1. Requirement elicitation

Requirement elicitation is the process of gathering the needed things which are required for the software implementation. These requirements will be gathered from the different and multiple sources based on the type of product that they are going to develop. This is base of the produce development which needs to be done more carefully for eliciting the project related requirements. The requirements will be decided based on the software to be development by using previous knowledge history.

1.2. Designing process

After requirement elicitation, gathered requirements need to be arranged in the proper manner for identifying the overall design of the product that we are going to develop. The software design is used to give the idea of software development process by using which knowledge; the well effective software will be developed. The overall design of the software which is going to developed will give the general idea of software implementation.

1.3. Development process

In this stage, the software design which is made previously with the help of requirements that are gathered, the development of software will be started. This software development phase consists of the real time implementation of the software based on the user requirements. Software implementation has to be done with more consideration for avoiding the future failure in case of any corruption. The well effective software development depends on the characteristics of the software requirements that are gathered previously and the software design which was made previously. The error free zone software needs the details of good software.

1.4. Testing process

After implementation of software, the developed software need to undergone testing phase in order to prevent it from the software failure at the time of real usage. The testing phase will test about the functionality and the objective of the software in order to make sure that the software is assures to provide the true output for any kind of information which is given as input. This testing phase is used to assure that the software is developed without any corruption which can lead to a failure of the software at the time software usage in real time scenarios. The testing phase may lead developed software to more iteration for improvement of it in case of any software failure.

1.5. Maintenance phase

In this phase, after delivering the software to the customers functionality of the software is maintained for the future development in case of user requirements. And also any corruption in software in future can be identified and resolved with the help of maintenance phase. The maintenance phase is used to detect the usage scenario of the software failure in case of any corruption of software.

These are all the software maintenance phases, which are followed to develop the well effective software which can be handled effectively with the help of the good requirements. However in these methodologies of software development, the failures are detected and handled at the time of maintenance phase only. But it is one of the limitations of the software which may lead to the failure of the software. And also there is no use I finding about the failure details after completion of software development which may lead to the more cost and time consumption for correcting or reproducing the software to avoid the failures. The software failures need to be identified previously before software development, so that the software with error free zone can be assured for the users. And also this may lead to the conservation of time cost of the software developers.

The main contribution of this word is to identifying and eliciting the software requirement before initialization of the software development. The gathered security level requirements are prioritized before developing the software, so that the threat with more risk level can be given more importance. The threats are prioritized in this work by using the methodology called fuzzy interference approach and then the threats are handled with the consideration of the more requirements.

In [1] discussed a multiple ways for constructing the software without error and also the good base software. The goods software is differed from other softwares in term of its security level which is defined in terms of the software failure and it is occurred at the time of the error free zone.

In [2], discussed a ways for building a secure and reliable software with the consideration of the software development. The reliable software development can be obtained with the consideration of the requirements which are gathered from the different phases in terms of previous history of similar type of software.

In [3], a novel approach is introduced to gather the software requirements from the various phases which intend to produce good quality software. Good quality software can be obtained with the consideration of the complete software requirements. However it is more difficult to gather the software requirements which are more relevant to the software which is going to be developed. In this work, personal driven approach is introduced which aims to satisfy the users with the requirements consideration which was gathered from them. To achieve this user interaction interface is created which is named as the persona, and through which the user requirements are gathered and processed effectively.

In [4], agile methodology is introduced to produce more secured software. It leads to effective implementation of the software development which is based on the incremental and iterative software development. This methodology will gather the requirements at each and every phase at the time of software development in order to improve the software. The software development may lead to a successive iteration of the every cycle by gathering the requirements based on the software stage.

In [5], discussed a novel approach for gathering the requirements in terms of reuse methodology. The requirements gathered for the software development could be maintained in the database so as to use further for other similar types of software. Also the characteristics of the current software could be gathered and stored in database to maintain good configurable software.

3. Security requirement collection

Security requirements gathering process before the initialization of the software is the important framework which is used to avoid the software failure considerably than the security requirement collection after software development. The security requirements related to out software which is going to be developed need to be gathered effectively in order to make them suitable for the software development process. This is done with the help of early security knowledge from the previously developed softwares. It is used to overcome the problem of gathering the security requirements in terms of software to be developed at the time of requirement gathering phase and provide it to the software developers integrated with the software development phase. And also these requirements are prioritized in this work to give the more importance to the threats which is having more risk level. The high level of security requirement gathering process is in following phases:

- Finding the threats and assets
- Analysing the relationship present among threat and vulnerability
- Calculating the risk level
- Prioritizing the threats
- Security requirement evaluation

These are all the streps which were followed to capture the security requirements which will be considered at the time software implementation phase. These processes will be done along with the requirement gathering phase. The step by step processes of security requirement analysis are explained in the detailed manner as follows:

3.1. Finding the threats and assets

For the well analysis of security requirement, one needs to know the type of software that we are dealing with. This is done by analysing the nature of software that we are going to handle. After analysing the nature of software, the similar types of software that has been developed in the previously will be found. Then those softwares will be analysed to gather the details about the types of fault that has occurred before while the development of the corresponding software. Likewise the assets will be identified that is the function which is the base for the software will be identified. After identification of assets, the possible threats which may occur due to the asset will be analysed and identified. These threats are used to capture the requirements which can prevent the software from the failure. Before that every type of possible threats that can be occurred due to the assets that are present in software will be analysed and handled. This is done through the softwares which were developed with the same concept as like the current software which is going to be developed.

3.2. Analysing the relationship present among threat and vulnerability

After analysing the threats which can lead software to the failure state, the vulnerabilities of the corresponding threat will be analysed. The vulnerability identification is used to prevent the software from the failure by given more importance to the threat with more vulnerability. That is vulnerability is the effect of the system failure which may lead to the huge failure because of its weakness. The relationship among the threat and the vulnerability is identified with the help of analysis of the security catalogue. Security catalogue is used to give the details about the software based on the user requirement. From the security catalogue information one can understand the way of preventing the software failure or type of software threat.

Based on such information the security requirement will be gathered with the help of the security catalogue in order to avoid the risk. The gathered requirements will then be used to build the software without failure.

3.3. Calculating the risk level

After gathering the requirements, the risk level of every threat will be calculated in order to evaluate the seriousness. This is done with the help of the impact level of the security requirement and the software that is going to be developed. This is done via calculating the risk level. The risk level calculation is done by using the following equation:

$$\text{Risk} = \text{Impact} * \text{Probability}$$

The impact of the software is analysed based on the damage level which can occur and the reproducibility of software. The probability defines the software success rate which can be elicited using the requirements which are gathered.

3.4. Prioritizing the threats

After calculating the risk, prioritization will be done based on the fuzzy interference rule. The prioritization using fuzzy interference method is used to increase the accuracy level. By using this prioritization method, the threat more prioritization can give more important in order to prevent the software from dangerous corruption. The fuzzy interference rule engine is works as follows: Risk level accession is done by proceeding following steps: Those are:

- Defining input and output variable
- Designing the fuzzifying linguistic variables
- Developing fuzzy rule base
- Aggregating output
- Defuzzification

i. Defining input and output variable

In this phase, the input for the fuzzy interference engine will be defined to produce an optimized output. The inputs are taken as the vulnerability of threat, damage level and reproducibility factor.

ii. Designing the fuzzifying linguistic variables

Here the fuzzy membership value will be calculated as per the crisp inputs which have given previously. Based on the fuzzy membership values, the input values will be taken and then the values will be adjusted in order to increase the performance.

iii. Developing fuzzy rule base

In this phase, fuzzy rule base will be constructed which intends to creates a relationship interference between the input variables and the crisp values. The relationships will be identified with the consideration of the given inputs and then it will be processed further to evaluate the performance. It is achieved with the consideration of the if then rules.

iv. Aggregate output

In this phase, outcomes from all the phases will be aggregated together to identify the fuzzy linguistic differentiation. Based on this aggregation output, the final decision will be made for the appropriate output.

v. Defuzzification

In the Defuzzification phase, the final output will be generalized by integrating all outputs into a single output. This is done via the defuzzification rule which intends to generate a more rules in regards to the fuzzification process.

The final output of this phase will results in a risk value and the priority level of each and every threat. Based on this result, threat prioritization will be done and finally the tasks will be presented with the consideration of all parameters.

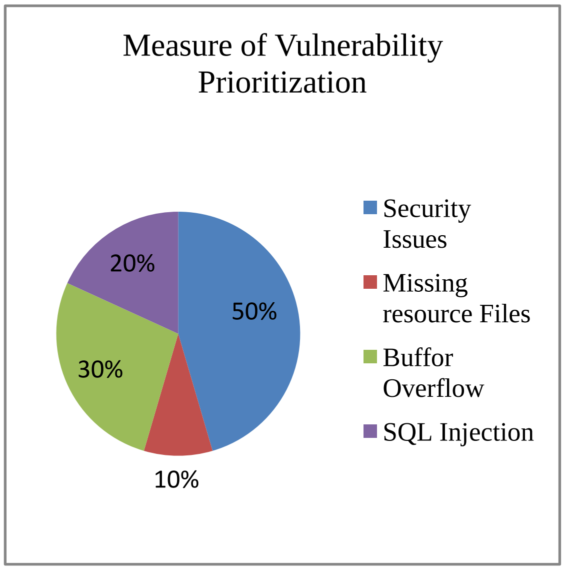
4. Experimental results

In this section, performance evaluation is done to prove the effectiveness of the proposed work by evaluating the security level. The prioritization based on the vulnerability level is calculated to prove the effectiveness of the proposed work. This algorithm is used to prove the effectiveness of the algorithm in terms of both metrics.

4.1. Vulnerability prioritization

The different types of vulnerabilities which has been identified during the requirement elicitation phase will be prioritized and based on that value risks are handled. The comparison is shown in Figure 1.

Figure 1. Measure of Vulnerability Prioritization



5. Conclusion

Security elicitation is done to implement the software with better security with the avoidance of the software failure. The proposed work is to prioritize the security threats based on the security level. Then the security requirements with high priority is processed with more consideration and handled effectively than the other security threats. The experimental tests conducted proved that the proposed methodology can provide the better result than the existing methodology in terms of vulnerability prioritization. Also the proposed work reduced software failure in terms of performance metrics.

6. Reference

1. Gary McGraw. Building secure software: Better than protecting bad software. *IEEE*. 2012; 19(6).
2. Barbara Fichtinger, Frances Paulisch, Peter Panholzer. Driving secure software development experiences in a diverse product environment. *IEEE Computer and Reliability Societies*. 2012; 10 (2), 97-101.
3. Jane Cleland-Huang. Meet elaine: A persona driven approach to exploring architecturally significant requirements. *IEEE computer society*. 2013; 30 (4), 18-21.
4. Lotfi ben Othmane, Pelin Angin, Harold Weffers, Bharat Bhargava. Extending the agile development approach to develop acceptably secure software. *IEEE*. 2014; 11(6), 497-509.
5. Laurent A. Hermoye, Axel van Lamsweerde, Dewayne E. Perry. A reuse-based approach to security requirements engineering. Proceedings of 9th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'03), 2003, 1-10.

The Publication fee is defrayed by Indian Society for Education and Environment (iSee). www.iseeadyar.org

Citation:

S. Mithun brindha, Dr.G. Singaravel. Fuzzy interference approach based prioritization of security requirements. *Indian Journal of Innovations and Developments.* 2014; 3 (4), 80-84