Anomaly Detection System in SDLC using Data Mining and Fuzzy Logic

Kanchan Arora, Harmanpreet Singh

Abstract- In the software life cycle, it is essential to protect corporate resources. Security audits and QA testing happen at the end of the development cycle where issues are most expensive to fix and when developers are focused on getting the release out and moving on to the next release. Audits are typically done late in the cycle to avoid having security experts review and re-review code that is likely to change before release. Due to its criticality, security should be integrated as a formal approach in the software life cycle in each and every phase. It includes the critical areas of requirements analysis and specification, design and code issues, and maintenance. We critically focus on the analysis of security and classification of attack pattern for the software life cycle. The key ideas are to use data mining techniques to discover consistent and useful patterns of system features that describe program and user behavior, and use the set of relevant system features(attribute selection by support vector machines) to compute classifiers(fuzzy inference and defuzzification methods) that can recognize anomalies and known intrusions.

Index Terms— Apriori Algorithm, Attribute selection, Defuzzification, Fuzzy Inference, Fuzzy Associatio Rules, Item-set, Support vector machine(SVM)

1 INTRODUCTION

The paper revolves around the software life cycle and the concern of protecting its resources. Data mining technique is used to pre-process the attributes collected from the packet headers through network and the packet payload plus the code issues. It selects the relevant attributes by outlier analysis. It computes the association rules based on apriori algorithm. Thereafter it generates the fuzzy association rules based on min-max derivation that is if the range falls within the min and max frequency then the attack is severe else attack is solvable. The inputs are given into fuzzy inference system proposed by mamdani. The fuzzy inference rules are generated. The antecedent lies with the "if" part and the consequent lies in the "else" part. The output of the fuzzy inference is either high or low which is the consequent part. It may or may not be defuzzified to a crisp value.

1.1 Attribute Selection and testing relevance

The set of attributes provided to the Data Analyzer is a subset of all possible attributes pertaining to the information contained in requirement analysis and design specification documents as well as the attributes contained in the packet pay load and packet headers. We retrieve the polynomial data set. The coefficients of a hyper-plane calculated by an SVM (Support Vector Machine) are set as attribute weights .The Weight by SVM operator is applied on it to calculate the weights of the attributes. All parameters are used with default values. The normalize weights parameter is set to true, thus all the weights will be normalized in the range 0 to 1. The sort weights parameter is set to true and the sort direction parameter is set to 'ascending', thus the results will be in ascending order of the weights. The select by weight operator calculates the relevance of the attributes by computing for each attribute of the input example-set the weight with respect to the class attribute. The select by weight operator selects only those attributes of an input example-set whose weights satisfy the specified criterion with respect to the input weights.

1.2 Data miner – Computes association rules based on apriori algorithm

The Apriori algorithm proposed by Agrawal et al. 1996 is an influential algorithm that can be used to find association rules. In this algorithm, a candidate k-itemset (kP1) containing k items is frequent (i.e., frequent k-itemset) if its support is larger than or equal to a user-specified minimum support. The well -known apriori algorithm[3] also says that a subset of a frequent item set is also frequent. We generate frequent item sets to generate association rules.

2 OUR APPROACH

2.1 Generating fuzzy association rules

Fuzzy sets are used to model the quantities of items in an association rule, e.g., large amount of money[1]. A linguistic representation describes quantities of an item in a way that is more interpretable and comprehensible for humans. Fuzzy-association rules were introduced with the F-APACS algorithm[2] to express quantitative attributes with linguistic labels in a way that is more natural to human reasoning and to overcome issues with discovering

- *Kanchan Arora is currently working as Software Engineer at Cerner, Bangalore, India. PH-08870511317. E-mail: kanchan719@gmail.com*
- *Harmanpreet Singh is currently working as Software Engineer at Mahindra Comviva, Bangalore, India. PH-09417631417. E-mail: rajan_harman@yahoo.co.in*

rules because of the crisp boundaries of attribute intervals. This represented quantitative attribute values of rules with linguistic labels modeled by fuzzy sets enhances the interpretability and handles anomalies in software resources to a greater extent.

2.2 Fuzzification and Defuzzication based on fuzzy inference system proposed by Mamdani

A fuzzy inference system (FIS)[4] proposed by Mamdani et al. is a system that uses fuzzy set theory to map inputs (features in the case of fuzzy classification) to outputs (classes in the case of fuzzy classification).It is used to simulate human thought for detecting intrusion.

To compute the output of this FIS given the inputs, one must go through six steps:

I. Determining a set of fuzzy rules

II. Fuzzifying the inputs using the input membership functions,

III. Combining the fuzzified inputs according to the fuzzy rules to establish rule strength.

IV. Finding the consequence of the rule by combining the rule strength and the output membership function,

V. Combining the consequences to get an output distribution, and

VI. Defuzzifying the output distribution (this step is only if a crisp output (class) is needed).

2.3 Creating fuzzy rules

Step1. ***if*** *code issues are* ***high*** *and probability of sql injection is* ***high then*** *vulnerability of life cycle is* ***high***. There would have to be membership functions that define what we mean by high code issues (input1), high sql injection (input2) and a vulnerable software development life cycle (output1). This process of taking an input such as code issues and processing it through a membership function to determine what we mean by "high" code issues is called fuzzification.

Step 2. Fuzzification using membership rules: The Fuzzy "and" is written as:

$$u_{A\cap B} = T(u_A(x), u_B(x))$$

(Minimum of a and b)

where μA is read as "the membership in class A" and μB is read as "the membership in class B".

Fuzzy "or"

The fuzzy "or" is written as:

$$u_{A\cup B} = T(u_A(x), u_B(x))$$

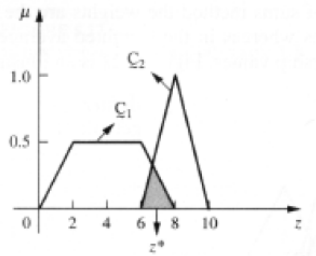
(maximum of a and b)

Step 3. Compute the rule strength using the above process.

Step 4. Clip the output membership at the rule strength.

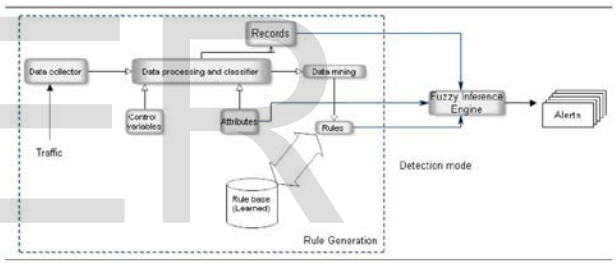
Step 5. The outputs of all of the fuzzy rules must now be combined to obtain one fuzzy output distribution.

Step 6. Center of sums (Defuzzification)- This method employs the algebraic sum of the individual fuzzy subsets instead of their unions. Calculations are very fast.



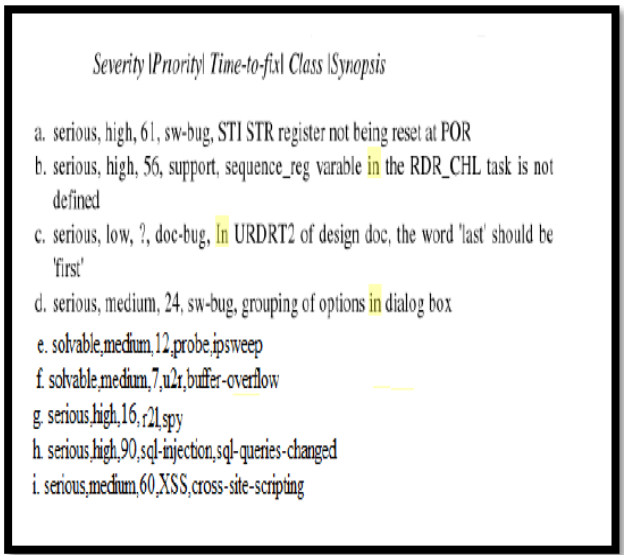
$$z^* = \frac{\int_Z z \sum_{k=1}^{n} \mu_{C_k}(z)\, dz}{\int_z \sum_{k=1}^{n} \mu_{C_k}(z)\, dz}$$

3 ARCHITECTURAL DIAGRAMS



4 NUMERICAL CALCULATIONS

4.1 Dataset Ready for mining



4.2 Rules

I. We use classification rules to find out to which class the particular testing data set belongs to by using the training data set.
II. We find the frequent words used in the synopsis and class using apriori algorithm.

If severity=critical and time-to-fix=24 days and priority is medium then class='sw-bug'. With Confidence=75% and support=2%, if they have severity equal to critical and priority medium ,it may take 24 days to fix the problem.

4.3 Applying apriori to synopsis and class

Step 1-

Attributes	Attack Category
Back	DOS, normal , sw-bug
Buffer_Overflow	U2R,R2L,normal,XSS,sw-bug
Guess_Password	R2L,normal,Sql_injection
Neptune	DOS, normal, probe, sw-bug
Smurf	DOS,R2L,sql_injection
Teardrop	Probe,R2L,DOS,SQL_injection
ftp_write	U2R,normal,SQl_injection
Load_module	R2L,U2R,DOS,sw-bug
Multi-hop	Normal, DOS
Perl	Probe,U2R,normal,XSS,sw-bug
Rootkit	DOS, normal
Spy	U2R,probe,normal,sql_injection
Ipsweep	DOS,normal
Sql_queries_changed	Sql_injection,R2L
Cross-site-scripting	XSS,R2L

Table 1: Applying Apriori Algorithm

Step 2-
Scanning the dataset

Item-set	Support
DOS	8
R2L	7
Probe	4
Normal	10
U2R	5
XSS	3
SQL_INJECTION	6
SW_BUG	5

Table 2: Scanning dataset

Step 3-
Calculating the frequent item-set from this data set

Support=30%
Reason: The probability that any two or more attacks can occur simultaneously on the same module is very less. That is x/15*100=30, Therefore x=4.5, we take the approximate value that is 5.

Step 4-
Calculating 2-frequent item-set.

We remove all the attributes whose support is <5.

Item-set	Support
DOS,R2L	2
DOS,PROBE	1
DOS,NORMAL	5
DOS,U2R	1
DOS,XSS	0
DOS,SQL_INJECTION	1
DOS,SW_BUG	3
R2L,PROBE	1
R2L,NORMAL	2
R2L,U2R	2
R2L,XSS	2
R2L,SQL_INJECTION	4
R2L,SW_BUG	2
PROBE,NORMAL	1
PROBE,U2R	2
PROBE,XSS	1
PROBE,SQL_INJECTION	2
PROBE,SW_BUG	2
NORMAL,U2R	2
NORMAL,XSS	2
NORMAL,SQL_INJECTION	3
NORMAL,SW_BUG	4
XSS,SQL_INJECTION	0
XSS,SW_BUG	2
SQL_INJECTION,SW_BUG	0

Table 3: 2-Frequent item-set

Step 5-
Frequent item-set

Item-set	Support
Dos,Normal	5

Table 4: Frequent item-set

Step 6-
Generating association rules

Dos-Normal=support{Dos,Normal}/support{DOS}=5/8=63%
Normal-DOS=support{DOS,Normal}/support{normal}=5/10=50%
Minimum confidence=55%
Set with the strong association rule is {DOS-NORMAL}

Step 7-
Deriving max-min deviation values

{DOS-NORMAL}: {1-5}
Min(freq)=f(min(freq(x))); Max=f(max(freq(x)));
Range={Min-Max}

Step 8-
Finding fuzzy rules

i. If DOS is greater than 5 , then the attack is severe.
ii. If DOS is in between 1 and 5, the attack is solvable.

Step 9-
Writing linguistic hedges

i. If DOS is greater than 5 , then the attack is high
ii. If DOS is in between 1 and 5, the attack is solvable.(apply and rule)

Step 10-
De-fuzzify

1. Convert linguistic hedges high to 1 crisp value.

5 CONCLUSIONS

In this paper, we propose a learning algorithm that can select a set of attributes from a given data set based on weight by SVM technique and then classify into fuzzy rules based on the processing of the Apriori algorithm and application of fuzzy inference engine to detect the anomalies in the software development process thereby alleviating the security issues.

References

[1] Hu, Y.-C.; Chen, R.-S. & Tzeng, G.-H. Finding fuzzy classification rules using data mining techniques Pattern Recogn. Lett., Elsevier Science Inc., 2003, 24, 509-519
[2] Dickerson, J. E. and J. A. Dickerson Fuzzy Network Profiling for Intrusion Detection. 19th International Conference of the North American Fuzzy Information Processing Society,2000,301-306.
[3](http://www.academia.edu/208304/Fuzzy_Intrusion_Detection_System_via_Data_Mining_Technique_With_Sequences_of_System_Calls)
[4] Deepa, S. S. S.Principles of Soft computing,Wiley India, 2012
[5] Williams, G. J.edited by Graham J. Williams, S. J. S. (Ed.)Data Mining: Theory, Methodology, Techniques, and Applications , Springer, 2006
[6] Shaik Akbar, Dr.K.Nageswara Rao, Intrusion Detection System Methodologies Based on Data Analysis International Journal of Computer Applications (0975 – 8887) Volume 5– No.2, August 2010.
[7] Anderson, P. Computer security threat monitoring and surveillance. Fort Washington.1980.
[8] Peyman Kabiri and Ali A. Ghorbani. "Research on Intrusion Detection and Response: A Survey". International Journal of Network Security, Vol.1, No.2, PP.84–102, Sep. 2005.
[9] Anazida Zainal, Mohd Aizaini Maarof and Siti Mariyam Shamsuddin "Ensemble Classifiers for Network Intrusion Detection System", Journal of Information Assurance and Security 4 2009 217-225.
[10] Lee, W. A "Data Mining framework for construction features and models for intrusion detection systems", Columbia University.2001.
[11] Manganaris, S.. "A data mining analysis of RTID alarms." Computer Networks2000. 34(4): 571-577.
[12] Stolfo, S. J,. "Data mining-based intrusion detectors: an overview of the columbia IDS project." ACM SIGMOD2001. 30(4): 5-14