Advances in Computer Science and Information Technology (ACSIT)
Print ISSN: 2393-9907; Online ISSN: 2393-9915; Volume 2, Number 1; January-March, 2015 pp. 61-63

http://www.krishisanskriti.org/acsit.html

Security Analysis of Web Application using Genetic Algorithms in Test Augmentation Technique

Keertika Singh[1] and Garima Singh[2]

[1]M.Tech (Software Engineering) Babu Banarasi Das University, Lucknow, INDIA
[2]Assistant Professor, BBDU Babu Banarasi Das University, Lucknow, INDIA
E-mail: [1]keertikasingh99@gmail.com, [2]garima_singh2108@yahoo.com

Abstract—*A security of web is a branch of Information Security that deal with the security of web application , website, web services. Security of the web deal with testing the security of the confidential data and make the data remain confidential. This Paper proposed an approach to test the web application by using a concept of genetic algorithms. The approach of security testing is based on understanding of how the client (browser) and the server communicate using HTTP. A researchable test suite was generated to cover as many fault sensitive transaction relation as possible with the genetic algorithms. SQL injection attack are very critical as attacker can get a vital information from the server database. The proposed methodology is to create a researchable test suite based on the collected user session with genetic heuristic. The main aim of this paper is to explain the application of genetic algorithms to generate a test cases of the web application on bases of user session.*

1. INTRODUCTION

Testing is the process of exercising software with the intent of finding errors. This fundamental philosophy does not change for WebApps. In fact, because Web-based systems and applications reside on a network and interoperate with many different operating systems, browsers, hardware platforms, and communications protocols, the search for errors represents a significant challenge for Web engineers[1]. Web Application is a application that is used over a network. Testing of web application is one of the time consuming task so many of the developer neglect the testing activity. Web application testing is a very expensive process in terms of time and resources due to the nature of web application. Testing, designing and generating test cases are challenging tasks because web application is complex and changeable.

The web application is considered as one of the distributed system, with a client– server or multi-tier architecture, including the following characteristics:-

-*Wide number of users* are distributed all over in the world access concurrently

-Web Application always run in *heterogeneous execution environment i.e.* different hardware, network connections, operating systems, Web servers, and Web browsers.

-It is able to generate *software components at run time* according to user inputs and server status.

1.1) The following WebApp characteristics drive the process:

Immediacy. Web-based applications have an immediacy that is not found in any other type of software. That is, the time to market for a complete Web site can be a matter of a few days Developers must use methods for planning, analysis, design, implementation, and testing that have been adapted to the compressed time schedules required for WebApp development.

Security. Because WebApps are available via network access, in order to protect sensitive content and provide secure modes of data transmission, strong security measures must be implemented throughout the infrastructure.

Aesthetics. An undeniable part of the appeal of a WebApp is its look and feel. When an application has been designed or sell products, aesthetics may have as much to do with success.

2. GENETIC ALGORITHM

Genetic algorithms is a heuristic based search practice and we use this because of their ability to generate near global optimum solution and their extensive use in the prose of heuristics. The use of genetic algorithms is also facilitated by the fact that many of the testing problems can be formulated as search problems. For occurrence, in our case, the problem of generating adequate test data can be formulated as the problem of searching the input domain of the program, for those input values that satisfy the adequacy test criteria or that can identify faults in the program.

There are many real life problems where genetic algorithm is applied.[2]

The population of chromosomes is a possible solution of the problem. This is the starting of genetic algorithm. A

chromosome is a string of binary digits and it is the set of values of input variable and are obtained from the input domain each digit that can form a chromosome is called a gene [3]. This initial population can be totally random or can be created manually using processes such as greedy algorithm. The pseudo code of a basic algorithm for GA is as follows;[4]

Initialize (population)
Evaluate (population)
While (stopping condition not satisfied)
{
Selection (population)
Crossover (population)
Mutate (population)
Evaluate (population)}

2.1) A GA uses three operators on its population

Selection
A selection scheme is applied to determine how individuals are chosen for mating based on their fitness. Fitness can be defined as a capability of an individual to survive and reproduce in an environment. Each chromosome is evaluated.

Crossover or Recombination: After selection, the crossover operation is applied to the selected chromosomes. It involves swapping of genes or sequence of bits in the string between two individuals. This process is repeated with different parent individuals until the next generation formed.

Mutation: Mutation alters chromosomes in small ways to introduce new good traits. It is applied to bring diversity in the population.

Table 1

CLASSICAL ALGORITHMS	GENETIC ALGRITHMS
Generate a single point at each iteration. The sequence of point approach an optimal solution.	Generate a population of a point at each iteration. The best point in the population approaches as a optimal solution.
Select a next point in the sequence by a deterministic computation.	Select the next population by computation which use random number generation

2.2) Estimation of Global minima for Stochastic Problem

GA can solve both constructed and non constructed optimization problem which is totally based on a natural process of selection. The GA differs from a classical, derivative- based, optimization algorithms.[4]

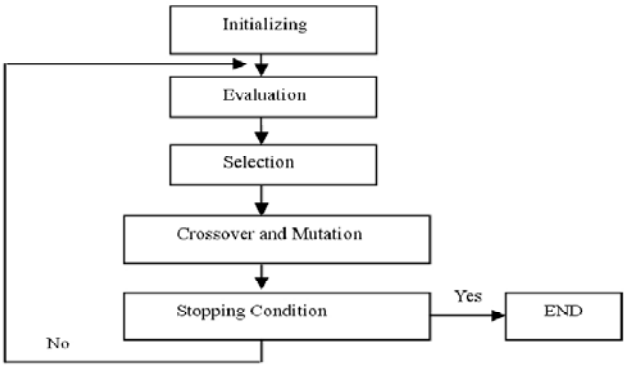

Fig. 1: Block Diagram of Genetic Algorithms

3) PROPOSED WORK; In a proposed methodology, a set of requests sent from clients are recorded as log of server. Three portions of the request is considered for the user-session-based testing[6]. The first portion is user IP and timestamp, which is used to identify a user session. In general, a user session is said to have began when a new IP address sends a request to the server and ends when the user leaves the web site, or the session is timed out. Another portion is composed of request pattern (GET/POST) and URL, which is called base request, while the third portion is the parameter-value pairs carried by base request. An identified user session can be simplified as a sequence of base requests and parameter-values to describe users' sequential actions for web resources. On each Web server, request by the user is considered as a record. The record generally include request source (user IP address), request time, request mode (such as GET, POST), the URL of requested information, data transport protocol (i.e. HTTP), status code, the number of bytes transferred and the type of client, etc. It scan the log however, it is difficult to organize these original records directly. We initial eradicate inappropriate data that include records whose status codes are erroneous, embedded resources such as script files and multimedia files having extension names are .gif, .jpeg or .css, etc., to obtain the set of user sessions for analysis. Then, we create user sessions through scanning the logs on Web servers. Once a new IP address occurs, a new user session is created.[7]

Table 2: Comparision Study of Function Used in Test Case Generation by Genetic Algorithms[9]

FITNESS FUNCTION	FILTERING FUNCTION	CROSSOVER FUNCTION	MUTATION FUNCTION	ACCEPTANCE FUNCTION	CONTROL FUNCTION
The fitness function is used to calculating the fitness value of chromosome.	In this sort chromosomes according to fitness from high to low, and select chromosomes in order.	Two chromosomes with the highest fitness and lowest fitness are selected from the chromosome Group and compared	A mutation probability is predefined to control whether or not a mutation operation will be carried out for a chromosome.	Acceptation function is used to compare the fitness values of 4 chromosomes, the offsprings and their parents.	Control function is used to control the coordination of the 5 modules. These 5 modules work logically and iteratively unit.

the fitness function, of "test case generation using GA", is **Fitness value = (_ * |*CDTR*| + |*CLTR*|) / (_ * |*DTR*| + |*LTR*|)**	Chromosome whose fitness is lower than a predefined percentage of the parents' average fitness should not be selected.	The *next pointer of the crossover points (if they are present) will be exchanged with each other.	Chains are selected from chromosomes of the initial population, until the common ratio between the selected chain and the mutating chromosome is smaller than a defined common ratio threshold.		The three modules of crossover, mutation and acceptation are in a nested loop to deal with chromosome.

3.1) ALGORITHMS FOR IDENTIFYING AFFECTED ELEMENT

Algorithm: ReduceUSession
input:
The set of user sessions Λ = {s1, …, sk}, where k is the number of user sessions;
The URL trace U1, …, Uk, which are requested by s1, …, sk respectively;
output:
The reduced set of user sessions denoted by Γ;
begin
Γ = Φ;
while (another user session that is not marked in Λ exists)
tag1 = FALSE;
tag2 = FALSE;
Select a user session si that is not marked in Λ, and then mark it with "USED";
for (the URL trace U_j requested by each user session s_j in Γ)
if isPrefix(Uj, Ui) //the URLs requested by si is //more, so s_j is *redundant*
Γ = Γ-{s_j};
tag1 = TRUE;
endif;
if isPrefix(Ui, Uj) //here, Γ keeps unchanged, //and s_i is *redundant*
tag2 = TRUE;
break; //exit *for* cycle
endif;
endfor;

3.2) Testing Web Applications Using Genetic Algorithm

When the prioritization and grouping of the user session is done, we get several initial test suites and test cases. However, the test scheme generated by the elementary prioritization is not fast in finding faults and can not satisfy the requirements earlier. Therefore, genetic algorithm is used further to optimize the grouping and prioritization. Selection, crossover and mutation are the 3 basic operator of GA. Mainly six modules are used in this it include fitness function, filtering function, crossover function, mutation function, acceptation function and control function.[8]

4) Implementation of genetic Algorithms: The user connects to our network using a standard, Java-enabled browser, such as Netscape Navigator or Microsoft Internet Explorer. The browser loads a web page in which a reference to the GAWebTutor applet has been embedded. The applet code migrates across the internet from the web server to the client browser where it is interpreted by the Java Virtual Machine.

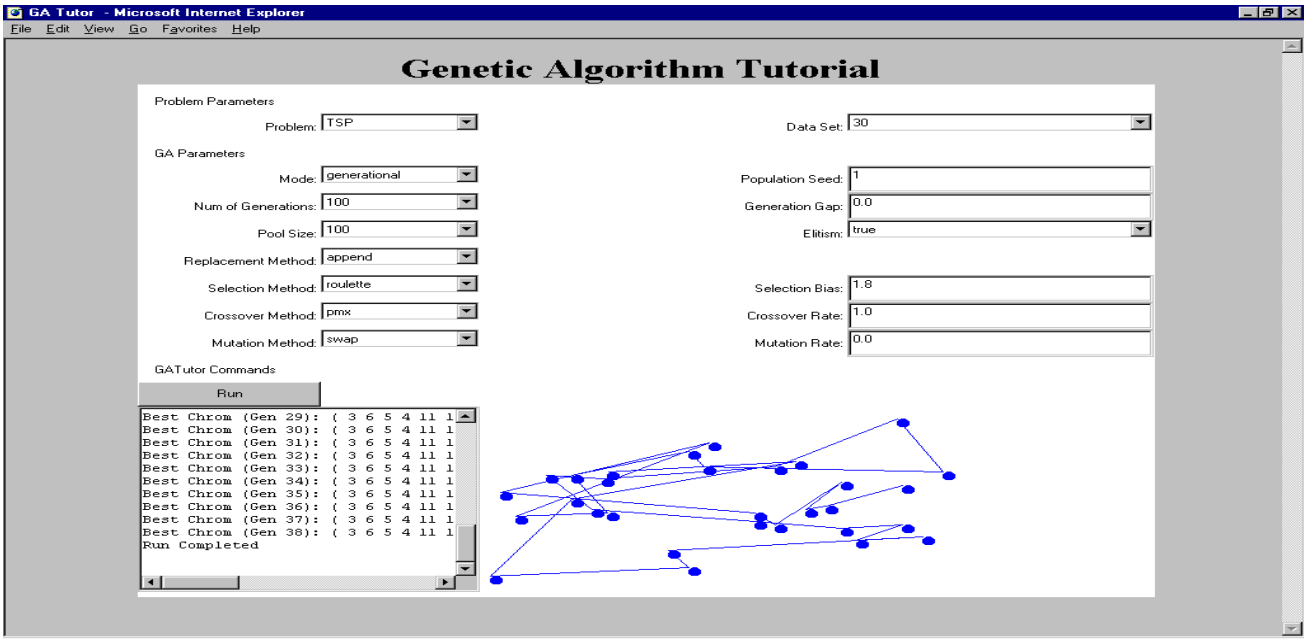

Fig. 2: GAWebTutor Interface

The client user enters parameter information through standard GUI components provided in the applet, such as drop-down lists, text boxes, etc. Once the user clicks on the "run" button the client waits for results. Once received, the client displays the contents of the best chromosome and its fitness rating for each generation of the GA. Additionally, a graphical illustration of the tour represented by the best solution from each population is displayed for the user. The chromosomes and tour graphics are displayed serially to reinforce the idea of an evolving solution set. The final tour of the last generation changes color to indicate to the user that all results have been displayed.[10,11]

3. CONCLUSION AND FUTURE WORK

The approach of this paper is to capture the series of logs of the server by the set of request send from the client, the three portion of the request is to be considered as user session and testing is being applied over it. We initially check the status code, when the record status code is erroneous, embedded resources then it is used then it is used to set the user session for analysis. Reduced user session algorithms and user session prioritization is applied to generate the initial test suit and test case.

In future research many question need to be answered and many factors are not yet considered like running cost of each test cases including loading time, time to save test state. The investigation should also be done on augmentation of the generated test suit to meet a full coverage from a structural analysis.

REFERENCES

[1] Roger S. Pressman, Ph.D. Senior Consulting Editor C. L. Liu, *National Tsing HuUniversity* Consulting Editor Allen B. Tucker, Bowdoin College Fundamentals of Computing and Programming Computer Organization and Architecture Systems and Languages Theoretical Foundations Software Engineering and Databases.

[2] S. Khor, P. Grogono, "Using a Genetic Algorithm and Formal Concept Analysis to Generate Branch Coverage Test Data Automatically", Proceedings of the 19th International Conference on Automated Software Engineering (ASE"04), 1068-3062/04 © IEEE.

[3] M. R Girgis, "Automatic Test Data Generation For Data Flow Testing Using A Genetic Algorithm", Journal of Universal Computer Science, Vol.11, No.6, pp.898-915, June 2005

[4] Sangeeta sabharwal, ritu sibal, chayanika Sharma "Prioritization of test case scenarios derived from activity diagram using genetic algorithm". ICCCT, pp.481-485 IEEE (2010).

[5] www.mathswork.com

[6] E. Hieatt and R. Mee, "Going Faster: Testing the Web Application," *IEEE Software*, Vol. 19, No. 2, 2002, pp. 60-65.

[7] D. C. Kung, C. H. Liu and P. Hsia, "An Object-Oriented Web Test Model for Testing Web Applications," *Proceedings of the* 1*st Asia-Pacific Conference on Web Applications*, New York, 2000, pp. 111-120.

[8] J. H. Holland, "Adaptation in Natural and Artificial System," University of Michigan Press, Michigan, 1975.

[9] User Session-Based Test Case Generation and Optimization Using Genetic Algorithm* Zhongsheng Qian ***J. Software Engineering & Applications*, 2010, 3, 541-547**

[10] L.R. Knight and R.L. Wainwright, "HYPERGEN: A Distributed Genetic Algorithm on a Hypercube," *Proceedings of the 1992 Scaleable High Performance Computing Conference*, SHPCC '92, Williamsburg, VA., April 26-29, 1992.

[11] C. Prince, R.L. Wainwright, D.A. Schoenefeld, and Travis Tull, "GATutor: A Graphical Tutorial System for Genetic Algorithms," SIGCSE Bulletin Vol. 26, No. 1, March 1994, pp. 203-207.