Application of Genetic Algorithm in Heterogeneous Redundant Web System

Tang Jingfan, Hu Enchao, Zhang Min, Liang Huibing, Jiang Ming
Hangzhou Dianzi University, Hangzhou310018, China

Abstract
Web heterogeneous system is a kind of intrusion tolerance system model, which blocks hackers sniffing and attack through the internal heterogeneous execution set. Thus, it’s very important to assess the heterogeneity of execution set, and remove the execution bodies with large similarity in the set before the system is absolutely adopted. In this paper, the modules with the same function on each executive body are divided into the same layer, and the similarity degree of each layer is calculated by the source code similarity measure method or the subjective score. Finally, the screening of the execution set is achieved through the genetic algorithm to maximize the heterogeneity of the execution set. The results of simulation show that the execution set with the greater degree of heterogeneity has better security degree and intrusion tolerance degree, which is meaning to design a distributed adaptive intrusion tolerant model that can provide continuous and reliable service with stable response time.

Key words: Intrusion Tolerance; Web Heterogeneous System; Similarity Measure; Genetic Algorithm

1. INTRODUCTION

With the developing of intrusion methods, firewall and intrusion detection technology cannot protect the system’s safety enough. Once the system's vulnerability is exposed or attacked, it will enter a fragile or unpredictable state, and will not be guaranteed to provide normal service. Intrusion tolerance system is mainly considered in the presence of the invasion of the system in the case of survivability, and to ensure that after the invasion, the system can external maintenance normal operation and provide the core or basic services.

In recent years, Web heterogeneous systems have been paid more and more attention by researchers. Web server is a multi-layer structure, which include application software layer, server software layer, database layer, operating system layer and layers of the sub-layers from top to bottom. This structure also provides multiple attack targets for attackers, which makes it more difficult for the web server to be defended. Network attacks usually rely on the specific properties of the specific system. Different technology stack design or implementation often makes the systems with similar function have completely different characteristics, such as different web services software, Apache, Nginx and Lighttpd have different characteristics in the stability, security, static file processing, reverse proxy, etc. For different operating systems, they will also have different strengths and weaknesses. For example, the loopholes CVE-2014-6324 of system privileges for enhancement only exists in Windows systems, while not exist in Linux systems at all. The differences among the different technical stack also provide the possibility for the security defense(YL Sun and W Yu, 2006; Lamsal P, 2001). It uses a variety of database storage technology in(Garg, Binny and K. Kaur, 2015), uses a rich virtual computing environment to achieve the intrusion tolerance in(T huang, Y Zhu, Y Wu, G Dobbie, 2015), and uses heterogeneous server software, file systems and operating systems in the execution set in(TONG Qing, ZHANG Zheng, ZHANG Wei-Hua, WU Jiang-Xing,2017). Although the design and implementation methods of heterogeneous systems are diverse, and the techniques and means used are different, the goal of heterogeneous systems is to protect the confidentiality of system data from the perspective of intrusion prevention. The heterogeneous execution bodies in the system are the key to shield the attacker from sniffing. The application of the heterogeneous execution set has been recognized by the researchers. However, the discussion of the heterogeneity of the execution set has mostly been conceptual.

The main contributions of this paper to the optimization of web heterogeneous systems are as follows:
(1) a way to optimize Web heterogeneous systems is proposed. The selection, improvement and design of the execution set are carried out according to the analysis and evaluation results on the similarity of each layer.
(2) The heterogeneity optimization of the web heterogeneous system belongs to the combinatorial optimization problem, and its time complexity is $O(n^2)$. This paper introduces the genetic algorithm, which avoids the iterative operation into the local minimal trap.

The rest of paper is organized as follows. Section 2 introduces the Web heterogeneous system model and the importance of heterogeneity for web heterogeneous systems. Section 3 gives the definition of similarity, and proposes an algorithmic model for optimizing the Web heterogeneous system. Section 4 shows the analysis of simulation results. Section 5 gives the conclusion.
2. WEB HETEROGENEOUS SYSTEM MODEL

2.1. Web Heterogeneous System Architecture

Web heterogeneous system architecture is shown in Figure 1. System functions can be summarized as "input-processing-output", as well as the structured design of IPO (Input-Process-Output). The Web heterogeneous system structure is processed using the heterogeneous execution set, copying the same input into \( n \) copies through the input agent, and distributing it to \( n \) isomorphic bodies to process the processing results to voter for voting, so as to get the only relatively correct output.

![Figure 1. Heterogeneous redundancy structure](image)

2.2. The Importance of Heterogeneity to Web Heterogeneous Systems

In order to facilitate the analysis of the importance of heterogeneity, it has the following restrictions:

1) The execution set only implements the isomerization of the database management system (DBMS), the operating system (OS) and the server software (Webserver) by ignoring the heterogeneous structure of other layers.

2) The loopholes between different layers will not affect each other.

3) On the same layer, different technology stacks produce completely different vulnerability sets, which means the intersection of loopholes is empty.

4) It assumes that other aspects of the system are absolutely safe.

The traditional web application's vulnerability set is defined as follows:

\[
L(S) = L(OS) \cup L(DBMS) \cup L(Webserver)
\]  

(1)

The vulnerability set for Web heterogeneous systems is defined as follows:

\[
L(S') = \bigcap_{k=1}^{3} L(OS_k) + \bigcap_{i=1}^{3} L(DBMS_i) + \bigcap_{j=1}^{3} L(Webserver_j)
\]  

(2)

where \( L(Webserver_j) \in L(Webserver), L(OS_k) \in L(OS), L(DBMS_i) \in L(DBMS) \) and \( S' \) represents a traditional web application, \( S' \) represents the Web heterogeneous system, and \( L \) indicates that the input and output do not meet the expected results.

The value of \( |L(S')| \) is related to \( \bigcap L(\Theta) \). Therefore, increasing the heterogeneous degree of database management system (DBMS), operating system (OS) or server software (Webserver) can reduce the value of \( |L(S')| \) and improve the security of the system.

3. OPTIMIZATION PROGRAM ON WEB HETEROGENEOUS SYSTEM

Because developers are influenced by educational background and expertise, they can generate similar
code vulnerabilities even if they develop projects independently. At the same time, cloning code in the history of the development of the software is also common, i.e. Linux has 16% to 25% of the cloning code, GNU has about 9% of the cloning code, JDK has about 21% to 29% of the cloning code. All of the above are potential threats to redundant system failures. Therefore, when discussing the heterogeneity of web heterogeneous systems, code similarity is an important reference indicator.

3.1. Similarity Evaluation Scheme on Source Code

1) when the source code in the technology stack can be obtained, the formula for the similarity of any two source codes is defined as follows:

\[
Sim^1(p, q) = \min(Moss(p, q))
\]  

where Moss(Schleimer, Saul, D. S. Wilkerson and A. Aiken, 2003), is an open-source code check system by the United States Stanford University, which returns the value pair (X,Y) to indicate that the X% code in p matches the Y% code in q. This paper takes smaller values as source code similarity.

2) when the source code in the technology stack cannot be obtained, the similarity of the subjective evaluation method is used. For example, if a development team's professional ability is general, and they also have strong tolerance on the system loopholes, the heterogeneous executive body constructed by them is more likely to be a high-risk product. And vice versa is the possibility of higher security products. This paper defines the method of subjective evaluation similarity by referring to the above two aspects (tolerance and technical) as follows:

\[
Sim^2(a, b) = e^{\frac{\max(a[\text{tolerance}], b[\text{tolerance}])}{\max(a[\text{technical}], b[\text{technical}])}}
\]

Where tolerance and technical are the attribute values of a,b and their range is (0,100).

3.2. Similarity of the Same Layer

The Web heterogeneous system model can be simplified as follows:

\[
\alpha_1 = \begin{bmatrix} c_{11} \\ c_{12} \\ \vdots \\ c_{1n} \end{bmatrix}, \quad \alpha_2 = \begin{bmatrix} c_{21} \\ c_{22} \\ \vdots \\ c_{2n} \end{bmatrix}, \quad \ldots, \quad \alpha_m = \begin{bmatrix} c_{m1} \\ c_{m2} \\ \vdots \\ c_{mn} \end{bmatrix}
\]

\[
S\alpha_k = \begin{bmatrix} S_{k1} \\ S_{k2} \\ \vdots \\ S_{km} \end{bmatrix}
\]

Where \(\alpha\) denotes the execution body, m denotes the number of execution bodies, and n denotes the number of layers of different technology stacks.

The similarity of the \(K^{th}\) layer is defined as follows:

\[
SR_k = \max(S_{k1}, S_{k2}, \ldots, S_{km})
\]

\(S_k\) is a one-dimensional vector, the value of the i-th element on vector \(S_k\) is compared with the rest of the m-1 executions on the k-th layer, and accumulate the value obtained after Sim operation. The function of Sim is defined as follows:

\[
Sim(a, b) = \begin{cases} 
1, & Sim^1(a, b) \geq \text{threshold or } Sim^2(a, b) \geq \text{threshold} \\
0, & \text{otherwise}
\end{cases}
\]
where \( \text{threshold}_k = 1 - w_k \), \( w_k \) represents the ability of the \( k \)-th layer technology stack to influence system security. Each technology stack can be considered as a property of the system, and the importance of the attribute is an important concept in the rough set (M Chen and X Xia, 2010). In the rough set, the importance of each attribute in the same field can be judged by the approximate classification quality, that is, the importance of the attribute is the classification ability of the attribute. In order to measure the importance of conditional attributes, this paper makes a reasonable evaluation of the impact of hierarchical vulnerability attacks by CVSS v2 (Gallon and Laurent, 2010) security test.

\[ w_k = \frac{BS_k}{\sum_k BS_k} \]  
\[ BS = \text{round}_1 \cdot \text{decimal}((0.6*IMP)+(0.4*EXP)-1.5)*f(IMP)) \]  
\[ EXP = 20*(AV)*(AC)*(AU) \]  
\[ IMP = -10.41*(1-(1-C)*(1-I)*(1-A)) \]  
\[ f(IMP) = \begin{cases} 0, & \text{IMP} = 0 \\ 1.176, & \text{otherwise} \end{cases} \]

Where \( AV, AC, AU, C, I \) and \( A \) are defined as follows:

- **Access Vector (AV)** is dependent on the amount of access an attacker needs to exploit a vulnerability. Thus, an attack that needs physical access to a system will have lower score than one that can be exploited over the Internet by any machine.

- **Access Complexity (AC)** represents the complexity of exploiting an attack. A buffer overflow attack on an Internet service is less complex than an e-mail client vulnerability in which a user has perform attachment downloads followed by executing it and hence has lower AC value.

- **Authentication (Au)** level required to execute the attack. For example, if no sign-up account is required to exploit the system, this value is high. In contrast, if one needs multiple accounts to exploit the vulnerability, the value is low.

- **Confidentiality Impact (C)** scores are low if only some (nonrelevant) information gets leaked. Highest impact occurs when say, the entire database is compromised if the vulnerability is successfully exploited.

- **Integrity Impact (I)** refers to the attacker’s power to modify files or behaviour of a system if he executes the exploit successfully. The more the power - say the attacker is able to change code or remove arbitrary files in the system – the higher this value.

- **Availability Impact (A)** represents the power of a successful exploit to bring down the availability of a system. A successful Denial of Service (DoS) that brings down an application server, will have high impact. Based on the above 6 independent values, we can calculate the values of impact (IMP), Exploitability (EXP) and basescore (BS) by the above formula.

3.3. Fitness Function of the Genetic Algorithm

For security problems, a priori knowledge is worth considering. Although the system cannot rely solely on it to defend against unknown vulnerabilities, it can objectively reflect the general distribution of vulnerabilities. In combination with the similarity computation of Web heterogeneous systems proposed in Section 3.2, a data table similar to the following can be obtained:

<table>
<thead>
<tr>
<th>Name</th>
<th>P</th>
<th>SR</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServerScript</td>
<td>72.2%</td>
<td>75.1%</td>
</tr>
<tr>
<td>DBMS</td>
<td>11.7%</td>
<td>5.0%</td>
</tr>
<tr>
<td>OS</td>
<td>9.8%</td>
<td>12.8%</td>
</tr>
<tr>
<td>Server software</td>
<td>3.3%</td>
<td>42.6%</td>
</tr>
</tbody>
</table>

Where \( P \) represents the proportion of the number of attack times on the Name and \( SR \) represents the similarity of Name layer.

The fitness function of the genetic algorithm (P Larrañaga and C Bielza, 2014) used in this paper is as follows:
\[ w(k) = P \left( C_1 \cap C_2 \ldots C_k \right) = \sum_{i=1}^{n} p_i \cdot SR_i \quad s.t. \quad \sum_{i=1}^{n} p_i = 1 \quad (12) \]

Where \( k \) represents the number of executions and \( n \) represents the number of layers. Obviously, the smaller value of \( w(k) \) means that the system is more secure.

### 3.4. Genetic Algorithm

The genetic algorithm is used in this paper to calculate the security of the different combinations according to \( w(k) \) and removes the unreliable execution body to form a new execution set. The specific operation of the algorithm is similar to (ZH Zhou, J Wu and W Tang, 2002), as follows:

1. **Chromosome coding:** \( n \) candidate execution bodies are encoded as binary strings. When the \( i \)-th is 1, it is selected. Otherwise, it is abandoned.

2. **Parameters and operations of genetic algorithms:**
   - The population size of the genetic algorithm is set by 50. The selection operator adopts tournament selection, and the best individual is retained in each generation, and the 10% generations are copied directly to the next generation. The probability of crossover is 2%, and the crossover and mutation should be adjusted according to the number of "1" in the individual as odd numbers and no more than the maximum redundant number. The evolutionary algebra is set at 100. The fitness function is \( w(k) \).

3. **In an actual Web heterogeneous system, too many redundant executions require a higher cost. Thus, in the evolutionary process, the algorithm limits the number of "1" in the chromosome to a maximum of seven. The specific operation is that when the population is initialized, the number of "1" in the chromosome is limited to 3, 5 or 7. If the number of "1" in the chromosome is not 3, 5 or 7, the number of "1" can be adjusted by random mutation of the gene in the chromosome. If the "1" number is less than 3, the number of genes "0" in the chromosome is randomly selected until the number of genes "1" reaches 3; if the number of "1" is greater than 7, the gene "1" in the chromosome is randomly selected until the number of gene "1" reaches 7; if the number of "1" is 4 or 6, it will be determined to be 3, 5, or 7 by random method, and then the chromosome is adjusted by the above-mentioned variant chromosome method.

From the above genetic operations, it is necessary to run the genetic algorithm at about 5000 times, irrespective of the size of the \( n \). If the traversal method is used, all combinations of redundant numbers of 3, 5, or 7 need to be calculated. When \( n = 20 \), the number of evaluations is about 95000 times, and as the \( n \) increases, the number of evaluations required by the traversal method increases exponentially and is difficult to compute. Therefore, this paper uses genetic algorithm to find out the optimal or sub optimal heterogeneous combinations quickly.

The pseudo code is as follows:

\[
\begin{align*}
g &\gets \phi \\
\text{while} \quad g \cdot g < 0 \\
\text{do} \quad u &\gets \text{EXTRACT-LOWEST-SM}(\text{POP}, \text{10\%}) \\
\text{NPOP} &\gets \text{NPOP} \cup \{u\} \\
\text{while} \quad \text{LENGTH}(\text{NPOP}) < m \\
\text{do} \quad w &\gets \text{EXTRACT-LOWEST-SM}(\text{POP}, \text{2}) \\
\text{ONE-POINT}(w) \\
\text{if} \quad \text{MUTATION} \quad \text{do} \quad \text{MUTATE}(w) \\
\text{if} \quad \text{ADJUSTMENT} \quad \text{do} \quad \text{ADJUST}(w) \\
\text{NPOP} &\gets \text{NPOP} \cup \{w\} \\
\end{align*}
\]

### 4. EXPERIMENTAL SIMULATION

#### 4.1. Experimental Configuration

The configuration used in the experiment is shown in Table 2. By using the table configuration, 20 executable bodies are constructed. And it will use these 20 executive bodies to combine 50 Web heterogeneous systems. In addition, all tests were conducted in the same environment.
Table 2. The configuration of the devices

<table>
<thead>
<tr>
<th>host OS</th>
<th>Virtual OS</th>
<th>Server software</th>
<th>Server script</th>
<th>Other security tools</th>
</tr>
</thead>
</table>

In the experiment, the Windows virtual machine retained two loopholes of ms08_067 and ms12_020 to simulate unknown vulnerabilities. The experiment objects are:
1) 50 Web heterogeneous systems (Random Combination)
2) Optimized Web heterogeneous systems (Genetic_Algorithm)
3) Traditional web system called Traditional, which uses the windows virtual machine (Traditional)
4) Traditional traversal way to build the Web heterogeneous system (Traversal)

4.2. Experimental Results and Analysis

Experiment 1: Test objects are Random_Comination and Genetic_Algorithm, test items are ms08_067 and ms12_020, the test results are as follows:

Table 3. The testing results of security

<table>
<thead>
<tr>
<th>Test Items</th>
<th>Name</th>
<th>Random Combination(50)</th>
<th>Genetic-Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>ms08_067</td>
<td></td>
<td>7 failed</td>
<td>pass</td>
</tr>
<tr>
<td>ms12_020</td>
<td></td>
<td>7 failed</td>
<td>pass</td>
</tr>
</tbody>
</table>

Experiment 2: Test objects are Traditional, Genetic_Algorithm and Traversal. In addition to ms08_067 and ms12_020, the test items also includes database security, front-end script, the main loopholes. The results are as follows:

Table 4. The testing results of security

<table>
<thead>
<tr>
<th>Test Items</th>
<th>Name</th>
<th>Traditional</th>
<th>Traversal</th>
<th>Genetic-Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Script</td>
<td></td>
<td>pass</td>
<td>pass</td>
<td>pass</td>
</tr>
<tr>
<td>Database Hacker</td>
<td></td>
<td>fail</td>
<td>pass</td>
<td>pass</td>
</tr>
<tr>
<td>MainStream Hacker</td>
<td></td>
<td>fail</td>
<td>pass</td>
<td>pass</td>
</tr>
<tr>
<td>ms08_067,</td>
<td></td>
<td>fail</td>
<td>pass</td>
<td>pass</td>
</tr>
<tr>
<td>ms12_020</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Experiment 3: Increase the usage of Windows virtual machine, build Traversal and Genetic-Algorithm under the corresponding conditions, and use ms08_067 and ms12_020 to test and record the consumption of both CUP, the results are as follows:

Figure 3. Adjust the configuration to get the corresponding experimental results
From the above three groups of tests, it shows that Web heterogeneous systems constructed by traversal or genetic algorithm with SR as the reference index has more advantages in security, and Web heterogeneous system of traditional Web system or random combination does not have this advantage. Considering the size of the project in the realistic scene, the genetic algorithm used to build Web heterogeneous system is more general than the traversal, and the response is quicker.

5. CONCLUSION

This paper provides a method for quantifying and analyzing the heterogeneous degree of the system, and optimizes the system by using genetic algorithm. Simulation results show that the greater the heterogeneity of the system, the better its data confidentiality and intrusion tolerance. Therefore, this method has important guiding significance on how to select and improve the execution set. The method still has shortcomings:

1) the factors influencing the system heterogeneity are simplified, and the value of similarity may not be accurate enough.

2) it still requires manual labor to deal with the technology stacks that cannot compare source code.

ACKNOWLEDGMENTS

This work is supported by national key research and development program (No. 2016YFB0800201) and Zhejiang Provincial Technical Plan Project (No. 2017C01064, 2017C01055, 2018C01088).

REFERENCES


