Research on the Evaluation of the Effectiveness of the Algorithm Based on Shellcode Code Analysis and Exploit Module Attack

Xin Yue\textsuperscript{1,a}, Xiuliang Mo\textsuperscript{2,b}, Chundong Wang\textsuperscript{3,c}

\textsuperscript{1}Tianjin University of Technology, China
\textsuperscript{2}Tianjin University of Technology, China
\textsuperscript{3}Tianjin University of Technology, China

\textsuperscript{a}1521481125@qq.com, \textsuperscript{b}moxiuliang@163.com, \textsuperscript{c}michale3769@163.com

\textbf{Keywords:} Attack tree module; Shellcode; Exploit; Algorithm effectiveness evaluation

\textbf{Abstract.} The paper first introduces the basic principle of shellcode analysis and exploit attack module, what’s more, the paper makes a research on the independence and dependence between shellcode code and exploit attack module. Based on the unique relationship between the two modules, the paper explores the implementation of the two modules, and then proposes a new method based on attack tree theory. Experiments show that attack model performs better and effective. According to different exploits, the proposed transformation of the attack tree evaluation algorithm can give out corresponding threat degree evaluation, and eliminate the generated interference during the middle of executive body attack, so the detection is more accurate.

1 Introduction

In today's era, vulnerability attacks has been the focus of information security issues as the focus of many researchers to explore [1]. Launched a vulnerability to attack, to attack all the presence of such vulnerabilities in the network host, not only in the background to monitor the host running, but also even the vulnerability to attack, get system permissions, wanton destruction of all the data in the host [2-4]. To this end, the vulnerability research has been the goal of security personnel, exploiting vulnerability attacks in the field of information security is an in-depth study of the object.

In recent years, experts in related fields mainly pay attention to the polymorphism of shellcode, on the other side, its static detection and dynamic analysis of detection methods. Executed the innovation of this paper is to exploit module carrying shellcode from the angle of research, and put forward an attack effectiveness evaluation algorithm for the attacks. The overall idea is to go for in-depth study of shellcode in exploit in the way of delivery and the implementation of different ways, and the implementation of different effects of research, the implementation of the attack effect, a method based on attack tree model of the algorithm to evaluate the effectiveness of the algorithm.

2 Analysis of shellcode and exploit

2.1 Basic concepts of shellcode and exploit modules

Shellcode called the attack buffer overflow into process code. This code can be an expurgated version of important documents for attack purposes, upload Trojans, malware running or attack the target host data, and even to format the hard disk and so on. Among them, the code need to use assembly language to write, and converted into binary code, its length and content are often subject to a lot of restrictions, so that development and debugging is very difficult.

Exploit module is the process of code implantation, which is the process of shellcode implementation. It usually emerges in the form of especially code, used to generate the attack of the
network data packets, or other forms of aggression. Its core purpose is to flood back to the address, attach the control of the process of hijacking, and then jump to implement the shellcode.

2.2 The independence and dependence of exploit module and shellcode code

Shellcode is the overflow buffer into the process of the code, and the Exploit module is the execution of the implantation process, the two parts are independent of each other. Exploit module which is the delivery mode of shellcode, it is mainly responsible for the successful invasion to the user host, and triggers the implementation of shellcode. If shellcode has been a successful invasion, the other side of the host would cause damage by controlling the implementation of the operation.

Between the two parts, the exploit module without shellcode cannot be an effective attack, even cannot cause attack damage, and separate exploit from the shellcode module cannot be successfully implemented, both of them, it is also a dependency relationship. As shown in figure 1:

![Figure 1. Exploit module launch shellcode attack.](image)

The exploit module carries the shellcode which has the effective attack effect as the attack missile to carry on the system to aim at or navigate, can accurately be the effective shellcode through the transportation line, arrives at the attack host, implements the true attack.

Missile warhead can be replaced, just like the front end of the shellcode can be replaced, and the missile route is decided by the corresponding restrictions on the conditions and the implementation, the same as the route to the missile.

2.3 The basic model building

$E$ for the exploit module, $S$ on behalf of the shellcode code, $G$ is said to be the implementation of the effect of the object. The Exploit module carries the shellcode code execution to appear the effect object can be expressed as $G_n = E_i(S_i)$, for example, when carries out the different exploit module and the shellcode code to be able to form the different implementation effect object. 

$G_{i1} = E_i(S_{i1})$, $G_{i2} = E_i(S_{i2}), G_{i3} = E_i(S_{i3})$; And so on, we will all have the effect of the implementation of the 

$$
G_{ni} = \begin{bmatrix}
G_{i1} & G_{i2} & G_{i3} & \ldots \\
G_{i1} & G_{i2} & G_{i3} & \ldots \\
G_{i1} & G_{i2} & G_{i3} & \ldots \\
\ldots & \ldots & \ldots & \ldots 
\end{bmatrix}
$$

The different behavior effects of different shellcode and exploit attacks are executed on behalf of the representative, the different levels of the damage effect, the establishment of the attack tree model in the following.
3 Attack effectiveness evaluation model establishment

3.1 Attack tree model

For the attack of software vulnerabilities, the evaluation system of standard attack effectiveness is not formed at the present stage. Based on the degree of the attack effect, this paper proposes a model evaluation method based on attack tree theory. The concept of attack tree was first proposed by Bruce Schneter[5-7] in 1999. It was originally designed to describe the security system in order to evaluate the security of the system accurately. However, there are some defects in the actual attack tree model. The main performance is that the weight of the node cannot be accurately reflected in the attack process. Therefore, To this end, Hui Wang and other persons[8] proposed a novel concept of the extended attack tree, the introduction of the weight of the attack, attack costs and other ideas, making the relationship between the tree structure and the main body more similar to each other.

3.2 Evaluation model of attack tree algorithm

(1) Universal formal definition:

1. Tree root node represents the malicious code attack process of the initial implementation of the object is usually the original module of malicious code, tree internal node on behalf of the malicious code execution generated in the process of each implementation of a temporary object, tree leaf node represents the final implementation.

2. Each edge in the tree indicates the relationship among the nodes, usually by an object that is executed or called by another object.

3. Each node of the tree has a malicious weight, which is used to show the malicious damage to the system.

(2) Transformation attack tree definition:

1. Transformation attack tree Tree= (V, E, S, I, W) is a relation tree with one or more nodes.

2. V(T) is a non-empty set of nodes, nodes that some of the execution of the attack behavior of the object, the root node Root said with the download or copy function or the initial implementation of the implementation of the object.

3. E(T) is subset of the V(T), if exist an edge E(v1, v2) between v1 and v2, that is called v1 is the parent node of v2, v2 is the child node of v1, in fact, the edge is informally described that executive v1 create or call the executive v2.

4. S(T) is property indicates the set legal execution of operating system services that is executed, so IsSystem(v) stand for discriminant node which is executive system function, if a node V is the system of legal execution, IsSystem(v)= True, or IsSystem(v)= False.

5. I(T) is a class of E(T) instruction set. The set element is composed of the result of return result IsTrust, and that is associated with each of the T in the attack tree. If the behavior of vn creates or called vm, that is known to be a legitimate behavior, the system is not malicious, the directed edges represent the trustworthy behavior, IsTrust(vm, vn)= true, otherwise, the IsTrust( vm, vn) = false.

6. W is a set of malicious nodes, and each element of the set is associated with the corresponding node, and the size of the V of the node is denoted as the weight of the node w(v), the value of the node V stands for the system's malicious effects, w(v) value is to determine whether the V is a direct basis attack for malicious code. Through the analysis of the typical virus sample serial number input device,
we can give an example of the implementation of the implementation process of the virus samples generated by the implementation of the collection and its composition:

![Attack Tree](image)

**Figure 2. Instance of the weight of the attack tree.**

The specific algorithm should be divided into two steps: the first step, a behavioral measure of single node and the independent malicious weights calculation; the second step, according to the transformation of the attack tree to comprehensive analysis, the implementation of a body whether malicious code to make a judgment. As shown in Figure 2: A on B effect is 10, and also leads to the effect of C, but C not to undermine the implementation of the operation, so it will be a weight between C is set to 0, similarity, B to D file damage extent that would be private and confidential data rewrite, so more serious hazards should be set to 50, B to E on the implementation effect of 20, B to F, execution is 2, the superposition of all the effect is a malicious software, the overall implementation of the effect of 82.

According to the characteristics of nodes, the behavior index of the subject is the basis of identifying the malicious code. May wish to transform the attack tree node V related behavior set is defined as B, b(B, b(v, o) is stand for an operation behavior between the main body V and the object O. Correspondingly, definite the influence index of the effect. For malicious code, the behavior characteristics of existence general character can be roughly classified, for example, the virus has self-propagation index, self-activation index, self-protective index and damage index, then the definition for the impact of malicious, k(v), k_d(v), k_p(v), k_d(v), respectively, which indicated that main V since the spread of the index, self-activation index, self-protective index and damage index value, 0 said the system without impact and the greater value the greater impact. For B, the object also has the impact on the system. With the right of α, β, χ, δ to represent self-propagation index, self-activation index, self-protective index and damage index in the calculation of malicious agents weights of embodied weight, the weight value set is an important factor influencing the accuracy of the detection algorithm can be adjusted through a lot of experiments.

(3) Calculation of independent malicious weight

Independent malicious weights calculation, hypothesis main subject V has n objects, each object 

\[ K(o) = \alpha k_1(o) + \beta k_2(o) + \chi k_3(o) + \delta k_4(o) = \sum_{j=1}^{n} K(o); \]

of the system impact index is 

\[ K(o) = \alpha k_1(o) + \beta k_2(o) + \chi k_3(o) + \delta k_4(o) + q = \sum_{j=1}^{n} K(o); \]

then the main subject malicious weights for

According to the above definition methods to calculate the independent malicious weights reflect is the body after the implementation of the system influence degree. As a matter of fact, in the system, a part of a legitimate software also have self-activation and self-protection, this kind of software is mainly use to install software. ∀v ∈ V(T), node V’s malicious comprehensive weight w(v), w(v) is in the independent node behavior analysis results based on the modification of tree T relationship
between the level of comprehensive calculation results. The specific calculation process is divided into two steps: the first step: to reflect all the malicious nodes of the parent node, the calculation results are recorded as $w'(v)$.

1. If $V$ is a leaf node, so $w'(v) = K(v)$;

2. If $V$ is one inside node of $n$ sub node, represented by the node array $q[i]$ $(1 \leq i \leq n)$, then $w'(v) = K(v) + \sum_{i=1}^{n} K(q[i])$ is not in the actual environment, the system is often implemented in the attack tree.

The second step: the system of the implementation of the weight of the node should be cleared, with the legitimate installation of software features or trusted behavior of the implementation of the number of nodes to take the opposite number.

1. If $\text{Is System}(v) = \text{true}$, that $w(v) = 0$;

2. Suppose $v$ is within the node, and have $n$ sub node, represented by the node array $q_i$ $(1 \leq i \leq n)$, then if there is $q_i$ $(1 \leq i \leq n)$ make $(v, q_i) \in E(T)$, and $\text{Trust}(v, q_i) = \text{True}$, so $w(v) = -w'(v)$.

4 Experiment and analysis

4.1 Experiment and analysis of threat situation assessment

In order to verify the rationality of the attack evaluation algorithm, we select the samples of the virus to represent the serial number input device in detail, in the experiment, the weight of $\alpha$、$\beta$、$\chi$、$\delta$ can be set for 1, modify the PE file format the self-propagation behavior index and damage index were add 1, the same as the corresponding index of behavior, such as modifying the behavior of the disk from the start since the activation index is increased by 2, to modify the registry related startup items will make self-activation index plus 1.

And 400 virus samples were tested, the results of these malicious code in the experimental results were randomly selected 55 records for statistical. According to the recognition algorithm based on the integrated behavior characteristic, a random sample of 55 experimental samples showed correctly identify the malicious code 50, the correct rate of above 90%, the false negative rate is less than 10%. Through the further analysis indicated that these samples did not produce damaging behavior on the system, analysis the reason mainly has two aspects: first, some malicious code using loopholes in the system to carry out attacks, when testing system patch version after the update does not have this part of the malicious code attacks; second, some malicious code although the completion of the attack, but little effect on the system, such as creating the some text files etc. Actually, the final solution of the malicious weights calculation results show the malicious code on the extent of the damage, for example, a PE virus sample analysis results for 4512, said the extent of damage is enormous, according to specific analysis found that the sample destruction of the disk of the vast majority of executable files (similar to panda burning incense virus) and its hidden in the system directory, to achieve the purpose of self-protection. Low failure rate cannot fully explain the effectiveness of an algorithm, the false alarm rate is another important index. In the experiment, eight types of typical legal installation software are selected, and the results are shown in Table 1:
Table 1. The experimental results of the calculation of the behavior of the software of all kinds of legitimate software.

<table>
<thead>
<tr>
<th>Legal installation software category and name</th>
<th>Instant messaging software (QQ)</th>
<th>Web browser software (360 browser)</th>
<th>Office software (Office)</th>
<th>programming software (Eclipse)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malicious weight</td>
<td>-35</td>
<td>-20</td>
<td>35</td>
<td>-25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Legal installation software category and name</th>
<th>Document reading software (Pdf reader)</th>
<th>Drawing software (AutoCAD)</th>
<th>Video player software (QvodPlayer)</th>
<th>Download tool (Thunder)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malicious weight</td>
<td>-34</td>
<td>-566</td>
<td>36</td>
<td>-64</td>
</tr>
</tbody>
</table>

From table 1, the most representative software have been tested. The experimental results show that the vast majority of legitimate software malicious weight less than or equal to zero, zero said without any malicious behavior, and for less than zero, the vast of value is not zero, that is to say the software with the sensitive operating characteristic has similar to malicious code behavior. However, due to the legal software, its own behavior characteristics, after the successful capture of these features into a negative number of computing results, so it is effective to distinguish between malicious code. Due to the current trustworthy behavior characteristics of finite sets, so with video software QvodPlayer is programming software error for malicious code, indicating that the main work of the next step will be placed in expanding the legitimate software trustworthy behavior feature set. Finally, the results of a comprehensive analysis of the experimental results of a variety of samples are shown in figure 3:

![Figure 3. Sample and behavior feature analysis.](image)

In order to show the effect of the graph, the graph ignores the absolute value of more than 60 of the behavioral characteristics of statistical results. Figure in the diamond marker said malicious code samples of malicious weights calculated results for distribution, square marked as legitimate software installation of malicious weights calculated results for distribution, in accordance with the algorithm proposed in this paper, zero is boundary between a malicious code and legitimate software (results for zero decision for legal code), the experiments are carried out under the premise of failing to get the sample feature codes, so all samples in this experiment are regarded as the unknown types of malicious code. The final results show that this algorithm has a positive significance for the identification of unknown attack codes.
5 Conclusion

In this paper, we first study the structural characteristics of the shellcode and exploit attack module, and combine the relationship between the two, which is based on the independence and dependence, to explore the effective way of the attack. Then through into the reform after the attack tree model, accurately describes the malicious code execution. Based on attack tree of malicious code detection algorithm, in line with the development trend of the malicious code detection technology. For a large number of virus samples test results show that, the method for unknown malicious code, including code detection can achieve lower false positive rate and false negative rate. In order to further reduce the false positive rate, the extraction and enrichment of the feature set of the legitimate software trusted behavior will be the next step to be studied.

6 References


