Software Behavior Model Based on Multi-attribute Decision Making of System Call

1Hongyan Ma, 2Fangfang Ou, 3Nan Yang and 3Zhen Li
1Industrial and Commercial College, Hebei University, 071002, Baoding, China
2College of Mathematics and Computer, Hebei University, 071002, Baoding, China
3CNPC Beijing Richfit Information Technology Co., Ltd., 100007, Beijing, China

Abstract: The system call attributes of traditional software behavior model based on system call are only related to deterministic attributes, such as system call name, system call context and system call argument policy which are not enough to model the software behavior accurately. For the problem, the fuzzy attributes of system call are introduced based on the traditional software behavior model and a software behavior model based on multi-attribute decision making of system call is presented. The model adopts information entropy to decide fuzzy attributes weight objectively and provides anomaly detection of fuzzy attributes based on interval data by constructing the trusted model of fuzzy attributes. The experimental results verify the effectiveness of trusted model of fuzzy attributes based on interval data and the high attack detection capability against the actual software.

Key words: software behavior, anomaly detection, system call, multi-attribute decision making, fuzzy attribute

INTRODUCTION

With the development of computer and network technology, software plays an important role in the information society. The expansion of software makes software flaws and vulnerabilities difficult to avoid. If the software behavior is always accordant with the expected behavior, we call the software is trustworthy (Wang et al., 2006). With continuous deepening of the software application in the sensitive fields such as finance, military affairs and economy, the trustworthiness requirement of software becomes more urgent.

The idea of trusted computing is to perfect the terminal computer fundamentally, while the present trusted computer only guarantees the static security of system resources. How to guarantee and measure the dynamic trustworthiness during the software running has become a key problem for software trustworthiness. The research on dynamic trustworthiness during software running has very important significance. When the software is untrustworthy, its behavior will deviate from the normal trace. Therefore, modeling normal behavior of software has become a current research focus in software trustworthiness.

The organization of this study is then described as follows. In Section Related Work, we give the related work on software trustworthiness and software behavior models. In Section Software Behavior Model, a software behavior model based on multi-attribute decision making of system call will be presented. In the next two sections, deterministic attribute decision making and fuzzy attribute decision making based on interval data will be discussed respectively. In Section Experiments and Analysis, the effectiveness of trusted model of fuzzy attributes based on interval data and the high attack detection capability against the actual software will be verified by experiments. In Section Conclusion, we will give concluding findings and future work ideas.

RELATED WORK

In recent years, researches on software trustworthiness have been paid more and more attention to by domestic and international researchers. Wang’s research team (Wen et al., 2010; Ding et al., 2011) introduce aspect-oriented architectural design approach and relevant techniques into the design and analysis of software, offer an effective approach of software architectural design for trusted software based on monitoring and a component model that supports the online fine-grained adjustment to software adaptability. Xu et al. (2013) propose a dependable entity model for Internetware and present a method of trust measurement based on Bayesian networks. Wang et al. (2012) present a verification model for trustworthiness of interaction between software component by combining the Unified

Corresponding Author: Hongyan Ma, Industrial and Commercial College, Hebei University, 071002, Baoding, China 3089

Because system call is the interface provided by the operating system to access system resources, system call can reflect the features of software behavior to a great extent. A lot of software behavior models based on system call have appeared (Tuo et al., 2010). Hofmeier et al. (1998) modeled the behavior of privileged process by short sequences of system calls and constructed the normal behavior feature library for intrusion detection. Inspired by Forrest et al.’s work, many researchers proposed the software behavior models, such as Finite State Automaton (FSA) model and Pushdown Automaton (PDA) model built by static analysis of source code (Wagner and Dean, 2001), Vi-Path model built by dynamic learning (Feng et al., 2003), HPDA model combining static analysis with dynamic learning (Liu et al., 2005), software behavior model based on system objects (Li et al., 2009), anomaly detection model of program behavior combining system call with Markov chain (Frossi et al., 2009), automaton of program behavior based on system call attributes (Li et al., 2012).

The above software behavior models based on system call can be divided into two categories. One is to construct feature library of normal software behavior through statistical learning or rule mining. The other one is to model software behavior by constructing the software behavior automaton. The first one determines whether the software behavior is abnormal or not according to the feature library of normal behavior. It cannot describe the running trace of entire software. The second one can describe the running trace of software but it is not enough to model the software behavior accurately because it is confined to deterministic attributes of system call, such as system call name, system call context and system call argument. For the above problems, we propose a software behavior model based on multi-attribute decision making of system call. The model introduces the fuzzy attributes of system call based on the traditional software behavior model. It adopts information entropy to decide fuzzy attributes’ weight objectively and provides anomaly detection of fuzzy attributes based on interval data by constructing the trusted model of fuzzy attributes of system call. The model makes software behavior be monitored more comprehensively and accurately.

**SOFTWARE BEHAVIOR MODEL**

The study introduces the fuzzy attributes of system call based on HPDA model (Liu et al., 2005) and proposes a software behavior model based on multi-attribute decision making of system call. Figure 1 shows the model expressed as an automaton for a small part of a software. The model captures the deterministic attributes and fuzzy attributes of each system call issued by the software monitored and decides whether to deviate from normal software behavior according to the abnormal value of system call. The model is defined as a 5-tuple: $(S, \Sigma, T, s, A)$.

- $S$ is a finite set of states
- $\Sigma$ is a finite set of input symbols. In our model, the set of input symbols is defined as

$$\Sigma = (X \cup \text{Addr}) \cup Y \cup \{s, f\}. \quad (1)$$

![Fig. 1: Software behavior model based on multi-attribute decision making of system call](image)
where \( X = \{\text{entry, exit}\}, Y = \{\text{Syscall}\}, \Sigma \) is the empty string. Addr is the set of all possible addresses defined by the executable software and entry and exit are two new added system calls. Entry is added before each function call and exit is added after each function call. Entry and exit get the return address before and after the function call respectively. \( Y \) is the set of system calls. For each system call, there is a group of attributes which can be divided into two categories:

- **Deterministic attributes**: Once any deterministic attribute of system call deviates from the normal value, the system call is determined to be abnormal directly. These attributes includes system call name, system call context, system call argument policy, etc.

- **Fuzzy attributes**: Fuzzy attributes cannot be expressed as accurate numbers. They are fuzzy and granted the prescribed error bounds. These attributes include time interval, memory occupancy rate, CPU occupancy rate, etc. The time interval is the difference between the beginning time of current system call and the beginning time of last system call.

- \( T \) is the set of transition functions: \( (S \to \Sigma) \cdot S \)
- \( s \) is the start state: \( s \in S \)
- \( A \) is the set of accept states: \( A \in S \)

The abnormal value \( AV_i \) of system call \( S_i \), issued by the software, depends on the abnormal value of deterministic attributes \( A_d \) and the abnormal value of fuzzy attributes \( AV_f \). The formula is as follows:

\[
AV_i = \begin{cases} 1, & AV_d = 1 \\ AV_f, & \text{else} \end{cases}
\]

(2)

If \( AV_i \) is greater than the threshold \( \tau \) of abnormal value of system call, then \( S_i \) is abnormal.

**DETERMINISTIC ATTRIBUTE DECISION**

The deterministic attributes of system call considered include system call name, system call context and system call argument policy. Among them, System call context can be represented by calling stack of system call. If the calling stacks of two system calls are same, the system call contexts of them are same; otherwise, the system call contexts of them are different. For current call stack, if the return address of function main is \( a_0 \), the return addresses of internal functions are \( a_0, \ldots, a_m \), and the return address of last system call is \( a_m \), then the system call context can be expressed as \((a_0, a_1, \ldots, a_m)\).

Let \( SC = \{S_1, S_2, \ldots\} \) be the set of system calls and let \( A^n = \langle a_1^n, \ldots, a_m^n \rangle \) be the vector of formal arguments for system call \( S_i \). Each invocation \( S_i \) of \( S \) has a concrete vector of values for \( A^n \) defined as \( A^{\nu} = \langle a_1^{\nu}, \ldots, a_m^{\nu} \rangle \). An argument set for system call \( S_i \) in the system call context \( C \) is the set of all argument vectors \( A^{\nu} \) observed for the chosen system call issued in the context \( C \). This is denoted by \( AS(C, S_i) \).

We limit our attention to unary and binary relations on system call arguments in our model.

Unary relations are attributes of a single argument. They can be represented using the form \( A^{\nu} \). Such a relationship can be learnt from software behavior traces. The set \( AS(C, S_i) \) is represented using an enumeration by listing all elements of \( A^{\nu} \) in the given system call context \( C \).

Binary relations are relationships between two system call arguments or between one system call argument and another system call return value. Let's take equality relation represented as equal for example. The file descriptor returned by open system call in context \( C_j \) is equal to the operand of a subsequent write system call in context \( C_p \).

\[
fld(C_i, \text{open}) = \text{equal} \quad \text{fd}(C_j, \text{write})
\]

(3)

Considering the characteristics of deterministic attributes of system call, if any deterministic attribute deviates from the normal value, the abnormal value of the system call is 1; otherwise the abnormal value is 0.

**FUZZY ATTRIBUTE DECISION MAKING BASED ON INTERVAL DATA**

**Trusted model of fuzzy attributes**: The normal values of fuzzy attributes cannot be expressed as accurate numbers and are granted the prescribed error bounds. Therefore, the normal values of fuzzy attributes need to be changed from a large sample of raw data to symbol data by training. Each fuzzy attribute of system call can be represented as interval data.

Let the number of fuzzy attributes be \( n \) and \( a_j^{\nu_0}, a_j^{\nu_1}, \ldots, a_j^{\nu_k} \) be the sample values of the \( j \) th fuzzy attribute of system call \( S_j \). First of all, fuzzy attributes need to be dealt with dimensionless method. The \( k \) sample values of the \( j \) th fuzzy attribute of system call \( S_j \) after pretreatment are denoted by \( a_j^{\nu_0}, a_j^{\nu_1}, \ldots, a_j^{\nu_k} \):

\[
a_j^{\nu_l} = \frac{A_j^{\nu_l} + P_l}{e_j^\nu} \quad (1 \leq j \leq n, 1 \leq l \leq k)
\]

(4)

Where:

\[
\rho_j = \frac{\sum a_j^{\nu_l}}{k}
\]
\[ e_i = \sqrt{\sum_{j=1}^{k} (\bar{p}_i - p_i)^2 / (k-1)} \]

\( \bar{p}_i \) and \( e_i \) denote the mean and standard deviation of the jth fuzzy attribute \( P_i \) respectively.

We use n-dimensional data vector \( X \in \mathbb{R}^n \) to describe the normal scope of the fuzzy attributes after pretreatment of training samples:

\[ x_i = ([p_{i1}, p_{i2}], [p_{i1}, p_{i2}], \ldots, [p_{in}, p_{in}])^T \]

Where:

\[ p_{i1} = \min_{1 \leq j \leq n} (a_{ij}^l), \quad p_{in} = \max_{1 \leq j \leq n} (a_{ij}^u) \]

The normal scopes of n attributes of system call \( S_i \) form a normal scope hyper-rectangle \( NH_{i\alpha} \) denoted by \( [u_{i\alpha}, v_{i\alpha}] \), where \( u_{i\alpha} = (p_{i1}, p_{i2}, \ldots, p_{in})^T \) and \( v_{i\alpha} = (p_{i1}, p_{i2}, \ldots, p_{in})^T \). The dimensions of \( NH_{i\alpha} \) are smaller than \( NH_{im} \). \( CH_{i\alpha} = [u_{i\alpha}, v_{i\alpha}] \), where \( u_{i\alpha} = (\bar{p}_i, \bar{p}_i, \ldots, \bar{p}_i)^T \), \( v_{i\alpha} = (\bar{p}_i, \bar{p}_i, \ldots, \bar{p}_i)^T \). \( CH_{i\alpha} \) has the same midpoint \((u_{i\alpha}+v_{i\alpha})/2\) as \( NH_{i\alpha} \) and it can be adjusted by the value of \( \gamma \), \( \bar{u}_{i\alpha} = u_{i\alpha} + \gamma \), \( \bar{v}_{i\alpha} = v_{i\alpha} - \gamma \). Boundary credible interval hyper-rectangle \( BH_{i\alpha} \) is denoted by \([u'_{i\alpha}, v'_{i\alpha}]\), where \( u'_{i\alpha} = (p_{i1}, \ldots, p_{i1}, \ldots, p_{in})^T \), \( v'_{i\alpha} = (p_{i1}, \ldots, p_{i1}, \ldots, p_{in})^T \), \( u'_{i\alpha} = u_{i\alpha} + \gamma \), \( v'_{i\alpha} = v_{i\alpha} - \gamma \).

The trust level can be divided into three levels: credible level, boundary credible level and incredible level. The credible region \( CR_{i\alpha} \), boundary credible region \( BCR_{i\alpha} \) and incredible region \( ICR_{i\alpha} \) of fuzzy attributes can be obtained through the set operations of \( CH_{i\alpha} \) and \( BH_{i\alpha} \):

\[ CR_{i\alpha} = CH_{i\alpha} \cap BCR_{i\alpha}, \quad BCR_{i\alpha} = BH_{i\alpha} \cap CH_{i\alpha}, \quad ICR_{i\alpha} = BH_{i\alpha} \quad (5) \]

**Computation of abnormal value of fuzzy attributes:** For system call \( S_i \), the vector of fuzzy attributes \( V_i \) is denoted by \([p_{i1}, \ldots, p_{in}]\). In the jth dimension, the distance between \( V_i \) and \( CR_{i\alpha} \) is expressed as follows:

\[ d_j(V_i, CR_{i\alpha}) = \begin{cases} 0, & p_j \in CR_{i\alpha} \\ \min(||p_j - p_{i1}||, ||p_j - p_{in}||), & p_j \in BCR_{i\alpha} \end{cases} \quad (6) \]

Then we can calculate the abnormal value of fuzzy attributes \( AV_f \).

\[ AV_f_i = \begin{cases} 1, & \exists p_j \in CR_{i\alpha} \\ \tanh \left( \sum_{j=1}^{n} w_j d_j(V_i, CR_{i\alpha}) \right), & \text{else} \end{cases} \quad (7) \]

where, \( W_i \) is the weight of the jth dimension fuzzy attribute.

We give the computational method of \( W_i \) below. For k samples of system call \( S_i \), after dealt with dimensionless method, the jth sample \( X \) is denoted by \([a_{ij1}, a_{ij2}, \ldots, a_{ijn}](1 \leq j \leq k)\). The sample data should be pretreated according to Eq. 8:

\[ a_{ij}^* = \begin{cases} 1.0 - \frac{\bar{p}_i - a_{ij}^l}{c_i}, & a_{ij}^l < \bar{p}_i \\ 1.0, & a_{ij}^l < \bar{p}_i, a_{ij}^u > \bar{p}_i \\ 1.0 - \frac{a_{ij}^u - \bar{p}_i}{c_i}, & a_{ij}^u > \bar{p}_i \end{cases} \quad (8) \]

where \( c_i = \max(\bar{p}_i - m_i, M_i - \bar{p}_i) \), \( M_i \) and \( m_i \) are the upper bound and lower bound of \( a_{ij}^l \), respectively.

We adopt information entropy to decide fuzzy attributes' weight objectively according to the dispersion degree of fuzzy attributes. Let the possible values of the jth fuzzy attribute of k samples be \( V_{ij} \), the entropy of the jth fuzzy attribute is as follows:

\[ e_i = -\sum_{j=1}^{k} q_j \ln q_j \quad (9) \]

Where:

\[ K = \begin{cases} 1, & h_i = 1 \\ \frac{1}{\ln h_i}, & h_i \geq 2 \end{cases} \quad (10) \]

And \( q_j \) is the probability of the value \( V_{ij} \) in k samples:

\[ q_j = \frac{|V_{ij}|}{\sum_{i=1}^{k} |V_{ij}|} \quad (11) \]

\(|V_{ij}|\) is the frequency of \( V_{ij} \) for the jth fuzzy attribute.

The weight of the jth fuzzy attribute can be calculated as follows:

\[ w_j = \frac{1-e_i}{\sum_{j=1}^{n} (1-e_j)} \quad (12) \]
EXPERIMENTS AND ANALYSIS

Anomaly detection of fuzzy attributes based on interval data: Our experiment uses machine learning databases (MLDBs) from UCI Repository (http://archive.ics.uci.edu/ml/index.html) including lots of different databases. This study selects Wine database which decides the place of origin through chemical analysis. It has 13 attributes which are all continuous real variables and has three classes. We select Class 1 as the normal dataset (60 samples in total) and Class 2 and Class 3 as abnormal datasets.

Because Wine database is high dimension data and the fuzzy attributes of system call usually have a lower dimension, we take any 6 attributes from 13 attributes in our experiment. We train the first 55 samples of class 1. Let $M_i = 5$ and $m_i = -5$ during pretreatment. The attributes' entropy can be computed according to Eq. 9. Figure 2 shows the entropies of six attributes when $\tau$ is 0.2, 0.5, and 0.8.

When $\tau$ is 0.2, the credible region $CR_{\alpha}$ is relatively large and the boundary credible region $BCR_{\alpha}$ is relatively small. The difference among the entropies of attributes calculated is very little. Therefore, the dispersion degree of attributes cannot be shown in a better way. When $\tau$ is 0.8, the attributes which values are centralized originally become dispersed because the credible region is too small which doesn’t correspond with our requirement. Therefore, we select the compromise value $\gamma = 0.5$. Table 1 shows the weights of attributes computed when $\gamma = 0.5$.

The last five samples in Class 1 and all samples in class 2 and 3 are tested. Let $\gamma = 0.5$, $\tau = 0.25$. All the abnormal values of the test samples in Class 1 are less than $\tau$ and these test samples are determined as normal. The abnormal values of 94.3% test samples in class 2 and 82.7% test samples in class 3 are greater than $\tau$ and these test samples are determined as abnormal. The rest test samples in class 2 and 3 cannot be determined as abnormal only by these six attributes. Table 2 shows the detection accuracies of two methods of weight calculation for test samples. The weight of each attribute for average method is about 0.17. The results show that the detection accuracy of the information entropy method is higher than that of the average method.

Attack detection capability: We have made experiments on a PC with Intel(R) Core(TM)2 Duo E7500 2.93 GHz and 2 GB of main memory running Linux kernel 2.4.20. Each system call is intercepted by Loadable Kernel Module (LKM) and modified to capture the attributes' value of the system call. We form the software behavior model of vi 6.1 in Red Hat 9 Linux after training and make some attacking experiments.

Red Hat 9 Linux distribution includes vi 6.1 which exists potential TOCTTOU (Time of Check to Time of Use) vulnerabilities (Wei and Pu, 2005). Specifically, if vi is run by root to edit a file owned by a normal user, then the normal user may become the owner of sensitive files such as /etc/passwd. The sequence of system calls that the vulnerability window <open, chown32> is related to is as follows:

```
..., open, write, close, chown, stat64, chown32, chmod, ...
```

Let, $M_i = 5$, $m_i = -5$, $\gamma = 0.5$

Attack 1: The attack consists of a tight loop constantly checking whether the owner of the winame file has become root. Once this happens, the attacker replaces the file with a symbolic link to /etc/passwd. When vi exits, it should change the ownership of /etc/passwd to the attacker.

If the attack succeeds, when vi runs to chown32 system call, the abnormal can be detected by our model because the attribute is not satisfied with the following argument policy:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Alcohol</th>
<th>Malic acid</th>
<th>Ash</th>
<th>Nonflavanoid phenols</th>
<th>Color intensity</th>
<th>Proline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>0.17</td>
<td>0.10</td>
<td>0.17</td>
<td>0.17</td>
<td>0.19</td>
<td>0.20</td>
</tr>
</tbody>
</table>

Table 1: Weights of Attributes when $\gamma = 0.5$

<table>
<thead>
<tr>
<th>Method</th>
<th>Class 1</th>
<th>Class 2</th>
<th>Class 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information entropy</td>
<td>100%</td>
<td>94.3%</td>
<td>82.7%</td>
</tr>
<tr>
<td>Average method</td>
<td>100%</td>
<td>76.3%</td>
<td>69.2%</td>
</tr>
</tbody>
</table>

Table 2: Detection accuracies of two methods for test samples

Fig. 2: Entropies of attributes for wine dataset
Fig. 3: Average time intervals and the time intervals when attack 2 happens

chown32(wfname) equal open(wfname)

The abnormal value of chown32 system call is 1, so the software behavior deviates from the normal.

**Attack 2**: The attack is against fuzzy attributes of system call. We take time interval attribute of system call for example and just focus on time interval. We inject a segment of code between the first chmod and stat64 system call. The segment of code doesn’t invoke system call, doesn’t change the arguments of stat64 and doesn’t change memory occupancy rate and CPU occupancy rate basically.

Figure 3 shows the average time intervals of system calls gained during the training phase and the time intervals when attack 2 happens. From Fig. 3, we can see that write and stat64 system calls have a relatively large deviation from their average time intervals when attack 2 happens.

The time of open system call is related to the size of file to open and it affects the time interval of write system call, so for write system call the values of time interval in the training phase are dispersed and the weight of time interval is low. While, the values of stat64 system call’s three fuzzy attributes are centralized and the difference of the weights of three attributes is very little. When attack 2 happens, for write system call, the abnormal value of fuzzy attributes calculated by (7) is 0.13 which is less than $\tau$, so write system call is normal; for stat64 system call, the abnormal value of fuzzy attributes is 1 which is greater than $\tau$, so stat64 system call is abnormal and the attack is detected.

**CONCLUSION**

The study introduces the fuzzy attributes of system call based on the traditional software behavior model and presents a software behavior model based on multi-attribute decision making of system call. The model constructs trusted model of fuzzy attributes, uses information entropy to decide fuzzy attributes’ weight objectively and provides anomaly detection of fuzzy attributes based on interval data. The attributes of system call in our model are not limited to the attributes listed in the study and have good expandability. However, there is also some limitation. Our model may miss some attacks that cause the fuzzy attributes to deviate from normal a little. The value $\gamma$ in the trusted model of fuzzy attributes and the threshold $\tau$ of abnormal value of system call should be adjusted for better effect of intrusion detection.

**ACKNOWLEDGEMENTS**

This work was supported by the Natural Science Foundation of China (Grant No.11101115), the National Natural Science Foundation of China (Grant No. 61170254), the Natural Science Foundation of Hebei Province (Grant No. F201201039) and the Science and Technology Research and Development Guidance Plan Project of Baoding City (Grant No. 13ZG012).

**REFERENCES**


