Scheduling Algorithm Based on Heterogeneity and Confidence for Mimic Defense

Wenjian Zhang*, Shuai Wei, Le Tian, Ke Song and Zhengbin Zhu

Information Engineering University, Zhengzhou 450002, China
E-mail: wenjian0509@163.com
**Corresponding Author*

Received 11 August 2020; Accepted 25 September 2020;
Publication 16 December 2020

Abstract

As a defense technology with endogenous security, mimic defense plays an important role in network security research. The scheduling of executors is one of the severe problems to take into account for mimic defense, and current research lacks comprehensive consideration of the influence of system architecture and attack behavior on scheduling algorithm. Based on previous research, this paper first introduces concept of heterogeneity and confidence according to vulnerability attributes and attack distribution characteristics to characterize the executors. Moreover, the TOPSIS (Technique for Order Preference by Similarity to an Ideal Solution) algorithm is brought in to optimize the system security and improve operating efficiency. Experimental results showed that,compared with the existing algorithms, Random, MD, RSMS, it improves the security of the system in non-uniform distributed attack scenario and the operating efficiency in each attack scenario.

Keywords: Mimic defense, security, heterogeneity, confidence, TOPSIS, operating efficiency.

Journal of Web Engineering, Vol. 19_7–8, 971–998.
doi: 10.13052/jwe1540-9589.19783

1 Introduction

With the rapid development of network information technology, people's living and economic standards have been greatly improved. At the same time, cyber network attacks are becoming more common. In order to solve the network security issues, the current network security technology [1] is mainly through patching, i.e., by adding security modules to the network. Meanwhile, some new defense technologies have emerged in recent years, such as moving target defense technology (MTD) [2], mimic defense technology [3], etc. By introducing diverse technologies to enhance the uncertainty and polymorphism of the system, MTD tries to minimize the exposure time of system vulnerabilities and increase the difficulty of attack, thereby ensuring system security [4]. Proposed by academician Jiangxing Wu, the mimic defense theory [5] increases the security of the system by introducing dynamics, heterogeneity, redundancy, and negative feedback into the system based on the attributes of the system itself. Compared with MTD technology, mimic defense technology additionally introduces arbitration and negative feedback mechanism, so that the attack surface of the system is narrowed down. The dynamic adjustment of the internal structure of the system, greatly increase the cost of the attackers. At present, the mimic defense theory has been researched and practiced in SDN [6, 7], Web service [8], cloud server [9] and other fields [10, 11], and has shown a better defense effect.

The research of mimic defense technology focuses on the security impact of architecture, scheduling space, scheduling timing and negative feedback characteristics. Among them, the scheduling algorithm of executors is the key technology. There are currently many studies related to scheduling algorithms for mimic defense technology. Starting from the mimic architecture, Hu [12] evaluated the impact of the scheduling period and the number of redundant executors on security gain, and analyzed the impact of the fixed period and the adaptive scheduling period on the security. Guo [13] proposed a scheduling algorithm based on sliding time window for trial-and-error attacks. Compared with Hu, the adaptability of the algorithm can better reflect the role of negative feedback characteristics in mimic defense. Considering the diversity of the executors, Qi [14] proposed a dynamic scheduling algorithm based on the security policy to evaluate the security of the mimic SDN controller, the proposed algorithm MaxSG is more secure than the random algorithm (Random). From the perspective of SDN service deployment, Li [15] introduced mimic defense technology to increase heterogeneity in scheduling parameters, and evaluated the impact of the scheduling algorithm based on

the maximum degree of heterogeneity (MD) on the security of the mimic system. Liu [16] proposed a random seed minimum similarity algorithm (RSMS), which introduces randomness and employs similarity as a scheduling parameter. Compared with other algorithms, RSMS has achieved better balance effect in terms of randomness and security.

There are two problems in the current research on scheduling algorithms. On the one hand, the definition of heterogeneity is mostly based on the heterogeneity between two executors. This method brings more security gain on the 3-on-line executors mimic system, but for a mimic defense system in which online executors are more than 3, the security gained is not obvious. On the other hand, the outputs of online executors reflect the historical attack behavior of attackers to a certain extent, but the current research scheduling algorithm lacks the consideration of attack behavior information.

From the perspective of system vulnerability attributes and attack history behavior, this paper proposes a mimic scheduling algorithm based on heterogeneity and confidence (Heterogeneity & Confidence, HET-CON) for different attack scenarios. Innovative work and contributions mainly include the following aspects:

1. A new quantized model of heterogeneity is proposed based on the high-order symbiosis of vulnerabilities.
2. Sliding window confidence is established and applied to executor scheduling, which increases the defense performance of non-uniformly distributed attacks, thereby improve the anti-attack performance of the system.
3. Combined with heterogeneity and sliding window confidence, a scheduling algorithm suitable for both uniformly distributed and non-uniformly distributed attack scenarios is constructed.
4. The simulation environment is constructed, and the results show that HET-CON achieves a better trade-off in terms of security and operating efficiency compared with other algorithms.

Section 2 describes the DHR architecture, threat models and evaluation criteria. In Section 3 the optimized mimic scheduling algorithm is proposed based on confidence and heterogeneity, and the security and effectiveness of the algorithm is analyzed; Section 4 set up a simulation environment and compare with several typical algorithms to verify the effectiveness of the proposed scheduling algorithm. Finally, we summarize our work and analyze the significance of proposed algorithm, pointing out the direction for further improvement in Section 5.

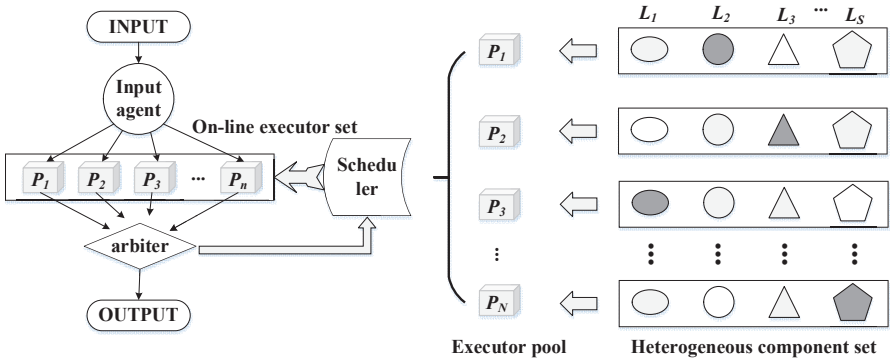


Figure 1 Structure of DHR.

2 Model Establishment and Evaluation Criteria Analysis

The basic architecture of cyberspace mimic defense is dynamic heterogeneous redundancy (DHR), which is shown in Figure 1. It mainly includes six parts, including input agents, heterogeneous executor pools, heterogeneous components set, scheduler, online executor set and arbiter. Among them, the input agent is responsible for the distribution of ingress data. The principle of input agent is replication and distribution, that is copying the ingress data into n copies and distributing them to n heterogeneous executors with functional-equivalent and structural-difference. Each executor is independent and processes the input data in parallel. And then, their respective results are summarized to the arbiter; the arbiter generates voting results with a certain voting algorithm. In addition, the result of the arbiter will be fed back to the scheduler. According to the current situation, the scheduler decides whether it needs to use a specific scheduling algorithm to select some executors from a heterogeneous executor pool to be on-line, and clean and restore the states and data of executors which will be off-line; each executor is composed of elements belonging to the component sets. It is precisely because of the different distribution of these elements among the executors that heterogeneous executor pools are formed. The dynamicity, heterogeneity, and redundancy of DHR makes the system capable of uncertainty in time and space, making it difficult for attackers to exploit the vulnerabilities of the system, and then enables the system to get endogenous security characteristics and natural immunity.

The symbols involved in this paper are shown in Table 1.

Table 1 Definition of algorithm symbols

Symbol	Meaning
$\mathbf{\Omega}$	Executor set in executor pool
$\mathbf{\Theta}$	Online executor set
N	Number of executors in executor pool
M	Number of online executors
P_i	The i-th executor
$\mathbf{\Gamma}$	Executor component set
$\mathbf{L}_k$	Category k component
Z_k	Number of component implementation schemes of category k
$\mathbf{V}$	Vulnerability set
$\mathbf{\Lambda}$	Result set of executor output

2.1 DHR Model

The executor set of executor pool is $\mathbf{\Omega} = \{P_1, P_2, \ldots P_N\}, |\mathbf{\Omega}| = N$. There are M online executors which form a set, $\mathbf{\Theta} = \{P_{i_1}, P_{i_2}, \ldots, P_M\}, |\mathbf{\Theta}| = M$, which is a subset of $\mathbf{\Omega}$. The component set of all executor is $\mathbf{\Gamma} = \{L_1, L_2, \ldots, L_S\}, |\mathbf{\Gamma}| = S$, and a single function equivalent executor consists of S kinds of components. The implementation schemes of executor components are $\mathbf{L_k} = \{L_k^1, L_k^2, \ldots, L_k^{Z_k}\}, |\mathbf{L_k}| = Z_k$, a single component $\mathbf{L_k}$ has Z_k realization scheme $(Z_k \leq N)$.

Assuming that there is an executor pool composed of five executors in the DHR, then the relationship of the executor set, the component set, and the component implementation scheme is as follows. In reality, due to the need of integration between various components and components, the maturity of the components and overall operating efficiency need to be considered when executors are composed. Therefore, the combination of components is limited. For example, certain applications are only suitable for a specific type of middleware.

$$\mathbf{\Omega} = \begin{pmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \end{pmatrix} = \begin{pmatrix} L_1^1 & L_2^1 & L_3^3 & L_4^3 \\ L_1^2 & L_2^2 & L_3^2 & L_4^1 \\ L_1^1 & L_2^2 & L_3^1 & L_4^2 \\ L_1^2 & L_2^1 & L_3^2 & L_4^2 \\ L_1^3 & L_2^3 & L_3^1 & L_4^1 \end{pmatrix} \tag{1}$$

Vulnerability set $\mathbf{V_\Omega} = \bigcup_{\mathrm{i}} V_{P_\mathrm{i}} = \bigcup_{\mathrm{i}}((\bigcup_j V_{L_\mathrm{j}})|P_\mathrm{i})$, where V_{P_i} represents the vulnerability set of the *i*-th executor and represents the vulnerability of the *j*-th component in the *i*-th executor.

Symbiotic vulnerability refers to the common vulnerability between different implementations of the same component with same function. The common vulnerabilities between two component implementations can be expressed as $V_{L_\mathrm{k}^i} \bigcap V_{L_\mathrm{k}^j}$, then the number of symbiotic vulnerabilities is $|V_{L_\mathrm{k}^i} \bigcap V_{L_\mathrm{k}^j}|$. Finally, the symbiotic vulnerabilities between two executors can be expressed as $V_{P_\mathrm{i}} \bigcap V_{P_j}$, and the number of symbiotic vulnerabilities is $|V_{P_\mathrm{i}} \bigcap V_{P_j}|$.

2.2 Threat Model

As shown in Figure 1, the input agent, scheduler, and arbiter can be protected by using hardware logic. The input agent is only responsible for the composition of number logic, and the arbiter is only responsible for data comparison and configuration interaction. The unreachable characteristics of scheduler and feedback controller can be proved by formal analysis. The probability of the mimic system therefore being compromised is only related to the executor vulnerabilities. In order to analyze the security of mimic system, the threat model need to be simplified to a certain extent, and the specific simplified processing is given in the form of hypothesis.

Hypothesis 1: Every attack on the vulnerability/backdoor can cause abnormal output performance.

For the usual attack methods, such as a certain middleware with a backdoor listening port, it will inevitably produce different outputs when communicating with it. Most of the attacks require the output to determine the attack effect, and some attacks are passed unconventional attack methods such as side channel, but the probability is small and it is not considered in this paper.

Hypothesis 2: When the vulnerabilities/backdoors are attacked in the system, the executors that share the vulnerabilities behave the same.

This assumption is true in most cases. For example, a certain middleware with a backdoor listening port, etc. It may not be true in some cases, such as buffer overflow attack. Although the application has the same buffer overflow vulnerability, due to the difference of underlying processors, the same binary

code can only be executed on specific platforms (such as ARM-based processors). While running on other platforms (such as X86-based processors), ordinary buffer overflow attacks cannot work, but attackers can also employ advanced methods such as branch instructions to attack to produce the same performance.

Hypothesis 3: The attacker has the same attack success probability to each vulnerability/backdoor of the system.

For a single component, the programmer will leave a vulnerability / backdoor on an average of 1,000 to 1,500 lines of code. The total number of vulnerabilities/backdoors of the component can be estimated in turn, and the ratio of the total vulnerabilities/backdoors can be used to estimate the probability of each attack. Then the number of vulnerabilities in executor can be obtained by adding the number of vulnerabilities in all components.

Hypothesis 4: Only known vulnerabilities of components are considered for heterogeneity evaluation.

Due to the unpredictability of vulnerabilities, the same component has the same function and the amount of code implemented is basically the same. If the level of code writing is similar, the number of vulnerabilities is basically the same. The distribution of known vulnerabilities and unknown vulnerabilities of each component is approximately the same. The security of the mimic system therefore can be approximately evaluated by the distribution of known vulnerabilities.

Based on the above assumptions, we conclude in the DHR model as below.

When the *k*-th component is attacked, the probability of the *i*-th implementation of the component being attacked can be expressed as

$$p(L_{\mathrm{k}}^{\mathrm{i}}) = \frac{|V_{L_{\mathrm{k}}^{i}}|}{\left|\bigcup_{j=1}^{Z_k} V_{L_{\mathrm{k}}^{j}}\right|} \tag{2}$$

That is, the probability of class i component being attacked is the ratio of the number of vulnerabilities to the total number of vulnerabilities in all implementations. Similarly, we can get the probability of the *i*-th executor being attacked when the mimic system is attacked.

$$p(P_i) = \frac{|V_{P_i}|}{\left|\bigcup_{j=1}^{M} V_{P_i}\right|} \tag{3}$$

2.3 Evaluation Metrics

The performance of mimic system needs to be evaluated from two aspects: system security and system operating efficiency. Security is the basic metric to evaluate the anti-attack ability of the mimic system, and the system operating efficiency reflects the performance overhead of the mimic system.

1. Security metric

The most intuitive method of judging the security of a mimic system is to evaluate its security based on whether voting result of the arbiter can tolerate the attack towards some heterogeneous executors of the mimic system. The success rate of mimic system attacks is the most intuitive indicator of its security. According to the hypothesis in the threat model, when an attacker targets a certain type of component vulnerability, it will cause the executors with the same implementation or different implementation schemes with symbiotic vulnerabilities to fail, which is different from the correct result. The online executor set is $\mathbf{\Theta}$, when the input is $\mathbf{I}$, there are m $(m < \lfloor \frac{M+1}{2} \rfloor)$ abnormal outputs in the output $\mathbf{J}$, the attack is unsuccessful; otherwise, the attack is successful. The probability of the mimic system being attacked and successful is

$$p(\Theta) = \frac{\sum |\bigcap_m V_{P_i}|}{\left|\bigcup_{j=1}^{M} V_{P_i}\right|},\ m \geq \left\lfloor \frac{M+1}{2} \right\rfloor \tag{4}$$

$\sum |\bigcap_m V_{P_i}|$ represents the number of vulnerabilities in which input incentives can lead to abnormal output of no less than $\lfloor \frac{M+1}{2} \rfloor$ executors, including symbiotic vulnerabilities in of the same component.

It can be seen that what really affect the security of the system are the symbiotic vulnerabilities and the component scheme used.

2. System operating efficiency metric

System operating efficiency: Given a DHR structure containing N heterogeneous executors, the online service time of the executor (P_i) is $T_r(P_i)$, and the offline processing time is $T_d(P_i)$. The system operating efficiency can be expressed as

$$\Upsilon = \frac{\sum_{i=1}^{N} T_r(P_i)}{\sum_{i=1}^{N} (T_r(P_i) + T_d(P_i))} \tag{5}$$

When mimic system is in scheduling time, some called executors need to be scheduled out, which ensures the safety of mimic system. However, excessive scheduling in and out will inevitably affect the online service

capabilities of the executors. System operating efficiency gives a suitable metric to evaluate the efficiency of mimic system.

3 Scheduling Algorithm Based on Heterogeneity and Confidence

3.1 Heterogeneity Metric

The quantification of heterogeneity in previous studies takes into account the complexity and differentiation between two executors. To some extent, it reflects the heterogeneity of the mimic system [17]. However, the difference between two executors cannot directly reflect the security of the system when there are symbiotic vulnerabilities in more executors of the mimic system. Therefore, we characterize the heterogeneity of the mimic system based on the high-order symbiosis of vulnerabilities.

Definition 1 (high-order symbiosis of vulnerabilities) If there are some vulnerabilities that can be achieved the same attack effect in different executors in the set of executors, and the number of executors meeting this condition is m, then we call the vulnerability is m-order symbiosis. When m $\geq$3, we call it high-order symbiosis.

On the one hand, high-order symbiotic vulnerabilities depend on the common vulnerabilities of different components. On the other hand, it also depends on whether different executors use the same components. If no less than three executors use the same components, the corresponding vulnerabilities will become high-order symbiotic vulnerabilities.

Supposing there are three equivalent heterogeneous executors P_1, P_2, P_3, the vulnerability of each executor is shown in Figure 2. Here, we can analogize the executor to a chromosome, the analogy of resources with vulnerabilities and without vulnerabilities are two kinds of gene fragments. Assuming that P_1, P_2 and P_3 all have 4 vulnerabilities, but the gene fragments corresponding to different vulnerabilities. There are two symbiosis vulnerabilities in scenario 1, and there are no high-order symbiosis vulnerabilities in scenario 2. In both scenarios, if the number of online executors is 5, then theoretically scenario 1 has the risk of being targeted by an attacker, while scenario 2 has no risk of being attacked. However, following the definition of heterogeneity in the previous study, the heterogeneity of both scenarios is equal.

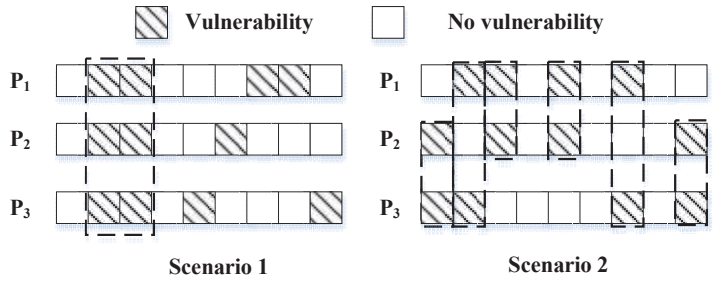


Figure 2 High-order symbiosis of vulnerability.

However, only the performance of pairwise heterogeneous executors in the executor set is considered in previous research. According to the voting property of mimic system, when the number of executors in the online executor set is $(2f + 1)$, if more than m $(m \geq f)$ executors have the same output due to the existence of symbiotic vulnerability. Then the voting result will fail, and the attack get work. Therefore, we need to consider the impact of high-order symbiosis vulnerabilities on heterogeneity. For m executors, we give the mathematical expression of m-order symbiosis vulnerabilities.

$$com_{km}(t) = \frac{\sum_{\upsilon} V_{L_k^m(\upsilon)}(t)}{\sum_{i=1}^{m} \sum_{\upsilon} V^{\mathrm{i}}_{L_k(\upsilon)}(t) - (m-1) \sum_{\upsilon} V_{L_k^m(\upsilon)}(t)}, \tag{6}$$

Among them, t represents time, υ represents vulnerabilities of a single component; meanwhile the common vulnerabilities of m components in the *k*-th component at time t is $V_{L_k^m(\upsilon)}(t)$; and $V^{\mathrm{i}}_{L_k(\upsilon)}(t)$ represents vulnerabilities of *i*-th component.

In the extreme case, when all components of the m executors are exactly the same, $com_{km}(t) = 1$. When the *m*-order symbiosis vulnerability is 0, $com_{km}(t) = 0$. From this we can derive the expression of the symbiosis evaluation of *m*-th order symbiotic vulnerabilities of k-type components in *n* executors:

$$COM_{km}(t) = \sum_{j=1}^{O} \frac{1}{O} com_{km}(t)|_j \tag{7}$$

Among them, *j* represents the specific executor in different executors of k-type component combination schemes, and O represents the number of combination schemes of *m* executors and *k*-type components selected from n executors, $O = \binom{m}{n}$, $m \geq 2$.

The *m*-th order symbiosis vulnerabilities in all component classes of executors can be expressed as

$$COM(t) = \sum_{i=1}^{S} \mu_i COM_{im}(t) \tag{8}$$

Among them, μ_i represents the proportion of the *i*-th component in the total number of vulnerabilities of the components in all executors.

According to formulas (6)–(8), we can deduce that

$$COM(t) = \sum_{i=1}^{S} \sum_{j=1}^{O} \frac{1}{O} \frac{\mu_i \sum_{\upsilon} V_{L_k^m(\upsilon)}(t)}{\sum_{i=1}^{m} \sum_{\upsilon} V^{\mathrm{i}}_{L_k(\upsilon)}(t) - (m-1) \sum_{\upsilon} V_{L_k^m(\upsilon)}(t)} \tag{9}$$

It can be seen that when the number of various components $\sum_{i=1}^{m} \sum_{\upsilon} V^{\mathrm{i}}_{L_k(\upsilon)}(t) - (m-1) \sum_{\upsilon} V_{L_k^m(\upsilon)}(t)$ in the mimic executors is fixed, the high-order symbiosis vulnerability is $V_{L_k^m(\upsilon)}(t)$. If the overall vulnerabilities remain no change, the more high-order symbiotic vulnerabilities are, the smaller the heterogeneity will be. From formulas (8)–(11)

$$H(t) = \sum_{i=1}^{S} \sum_{j=1}^{O} \frac{1}{O} \frac{\mu_i \left(\sum_{i=1}^{m} (\sum_{\upsilon} V^{\mathrm{i}}_{L_k(\upsilon)}(t) - \sum_{\upsilon} V_{L_k^m(\upsilon)}(t)) \right)}{\sum_{i=1}^{m} \sum_{\upsilon} V^{\mathrm{i}}_{L_k(\upsilon)}(t) - (m-1) \sum_{\upsilon} V_{L_k^m(\upsilon)}(t)} \tag{10}$$

In addition, the smaller $COM(t)$ is and the larger $H(t)$ is, the less likely the system will be successfully attacked, and the higher security the system is. There are the following properties for $H(t)$.

Property 1: When the overall vulnerability of mimic system is certain, *H* is negatively related to the probability of the system being successfully attacked, we record it as $H \propto 1/p(\Theta)$.

When the number of online executors is (2*f*+1), the key factor in the security of mimic system is the symbiosis of m-order $(m \geq f)$ vulnerabilities. When the online executors of mimic system need to be scheduled, it is necessary to consider selecting executors from the heterogeneous executor pool, and try to maximize the heterogeneity of the online executors.

3.2 Confidence Metric

According to the assumption (3), the maximum heterogeneity can ensure the maximum security of mimic system. However, the security of mimic system

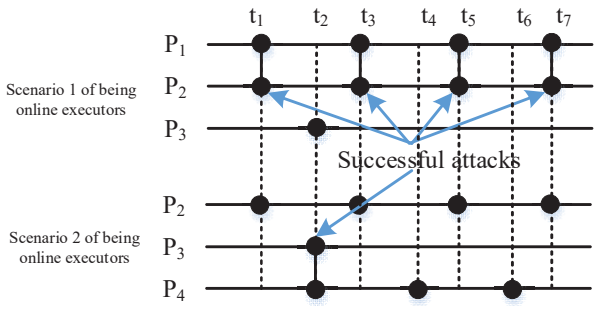


Figure 3 Analysis of specific attack scenarios.

will be affected if an attacker selectively exploits some special vulnerabilities. As shown in Figure 3, in two different online executor scenarios, the heterogeneity of online executor scenario1 is higher than online executor scenario 2. As shown in Figure 3, the symbiosis vulnerabilities in P_1 and P_2 with higher heterogeneity are under 4 attacks, and due to the difficulty of exploitation or other reasons, the attackers only targeted the symbiosis vulnerabilities in P_3 and P_4 with lower heterogeneity once. Obviously, the advantages of heterogeneity cannot be reflected in this scene. Since the mimic system needs to be adjusted according to the attack characteristics. If the historical information of executors can be used, the attacker's continuous attack on the specific vulnerabilities of mimic system can be mitigated to a certain extent.

The historical behaviors of executors can be reflected in confidence. Global confidence [18], i.e., all historical confidence performance of the executors, is widely used in most current researches. The confidence mentioned in this paper reflects the recent historical behaviors of the executor. We call it sliding window confidence. To a certain extent, the security of executor can be reflected by using the sliding window confidence. Sliding window confidence refers to the degree to which the executor can be trusted. Within a certain period of time, it reflects the degree to which the executor is not attacked. The confidence is a priori information, which means that historical behaviors of the executor when being under attack. For the executor P_i, its confidence can be expressed as

$$\psi_i(t) = \begin{cases} \dfrac{\sum_{\tau=0}^{t} \delta_i(\tau)}{t}, & t < T \\ \dfrac{\sum_{\tau=t-T}^{T} \delta_i(\tau)}{T}, & t \geq T \end{cases} \tag{11}$$

Where T represents the confidence time window, and the number of scheduling cycles calculated by the confidence is used as a quantitative metric. This parameter can be adjusted appropriately according to the application scenario of the system.

$\delta_i(t)$ is the state transition function related to the executor P_i, there are four state transitions for executor in mimic system: on-line, being online, off-line and being offline. On-line means that the executor is scheduled to the online executor from the pool of executors according to the scheduling algorithm; being online means that the executor is always online before and after scheduling; off-line means that the scheduler will transferred away the executor from the online follow the voting result in the event of an attack ; being offline means that the executor is offline before and after the scheduling. Let $\delta_i(t)$ be

$$\delta_i(t) = \begin{cases} 1, & \text{on-line or being online} \\ 0, & \text{being offline} \\ -1, & \text{off-line} \end{cases} \tag{12}$$

It is easy to deduce from (11) and (12) that

$$\Psi(t) = \sum_{i=1}^{M} \psi_i(t) = \begin{cases} \dfrac{\sum_{i=1}^{M} \sum_{\tau=0}^{t} \delta_i(\tau)}{t}, & t < T \\ \dfrac{\sum_{i=1}^{M} \sum_{\tau=t-T}^{T} \delta_i(\tau)}{T}, & t \geq T \end{cases} \tag{13}$$

Among them, $\Psi(t)$ represents the confidence of M online executors, $\psi_i(t)$ is the confidence of executor P_i, and T is the times the mimic system is scheduled when confidence is calculated. It can be seen from Equation (13) that when the total number of scheduling times is less than T, the confidence is related to the state transition of executor during the entire scheduling period; when the total scheduling times is larger than T, the confidence is only related to the state transition of executor during T scheduling times before the current moment. The significance of this evaluation method is that the confidence only considers the attack situation within a certain time window. Because most attacks are time-aggregated, the global confidence does not reflect the anti-attack of the executor in a certain period of time. Therefore, the sliding window confidence proposed in this paper has better applicability when the executors are selected in the scheduling algorithm.

3.3 HET-CON Algorithm

As analyzed above, heterogeneity is the indicator that reflects the dissimilarity of online executors. Under the condition of the same probability to exploit vulnerabilities, the greater the heterogeneity is, and the higher the security. The sliding window confidence reflects the successful attack by attackers on the executors within a certain period of time. The higher the confidence of sliding window, the better the recent attack resistance of the executors, so the higher the security is. Considering heterogeneity and confidence impact security of mimic system, we use the TOPSIS algorithm [19] to evaluate the safety of the mimic system.

$$\Phi = \alpha H(t) + (1-\alpha)\Psi(t) \tag{14}$$

Among them, α is the parameter of comprehensive evaluation weight, which is related to the specific environment. With the attacker's cognition of mimic system executors, the proportion of sliding window confidence gradually increases. Considering that overall attack resistance of the mimic system can be assessed by heterogeneity, the sliding window confidence only reflects the characteristics of the attack on mimic defense system within a certain period of time, and the value range of α can be defined as $\alpha \in (0.5, 1)$.

Assuming to set the optional schemes set of the online executors to $\mathbf{G} = [\mathrm{G}_1, \mathrm{G}_2, \ldots, \mathrm{G}_o]^T$, where $G_j = [g_1, \ldots, g_N]_{1\times N}$, and

$$\mathrm{g}_i(j) = \begin{cases} 1, & \mathrm{P}_i \text{ on} - \text{line} \\ 0, & \mathrm{P}_i \text{ off} - \text{line} \end{cases} \tag{15}$$

Then $\mathbf{G}$ is optional schemes of online executors, and $|\mathbf{G}|$ is the number of online executors. When $\mathrm{G}_i \neq \mathrm{G}_j, i \neq j$, it means that the optional schemes of online executors are different.

Suppose the output result set of executors is $\mathbf{\Lambda} = \{\lambda_1, \lambda_2\}$, where λ_1 represents the most consistent output result set, and λ_2 is the minority consistent output result set. Then the output result shall subject to λ_1. $\mathbf{K} = \mathbf{G}|\lambda_2$ can be used to indicate the executors that needs to be scheduled.

Algorithm 1: HET-CON algorithm
Input: initialize M,N,H =0,Φ=0
% heterogeneity calculating
(1) for *i*=1:O
(2) H(G_i) = Computer(G_i);
(3) H($\mathbf{G}$) = Store(H(G_i));

(4) end
(5) Reorder(H($\mathbf{G}$));
(6) $\mathbf{\Theta}$= argmax(H($\mathbf{G}$));
% scheduling
(7) Wait($\lambda_2 \neq \emptyset$);
(8) $\mathbf{\Theta'} = \mathbf{\Theta} - \mathrm{K}$;
(9) for i=1:O
(10) if($\mathbf{\Theta'} \in \mathrm{G}_i$)
(11) $\mathbf{G}^*$=Store(G_i);
(12) end
(13) $\mathbf{\Theta}$=argmax($\Phi(\mathbf{G}^*)$);
% confidence updating
(14) update(Φ) according to equation 13;
(15) jump to step 7;
Output: Θ

First of all, the heterogeneity of each executor scheme can be calculated according to formula (10), and stored in order according to the heterogeneity, then the executor scheme with the largest heterogeneity are selected as the first online executor set (line (1)–(6)). Secondly, when the executors are attacked and the voting results are inconsistent, a few inconsistent executors are scheduled out. Meanwhile we search for the executor schemes containing the remaining online executors, and calculate the value of each scheme according to formula (14). The scheme with maximum value is selected and the corresponding executors are scheduled to on-line simultaneously (line (7)–(13)). Finally, the confidence of each executor is updated according to formula (12) (line (14)). At this point, the entire scheduling process from system initialization to suffering the first attack is completed, and the subsequent scheduling jumps to line (7). In other words, when the arbiter finds that the output is inconsistent, the scheduling process will be performed again.

When calculating the heterogeneity, the time complexity is non-polynomial. Therefore, as N increases, there is a problem of complexity explosion. However, the value of N in most application scenarios is less than 10 in the mimic defense system, so the complexity of the algorithm is acceptable.

4 Results and Discussion

4.1 Simulation Environment

The heterogeneity mentioned in this paper is determined based on the vulnerability characteristics of multiple executors. The confidence refers

to the degree of attack on the executor within a certain period of time. Scheduling algorithms based on heterogeneity and confidence aims to balance system security and operating efficiency. In order to verify the effect of the proposed method, on the one hand, the scenario of DHR with a special vulnerability distribution in executors need to be constructed; on the other hand, it is necessary to construct two attack scenarios: attack characteristic oriented to equal probability distribution of vulnerabilities and attack characteristic of time-aggregated attack(for example, non-uniform distribution).

A system generally consists of applications, middleware, operating systems, etc. Here we assume that each executor includes 3 components. Considering that the multi-executors of mimic systems will increase the consumption of resources, the heterogeneous executor pools of most mimic systems generally do not exceed 10. In addition, according to the statistics of the NVD vulnerability database in the previous literature [20], operating systems with high-order symbiotic vulnerabilities generally do not exceed 4. We select the number of executors in the executor pool $N = 10$ in the simulation experiment, and the number of online executors M is 5. The simulation platform of this paper is MATLAB R2018a.

In order to facilitate simulation, we use simulation software to generate the heterogeneity between the executors automatically. The generation principle is: 10,000 vulnerabilities randomly distributed in each component of the 10 executors, and there are 1,000 placement points in each executor. According to the ratio of vulnerabilities in common applications, middleware and operating systems, and the ratio of vulnerabilities among components can be set to 1:2:7. The distribution of vulnerabilities generated by the method is shown in Figure 4. From the figure, we can see that as the number of vulnerabilities increases, high-order symbiotic vulnerabilities decrease exponentially, and the highest order is no more than 5. The high-order symbiosis distribution of the vulnerability is basically consistent with the vulnerability distribution [21] evaluated according to NVD.

It should be noted that the vulnerability placement point is preset for the convenience of simulation. The actual evaluation of the executor should be based on the actual vulnerability of the executor.

Two attack scenarios have been set up, namely uniform attack and non-uniform attack. In uniform attack scenario, the probability of each vulnerability being attacked is equal; in non-uniform attack scenario, considering that the attacker will evaluate the composition of the executors used by mimic system, we delimit unequal probability attack according to the heterogeneity

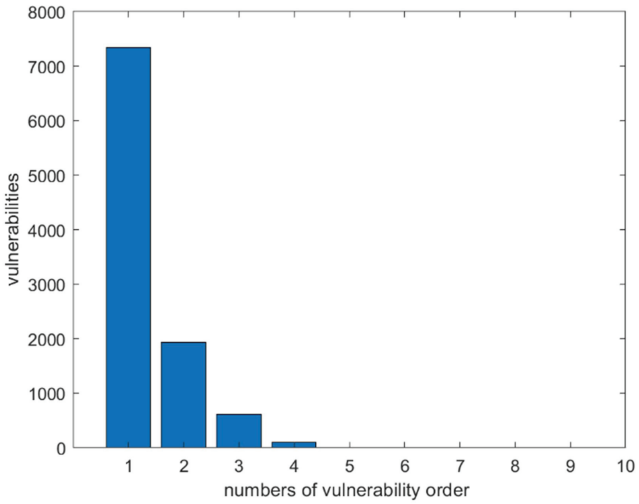


Figure 4 Vulnerability distribution.

of different executors composition schemes, and unequal probability attacks are divided into 5 groups.

In terms of algorithm comparison, since the vulnerabilities of all components are taken as the target of attack in simulation environment, the probability of traditional system being successfully attacked at this time is 100%. Therefore, the traditional system is not selected in this paper as a reference. Instead, the commonly used mimic scheduling algorithm are selected: random scheduling algorithm (Random algorithm) [14], scheduling algorithm based on heterogeneity (MD algorithm) [15] and scheduling algorithm based on random seed (RSMS algorithm) [16] . The effect of the proposed algorithm, HET-CON, is verified by comparing with these algorithms in terms of security and operating efficiency.

4.2 Simulation Result and Analysis

1. Security analysis

In the uniform distribution attack scenario, the simulation result of the attack success rate in mimic defense scenario is shown in Figure 5. The security of Random algorithm is lower because executors are selected randomly ignoring the impact of executors' heterogeneity. While MD algorithm is more secure because it always selects executors on-line with the highest heterogeneity and fewer symbiosis vulnerabilities. When the system is attacked, MD algorithm

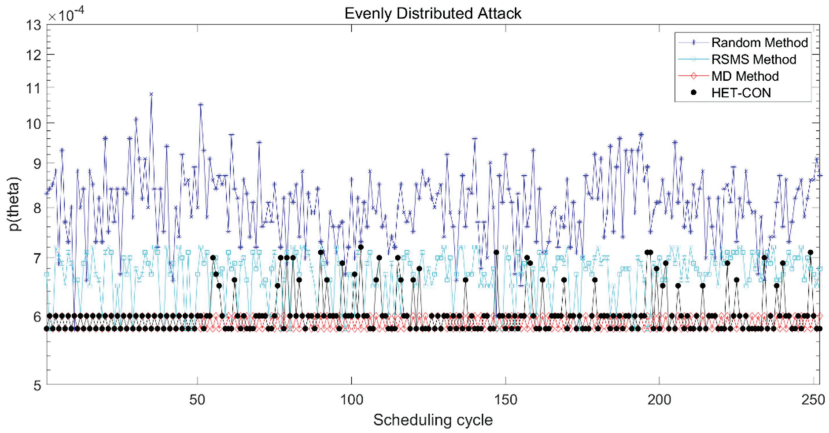


Figure 5 Statistics of the attack success rate in uniform attack scenario.

will select the highest heterogeneity from executor pool except the current scheme. The security of HET-CON algorithm is almost the same as MD algorithm in initial stage, but, as time goes on, confidence becomes another major factor in scheduling. Then there will be lower security than the MD algorithm in the subsequent scheduling. For the RSMS algorithm, due to the use of the random seed method, the selection of heterogeneity cannot reach the global maximum. Its security therefore is higher than Random algorithm, but lower than HET-CON algorithm and MD algorithm.

In the non-uniform distributed attack scenario, attackers are likely to attack a system by continuous attacks. The simulation results are shown in Figure 6. MD algorithm is relatively safe in the initial stage, since the attackers master the historical information and launch targeted attacks, the security of MD algorithm gradually decreases. Due to the introduction of random in Random algorithm and RSMS algorithm, the security of both algorithms fluctuates in a certain range. Similar to a uniform attack scenario, by using a random seed method to select a local scheduling scheme with a greater heterogeneity, the overall security of RSMS is higher than the Random algorithm. For the HET-CON algorithm, the initial performance is similar to the uniform distribution attack, and the security is similar to MD, which has high security. However, as the number of scheduling increases, confidence becomes a factor of influencing scheduling, the security of this algorithm also fluctuates within a certain range. However, by using sliding window confidence and heterogeneity as the benchmark for dispatching scheduling strategies, the floating range of HET-CON is smaller than RSMS.

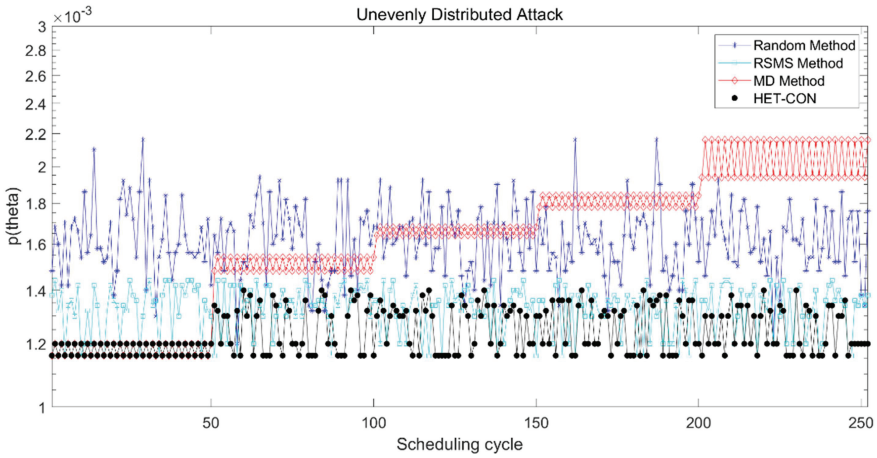


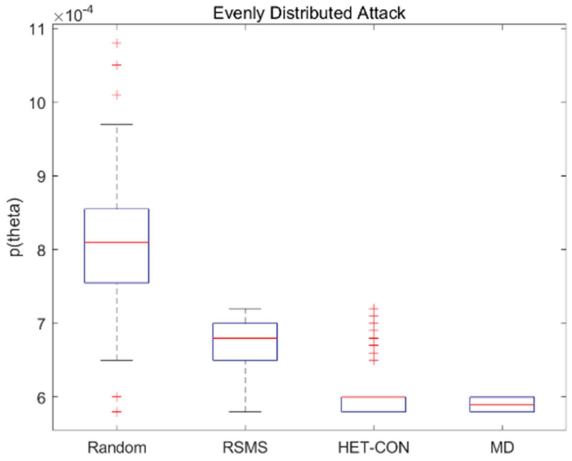
Figure 6 Statistics of the attack success rate in non-uniform attack scenario.

The boxplots of the attack success rate of each algorithm in each attack scenario are shown in Figure 7. From Figure 7(a), we can see that the attack success rate for the Random algorithm is relatively scattered, and the most concentrated is MD algorithm in the uniform distributed attack scenario. As can be seen from Figure 7(b), in the non-uniform distributed attack scenario, the MD algorithm rate of successful attack is more dispersed than that in the uniform distributed scenario. The reason is that the probability of executors with higher heterogeneity being attacked is greatly affected by the type of attack when the attack types are clustered.

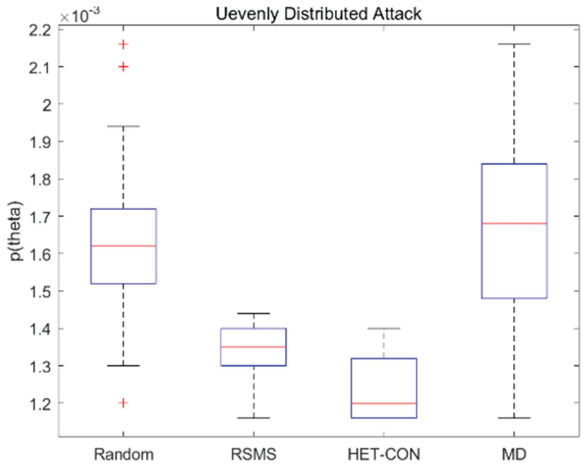
The reason for small fluctuation of HET-CON algorithm in the two scenarios is that the historical performance of executor is considered by introducing the sliding window confidence during scheduling. The impact of the sliding window confidence on the attack success rate at a specific time is greater than the heterogeneity. Compared with the RSMS algorithm, the attack success rate of HET-CON algorithm in non-uniform scenario can be reduced by about 5% and 30% lower than the Random algorithm and the MD algorithm.

2. System operating efficiency analysis

In this section, we mainly compare and analyze the system operating efficiency of the HET-CON algorithm and the other three algorithms. These algorithms are simulated in two scenarios of uniformly distributed attacks and non-uniformly distributed attacks. The simulation results are shown in Figure 8.



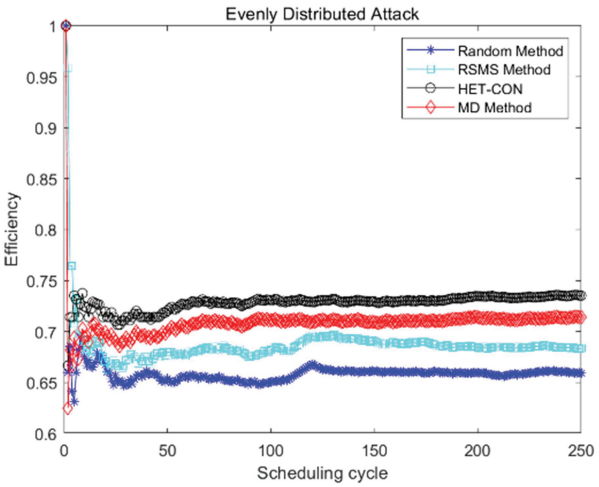
(a) p($\boldsymbol{\Theta}$) in unevenly distributed attack scenario



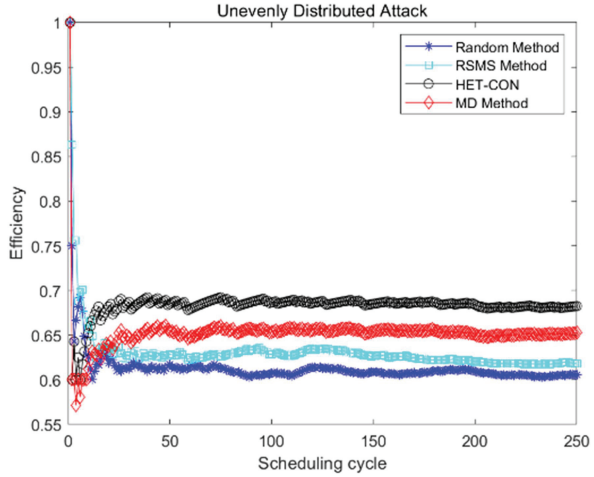
(b) p($\boldsymbol{\Theta}$) in unevenly distributed attack scenario

Figure 7 The attack success rate of each algorithm in each attack scenario.

At the beginning of scheduling, the operating efficiency difference of executors is not obvious. As the number of scheduling increases, the operating efficiency difference gradually becomes clear. Because of the certain randomness of Random and RSMS algorithms, the frequency of call in and call out and the number of executors is more than HET-CON and MD algorithms during scheduling, the system operating efficiency is low. Compared



(a) Efficiency in evenly distributed attack scenario



(b) Efficiency in unevenly distributed attack scenario

Figure 8 Operating efficiency of each algorithm system in two attack scenarios.

to the MD algorithm with heterogeneity to schedule, its scheduling frequency and the number of executors of the HET-CON algorithm is relatively small in the scheduling process. Since non-uniform attacks are more targeted, the operating efficiency of non-uniform attack scenario is slightly lower than that of uniform attack scenario.

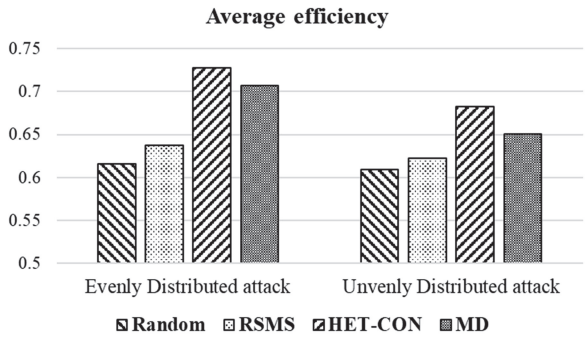


Figure 9 Comparison of operating efficiency of each algorithm.

Table 2 Comparison of effects of each algorithm

Ranking	Security 1 (uniform attribution)	Security 2 (non-uniform attribution)	System operating efficiency
1	MD	**HET-CON**	**HET-CON**
2	**HET-CON**	REMS	MD
3	REMS	Random	REMS
4	Random	MD	Random

In both scenarios, the system operating efficiency rate of HET-CON algorithm is 5% higher than that of the MD algorithm, 11% higher than the RSMS algorithm, and 15% higher than Random. The system operating efficiency statistics are shown in Figure 9.

4.3 Discussion

In the previous section, we have done experiments and analyzed the results on proposed and three compared methods, respectively in evenly and unevenly scenarios. In this section, we summarize the effect of each algorithm in Table 2.

In general, the security of HET-CON algorithm is slightly lower than MD algorithm in uniform distributed attack scenario, but higher than the other two algorithms. In non-uniform distributed attack scenario, HET-CON algorithm has the highest security; meanwhile the HET-CON algorithm is the highest in terms of operating efficiency. Therefore, compared with other algorithms, the HET-CON algorithm has a better performance on the balance of security and operating efficiency.

5 Conclusion

By introducing dynamic, heterogeneous, redundant and negative feedback characteristics, mimic defense technology increases the dynamic changes of internal structure in a targeted manner, thereby enhancing the security of the system. The executor scheduling algorithm is the core of the mimic defense. However, there are rare researches considering both system architecture and attack attributes to evaluate the security and efficiency of mimic system in scheduling algorithm. Based on the attack distribution characteristics and system vulnerability distribution, we constructed a scheduling algorithm, HET-CON. Firstly, mimic system model, DHR, is introduced, the threat model is constructed, and evaluation metrics, security metric and operating efficiency metric, are defined. Secondly, based on the DHR, threat model and evaluation metrics, HET-CON algorithm is proposed. In this method, we redefined heterogeneity with high-order symbiosis of vulnerabilities, and offers the calculation methods. Meanwhile, sliding window confidence is defined to reflect the historical attack behaviors in a certain time. The combination of both metrics offers better performance to scheduling. Thirdly, simulation environment is set up and the results prove that HET-CON achieving better defense effects and system operating efficiency in different scenarios for mimic system, compared with three other scheduling algorithms.

In the future, the algorithm will be implemented on the SDN control plane constructed with mimic defense. However,more problems will be emerged in implementation and reasonableness. In order to solve these problems, the effectiveness of the algorithm and process should be studied, improving the update calculation for scheduling control parameters, the setting for sliding window values, and other related values.

Acknowledgements

The research has been partially supported by the National Core Electronic Devices, High-end Generic Chips and Basic Software Major Projects (No. 2017ZX01030301), The National Natural Science Foundation of China (No. 61572520), Major Project for Committee on Economy and Informatization of Shanghai (No. 201701046).

References

[1] E. Cole, Network security bible, John Wiley & Sons, 2011.
[2] J. H. Cho, et al., Toward Proactive, Adaptive Defense: A Survey on Moving Target Defense, Ieee Communications Surveys and Tutorials, 22(1): 709–745, 2020.
[3] J. X. Wu, Intention and vision of mimic computing and mimic security defense, Telecommunications Science, 30(7): 2–7, 2014.
[4] J. J. Zheng, and A. S. Namin, A Survey on the Moving Target Defense Strategies: An Architectural Perspective, Journal of Computer Science and Technology, 34(1): 207–233, 2019.
[5] J. X. Wu, Research on mimic defense in cyberspace. Journal of information security, (4): 1–10, 2016.
[6] C. Qi, et al., An aware-scheduling security architecture with priority-equal multi-controller for SDN, China Communications, 14(9): 144–154, 2017.
[7] H. C. Hu, et al., MNOS: a mimic network operating system for software defined networks, IET Information Security, 11(6): 345–355, 2017.
[8] Z. Zhang, B. L. Ma, and J. X. Wu, Test and analysis of web server mimic defense principle verification system, Journal of information security, 2(01): 13–28, 2017.
[9] L. M. Pu, et al., Heterogeneous executor scheduling algorithm for mimic cloud service, Journal on Communications, 41(03): 17–24, 2020.
[10] K. Song, et al., Endogenous security architecture of Ethernet switch based on mimic defense, Journal on Communications, 41(5): 18–26, 2020.
[11] W. J. Zhang, et al., A programmable semantic parsing method for mimic judgment, Journal on Communications, 41(4): 62–69, 2020.
[12] H. C. Hu, et al., Mimic defense: a designed-in cybersecurity defense framework, IET Information Security, 12(3): 226–237, 2017.
[13] W. Guo, et al., Scheduling Sequence Control Method Based on Sliding Window in Cyberspace Mimic Defense, IEEE Access, 8: 1517–1533, 2019.
[14] C. Qi, et al. Dynamic-scheduling mechanism of controllers based on security policy in software-defined network, Electronics letters, 52(23): 1918–1920, 2016.
[15] C. H. Li, et al., Mimic defense method of service deployment in SDN. Journal on Communications, 39(S2): 121–130, 2018.

[16] Q. R. Liu, S. J. Lin, and Z. Y. Gu, Heterogeneous functional equivalence scheduling algorithm for mimic security defense, Journal on Communications, 39(07): 188–198, 2018.
[17] J. X. Zhang, J. M. Pang, Z. Zhang. A mimic structured method to quantify the heterogeneity of web servers, Journal on Communications, 31(02): 564–577, 2020.
[18] Z. Q. Wu, et al., A mimic ruling optimization method based on executive heterogeneity, Computer Engineering: 1–8, 2019.
[19] M. Behzadian, et al., A state-of the-art survey of TOPSIS applications, Expert Systems with applications, 39(17): 13051–13069, 2012.
[20] M. Garcia, et al., OS diversity for intrusion tolerance: Myth or reality?, IEEE/IFIP 41st International Conference on Dependable Systems & Networks (DSN), Hong Kong: 383–394, 2011.
[21] M. Garcia, et al., Analysis of operating system diversity for intrusion tolerance, Software-Practice & Experience, 44(6): 735–770, 2014.

Biographies

Wenjian Zhang is a Research Associate of the Information Engineering University, Zhengzhou, China. He received the B.S. and M.S. degrees in Electronic and Information Engineering System from Information Engineering University, Zhengzhou, Henan, China, in 2010 and 2013, respectively, where he is currently pursuing the Ph.D. degree in system from Information Engineering University. His research focuses on information security, network programmable design, integrated circuit design.

Shuai Wei is a Research Associate of the Information Engineering University, Zhengzhou, China. He received the B.S. degree in computer science and technology, the M.S. degree in Software Engineering, and the Ph.D. degree in High performance computing and Parallel Compiling in 2005, 2008, and 2012, respectively. His research focuses on high performance computing, cyber security, and machine learning.

Le Tian is a Research Associate of the Information Engineering University, Zhengzhou, China. He received the B.S. and M.S. degrees in electronic and information engineering system from Information Engineering University, Zhengzhou, Henan, China, in 2010 and 2013, respectively. And Le Tian received his Ph.D. degree in Computer Science from the University of Antwerp in the Internet technology and Data science Lab (IDLab), Belgium, in 2019. His research focuses on IEEE 802.11 WLANs, Internet of Things, network protocols and network architectures.

Ke Song is a Associate researcher of the Information Engineering University, Zhengzhou, China. He received the B.S. in Automation from Hefei University of Technology, Hefei, Anhui, China, in 1999. And he received the M.S. degrees in Control Science and Engineering from Xi'an Jiaotong University, Xi'an, Shanxi, China, in 2004. And Ke Song received his Ph.D. degree in Computer Science and Technology from Fifty-sixth Institute, Wuxi, Jiangsu, China, in 2020. His research focuses on Integrated circuit design technology and information security VLSI/SoC, Network architecture and network security technology.

Zhengbin Zhu is a Ph.D. student at the University of Information Engineering University since spring 2019. He attended the Wuhan University, majoring in mathematics where he received his B.Sc. in Information and Computing Science in 2015. Zhu is now mainly focusing on Cyberspace Mimic Defense.