面向拟态安全防御的异构功能等价体调度算法

刘勤让，林森杰，顾泽宇
（国家数字交换系统工程技术研究中心，河南 郑州 450001）

摘 要：拟态安全防御的一个关键环节是异构功能等价体的调度，现有的调度策略缺乏对冗余体间相似度的考虑，且调度算法较为单一。基于此，提出了一种兼顾动态性和可靠性的异构功能等价体调度算法——随机种子最小相似度算法，首先随机确定种子冗余体，然后再根据相似度指标选择整体相似度最小的最终调度方案。理论分析和仿真实验表明，该算法的调度周期远高于最长相异性距离算法，而失效率远低于随机调度算法，在动态性和可靠性之间达到了较好的平衡。

关键词：拟态安全防御；异构冗余调度；相似度；随机种子
中图分类号：TP309
文献标识码：A
doi: 10.11959/j.issn.1000-436x.2018124

Heterogeneous redundancies scheduling algorithm for mimic security defense

LIU Qinrang, LIN Senjie, GU Zeyu
National Digital Switching System Engineering and Technological Research Center, Zhengzhou 450001, China

Abstract: The scheduling of heterogeneous redundancies is one of the key lines of mimic security defense, but the existing scheduling strategies are lack of consideration about the similarity among redundancies and the scheduling algorithms are incomprehensive. A new scheduling algorithm called random seed & minimum similarity (RSMS) algorithm was proposed, which combined dynamics and reliability by determining a scheduling scheme with minimum global-similarity after choosing a seed-redundancy randomly. Theoretical analysis and simulation results show that RSMS algorithm possessed a far longer scheduling cycle than maximum dissimilarity algorithm, as well as a far lower failure rate than random scheduling algorithm, which represents an effective balance between dynamics and reliability.

Key words: mimic security defense, heterogeneous redundancies scheduling, similarity, random seed

1 引言

随着网络设备和服务的不断发展和普及，人们对网络空间的依赖性越来越强，而网络安全的重要性也越发凸显。传统的网络空间领域中，完成特定服务功能的设备和装置（包括软硬件）对外表征的属性是静态的、确定的，且与其内在结构之间存在强相关的对应关系，攻击者通过对其表征内容的收集与分析，可以在一定程度上掌握有关设备和装置内部的具体信息，并可能发现可利用的漏洞或缺陷。另一方面，攻防双方的不对称性导致防御方十分被动，且需要为防御未知的、不确定的攻击行为付出高昂的代价。

拟态防御作为一种转变网络安全格局的新思想，通过构建动态异构冗余的系统架构和运行机制实现针对特定系统漏洞或后门的入侵容忍[1-3]。在拟态防御系统中，冗余控制器接收外部控制参数，生成冗余调度策略和结果仲裁策略，分别发送给输入

收稿日期：2017-12-25；修回日期：2018-03-29
基金项目：国家自然科学基金资助项目（No.61572520, No.61521003）；上海市科研计划基金资助项目（No.14DZ1104800）
Foundation Items: The National Natural Science Foundation of China (No.61572520, No.61521003), Shanghai Research Project (No.14DZ1104800)

代理器和输出代理器，输入代理器依据接收到的冗余调度策略选择相应的异构功能等价体响应外部服务请求，异构功能等价体将结果发送至输出代理器，输出代理器根据冗余控制器生成的结果仲裁策略对各个结果进行判决，最终选择一路作为响应输出。拟态防御系统架构如图 1 所示。

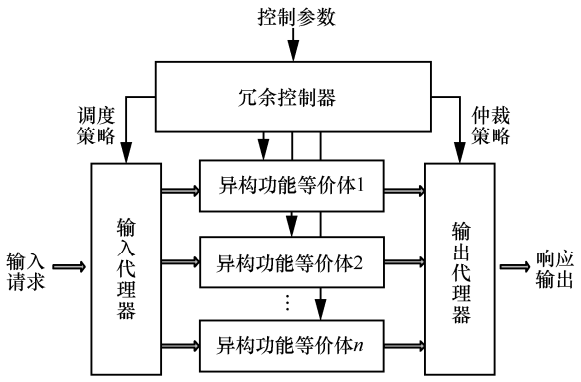


图 1　拟态防御系统架构

传统的容错调度本质上还是为确定的处理机调度任务[4-7]，而拟态调度要求对任务调度不确定的冗余体。现有的多模冗余调度主要从可靠性的角度考虑任务执行体[8-10]。文献[11]提出了一种随机调度策略，采用完全随机的调度策略降低攻击者对特定系统漏洞或缺陷的利用，冗余控制器根据外部控制参数随机确定异构功能等价体的数量和编号，输入代理器根据生成的调度策略分配异构功能等价体响应外部服务请求。文献[12]分别从可信度和性能权重 2 个方面考虑冗余体的调度问题，通过构建异构冗余池中功能等价体的可信度属性值和性能权重属性值，对权值越高的冗余体赋予更高的调度可能性，但没有考虑 2 种方案的综合性能。这些调度策略从不同的角度考量异构体受到调度的合理性，但不够全面。

异构功能等价体作为系统响应外部服务请求的功能主体，其关键特征在于“异构”，“异构”是功能等价体规避攻击者嗅探和利用系统漏洞缺陷的基础。现有的冗余体调度策略将异构冗余池内的等价体认为是相互独立的个体，没有考虑等价体之间的相似程度。实际上，由于功能的等价，各异构体之间也不可能做到完全相异，且相互之间的差异也有大小之分，故异构度小的几个等价体之间存在漏洞或缺陷交集的可能性较大。由于现有的拟态判决策略以多数一致性判决为主，若产生的调度策略选择了相互之间异构度较小的功能等价体集，则攻击者可以利用这些可能存在的漏洞缺陷交集进行攻击，得到多数一致的错误结果，进而通过仲裁得到错误输出。文献[13]提出了 2 种相异性组件选择算法，分别是最长相异性距离（MD, maximum dissimilarity）算法、最佳平均相异距离（OMD, optimal mean dissimilarity）算法。其中，MD 算法选择相异距离最大的组件构成相异性系统，算法得到的是一个确定的全局最优解，对于异构组件较多的系统，算法复杂度较大，且在应用中相当于构成一个静态的异构冗余系统，缺乏动态性，并不适合拟态系统；OMD 算法选取相异距离近似的组件构成相异性系统，避免了局部最大值对组件选择的影响，但并不能保证得到异构度最大的组件集合。

本文首先提出通用拟态安全防御的异构功能等价体调度模型，该模型充分考虑了执行体之间相似度对系统性能的影响，同时定义了相似度指标，将调度方案的相似程度进行量化，然后在此基础上提出了一种兼顾动态性和可靠性的调度算法——随机种子最小相似度（RSMS, random seed & minimum similarity）算法，最后通过理论分析和仿真实验对该算法和现有算法的调度周期和失效率进行比较，观察算法的综合性能。

2　异构功能等价体调度模型

考虑一个通用拟态防御系统的异构冗余池，符号表示如表 1 所示，并由以下定义进行形式化描述。

表 1　　**符号表示**

符号	定义
Ω	异构功能等价体集合，即冗余池
N	冗余池内异构功能等价体数
P_i	异构功能等价体 i，$1<i<N$
R	调度规则集合
F	调度方案集合
$\boldsymbol{E}_z$	调度方案 z 的调度向量
$\boldsymbol{L}_m^i$	组件数为 m 的功能等价体 P_i 的特征向量
$\boldsymbol{Y}_k$	第 k 个组件的特征相似矩阵
h_{ij}	P_i、P_j 之间的相似度，$1\leqslant i,j\leqslant N, i\neq j$
S	调度方案的整体相似度
$h_{\lim}$	相似度阈值

定义 1　异构冗余池可描述为一组异构功能等价体（以下称为冗余体）集合，即 $\Omega=\{P_1,P_2,\cdots,P_N\}$，

且各个冗余体之间相互独立。

定义 2 给定调度规则 $R=\{rul_1, rul_2, \cdots, rul_w\}$，满足规则的调度方案集合 $F|_R=\{f_1, f_2, \cdots, f_n\}$，调度方案 f_z 为 Ω 的子集，各调度方案的调度向量 $\boldsymbol{E}_z=[\eta_{1z} \quad \eta_{2z} \quad \cdots \quad \eta_{Nz}]$，$1 \leqslant z \leqslant n$ 表示各冗余体的调度情况，η_{iz} 的计算式为

$$\eta_{iz}=\begin{cases}1, P_i\text{被调度}\\ 0, P_i\text{不被调度}\end{cases} \quad i=1,2,\cdots,N \tag{1}$$

当任务到达时，控制参数输入冗余控制器并生成满足调度规则的调度向量集合，并根据设定指标从调度向量集合中选择最优调度向量 $\boldsymbol{E}_B$，决定最终执行任务的冗余体。调度向量包含 2 个方面的信息，一是调度方案的余度，即参与任务执行的冗余体数目；二是受到调度的冗余体个体信息。

定义 3 每个冗余体 P_i 可按固定的标准分为若干组件，每个组件又分为不同类目，特征向量集合 $\boldsymbol{L}_m=\{\boldsymbol{L}_1^i, \boldsymbol{L}_2^i, \cdots, \boldsymbol{L}_m^i\}, i=1,2,\cdots,N$。$\boldsymbol{L}_m^i$ 包含 P_i 的特征信息，m 表示组件数。

特征向量 $\boldsymbol{L}_k^i$ 代表冗余体 P_i 中第 k 个组件对应的类目，例如，考虑一个拟态处理机系统，将冗余体按功能模块分为处理器、操作系统、应用软件、协议栈等组件，其中处理器组件包括 CPU、GPU、FPGA、ARM 等类目，若某冗余体的处理器类目为 CPU，则其处理器特征向量为[1 0 0 … 0]。

对于一个异构冗余池，其组件成分可根据具体拟态边界和功能模块划分，考虑到成本和功耗因素，组件类目数量总是有限的。

虽然对组件的类目进行了划分，但同一组件的各类目并不是完全孤立的，实际上，一个组件代表一个功能模块，由于功能的等价，各类目之间总存在一定的相似程度，即可能存在相近甚至相同的漏洞和缺陷为攻击者所用，故冗余体的调度需要考虑组件类目之间的相似程度。

定义 4 冗余池中具有 s 个类目的第 k 个组件的特征相似矩阵为

$$\boldsymbol{Y}_k=\begin{bmatrix}1 & \varphi_{12}^k & \cdots & \varphi_{1s}^k\\ \varphi_{21}^k & 1 & & \varphi_{2s}^k\\ \vdots & & \ddots & \vdots\\ \varphi_{s1}^k & \varphi_{s2}^k & \cdots & 1\end{bmatrix}$$

易知 $\boldsymbol{Y}_k$ 为对称矩阵，其中，$\varphi_{pq}^k\,(0<p\neq q\leqslant s)$ 表示第 k 个组件的第 p、q 个类目的相似度且 $0 \leqslant \varphi_{pq}^k < 1$，特征相似度矩阵中具体的类目相似度由经验给出。$\varphi_{pq}^k$ 越小，p、q 之间的异构度越大；若 $\varphi_{pq}^k=0$，说明 p 和 q 完全异构，达到理想情况。

3 相似度指标

现有的冗余体调度方案常以调度个体的可靠性作为主要的调度指标，但系统的可靠性并不是个体可靠性的简单叠加。考虑一个冗余系统 A，每个子系统具有较高的可靠性，但各子系统十分相似，漏洞相近，针对整个系统的攻击成本与单个子系统的攻击成本相差无几；而另一个冗余系统 B，虽然每个子系统可靠性较 A 中的子系统低，但子系统间异构度较大，难以找到相似的漏洞，由于存在仲裁机制，攻击难度呈几何倍数增加。故从安全性的角度看，冗余体之间的异构程度是评价冗余体调度方案的重要指标。文献[14]对计算机系统的异构性进行了描述，但“异构”是发散的，与某个冗余体相异有多种形式，而“同构”是收敛的，故本文采用“相似度”作为衡量冗余体间差异的指标。相似度指标定义如下。

定义 5 r 余度冗余体集合 Ω^r 的相似度为集合中所有不同元素的相似度之和的归一化，即

$$S|_{\Omega^r}=\frac{1}{\mathrm{C}_r^2}\sum_{i=1}^{r-1}\sum_{j=i+1}^{r}h_{ij} \tag{2}$$

其中，h_{ij} 为 Ω^r 中的冗余体 P_i 和 P_j 的相似度。当然，$S|_{\Omega^r}$ 的大小只有在相同余度的冗余体集合之间才有比较的意义，故比较相似度首先要确定执行体集合的余度。

定义 6 冗余体间的相似度等于各组件之间相似度的加权和，即

$$h_{ij}=\sum_{i=1}^{m}\left(\xi_l\, h_{ij}|_l\right) \tag{3}$$

其中，$h_{ij}|_l$ 为 P_i、P_j 之间第 l 个组件的相似度，m 为组件数，ξ_l 为相似度权重，$\sum_{l=1}^{m}\xi_l=1$，代表组件在冗余体中的重要程度，也可理解为攻击者利用该组件缺陷进行攻击的可能性。

定义 7 组件相似度由对应的特征向量和特征相似矩阵点乘得到，即

$$h_{ij}\big|_l = \boldsymbol{L}_l^i \boldsymbol{Y}_l \boldsymbol{L}_l^{j\mathrm{T}} \tag{4}$$

可以得到 r 余度冗余体集合 $\boldsymbol{\Omega}^r$ 的相似度为

$$S\big|_{\Omega^r} = \frac{1}{\mathrm{C}_r^2}\sum_{i=1}^{r-1}\sum_{j=i+1}^{r}\sum_{l=1}^{m}\left(\xi_l \boldsymbol{L}_l^i \boldsymbol{Y}_l \boldsymbol{L}_l^{j\mathrm{T}}\right) \tag{5}$$

需要注意的是，这里的异构冗余体标号是指在冗余体集合中的序号，与第 2 节冗余池中的冗余体标号不存在对应关系。由于调度的冗余体集合是冗余池的一个子集，由调度向量 $\boldsymbol{E}_z$ 决定，故相似度也可以表示为

$$S\big|_{\Omega^r} = \frac{1}{\mathrm{C}_r^2}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\sum_{l=1}^{m}\left[\xi_l\left(\eta_{is}\boldsymbol{L}_l^i\right)\boldsymbol{Y}_l\left(\eta_{js}\boldsymbol{L}_l^{j\mathrm{T}}\right)\right] \tag{6}$$

4　随机种子最小相似度算法

拟态调度的动态性导入可以有多种形式，例如，对一个确定的有限冗余池枚举出所有可能的冗余体集合，为达到动态性和可靠性的折中，选择其中相似度最小的若干个冗余体集合作为调度方案集合，再根据既定策略（如轮转、彩票、随机等）进行调度。这种方法较为直观且易于实现，但不适用于不确定的、动态变化的冗余池，由于拟态防御存在负反馈机制，若异构冗余体因故障下线或强制清洗，调度方案集合可能受到影响，严重时甚至无法提供正常服务。

本文提出一种随机种子最小相似度算法，算法原理如下。首先，在正常工作的异构冗余体中随机确定任务执行余度和一个种子冗余体，为拟态调度引入动态性（种子冗余体包含于调度方案中），然后根据最小相似度原则选择整体相似度最小的调度方案。

式(5)结合调度向量、特征向量和特征相似矩阵，给出了具体调度方案的相似度指标，直观上考虑，理想的调度方案是选择符合余度条件且使整体相似度最小的若干冗余体作为任务执行体，但仍存在一个问题：整体相似度最小的冗余体集合可能并不是最合适的调度方案，以下举例说明这种情况。

考虑一个 3 余度拟态防御系统，以 3 个点代表冗余体，3 个边长代表各冗余体之间的异构度（异构度与相似度成反比关系），如图 2 所示。其中 $\{P_1,P_2,P_3\}$、$\{P_1,P_2,P_4\}$ 表示 2 种调度方案，$A+B_1+C_1 > A+B_2+C_2$，方案 1 整体相似度小于方案 2，但实际上由于 B_1 很小，即 P_1 和 P_2 比较接近，攻击者对 P_2 和 P_3 的相似漏洞进行利用并实现成功攻击的概率相对较高。因此，相似度指标不仅要考虑整体大小，还要防止出现局部极值的情况。

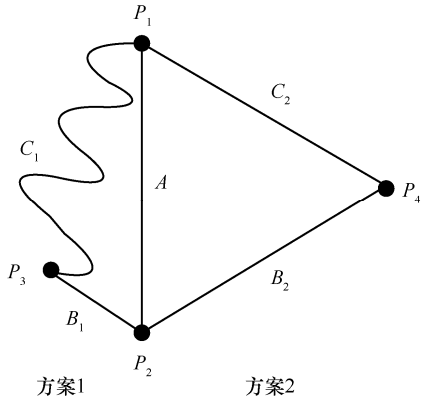


图 2　2 种调度方案示意

定义 8　调度方案的任意 2 个冗余体之间的相似度要小于设定的阈值，否则方案不成立。即在方案 Ω^r 中，$\forall i,j\in[1,r], i\neq j$，有 $S_{ij} < S_{\lim}$，以保证调度方案符合基本的相似度要求，$S_{\lim}$ 为冗余体间相似度阈值。

可以看出，最小相似度算法体现了 4 个调度规则，即 $R=\{rul_1,rul_2,rul_3,rul_4\}$

rul_1：调度方案余度为 r。

rul_2：调度方案包含种子冗余体 P_s。

rul_3：调度方案中的冗余体之间的相似度必须小于阈值，即 $S_{ij} < S_{\lim}$。

rul_4：调度方案的整体相似度最小。

算法的具体流程如下。

1) 随机确定执行余度和种子冗余体

冗余控制器根据控制参数随机生成 2 个参数 r、P_s，其中，r 代表执行余度，P_s 代表种子冗余体。种子冗余体作为调度方案首先确定的元素，可以避免调度方案的单一性。若单纯依据相似度最小原则，生成的调度方案即使不唯一也十分有限，不符合拟态防御系统的动态特性。同时，确定种子冗余体相当于减少了调度方案余度，即减少了计算复杂度。

2) 排除与种子冗余体相似度超过阈值的冗余体

将冗余池中与种子冗余体 P_s 相似度超过阈值的冗余体排除，即对冗余池进行“初步过滤”。对冗余体进行初步的筛选，较小的冗余池可以有效降低初步调度方案的数量，减少计算量。若过滤后的

冗余池内的冗余体数量超过或等于余度要求，则转到第 3)步；若低于余度要求，则转到第 1)步重新开始。

3) 生成初步调度方案集合

根据过滤后冗余池内的元素，生成满足余度要求且包含种子冗余体的所有调度方案，排除冗余体间相似度超过阈值的方案，得到初步调度方案集合 $F\big|_R=\{f_1,f_2,\cdots,f_n\}$。若初步调度方案集合元素多于一个，则转到第 4)步；若为∅，则转到第 1)步重新开始；若等于一个，则其为最终调度方案。

4) 确定最终调度方案

计算初步调度方案集合中各元素的整体相似度，取其中整体相似度最小的作为最终调度方案。

算法伪代码如算法 1 所示。

算法 1 RSMS 算法

输入 余度取值范围 *range*；冗余池 Ω；组件特征相似矩阵集合 $\boldsymbol{Y}$；相似度阈值 $S_{\lim}$

输出 最终调度方案 f_d

1) $r\leftarrow \text{random}(range)$ //确定余度

2) $s\leftarrow \text{random}(1{:}N)$ //随机选择种子冗余体

//排除不符合相似度要求的冗余体

3) $P\leftarrow\varnothing$

4) for $i\leftarrow 1,2,\cdots,s-1,s+1,\cdots,N$ do

5) if $S_{is}\leqslant S_{\lim}$ then

6) $P\leftarrow P\cup\{P_i\}$

7) end if

8) end for

9) $M\leftarrow \text{size}(P)$

10) if $M+1<r$ then

11) restart

12) else

//生成初步调度方案集合//

13) $k\leftarrow \mathrm{C}_M^{r-1}$

14) $F\leftarrow\varnothing$

15) for $j\leftarrow 1,2,\cdots,k$ do

16) $f_j\leftarrow\{P_s,\cdots\}$ // f_j 余度为 r 且包含 P_s

17) if $\forall P_p,P_q\in f_j,S_{pq}\leqslant S_{\lim}$ then

18) $F\leftarrow F\cup\{f_j\}$

19) end if

20) end for

//确定最终调度方案

21) if $\text{size}(F)=0$ then

22) restart

23) else if $\text{size}(F)>1$ then

24) if $S_z=\min(S\big|_F)$ then

25) $f_d\leftarrow f_z$

26) end if

27) else

28) $f_d\leftarrow f_1$

29) break

30) end if

31) end if

32) end if

5 算法性能分析

相对于 MD 算法和 OMD 算法，RSMS 算法每次确定的冗余体集合是不确定的，同时具备随机调度算法缺乏的相异性考虑，达到了二者的折中，下面对 RSMS 算法、随机调度算法、MD 算法和 OMD 算法的动态性和可靠性进行理论分析。

5.1 动态性分析

算法的动态性体现在调度方案的平均周期。理想的动态性要求调度方案完全不重复，这在无限余度冗余池中才能实现，事实上，由于余度的限制，调度方案必然存在重复的情况，进而存在一个平均调度周期。而只要冗余池确定，则 MD 算法和 OMD 算法得出的调度方案都是确定的，即周期为 1。以下通过理论推导随机调度算法和RSMS算法的平均调度周期来比较二者的动态性。

几个基本假设如下。

假设 1 随机调度算法中，各个冗余体被调度的概率相同；RSMS 算法确定种子冗余体 P_s 时，各个冗余体被选中的概率相同。

假设 2 由于 RSMS 算法的执行余度是随机确定的，本文假设随机调度算法也随机确定余度。

假设 3 从安全性和多数一致性表决的可靠性考虑，拟态防御系统的余度应大于或等于 3，令调度方案余度范围 $range=(3,4,\cdots,k)$，$3<k<N$。

假设 4 为简化分析，暂不考虑相似度阈值。

对随机调度算法来说，每个冗余体都是无差别的，算法确定的每种可能的调度方案概率都相等，则随机调度算法的平均调度周期 T_{R} 等价于所有调

度方案的数量总和，即

$$T_{\mathrm{R}}=\sum_{r=3}^{k}\mathrm{C}_{N}^{r}=\sum_{r=3}^{k}\frac{N!}{r!(N-r)!} \tag{7}$$

而对于给定的余度，RSMS 算法的平均调度周期决定于选中的种子冗余体和包含它的整体相似度最小的方案。RSMS 算法首先确定种子冗余体 P_s，根据假设 1，种子冗余体的可能性有 N 种，然后确定一个包含 P_s 且整体相似度最小的调度方案，即能够确定的方案一共有 $(k-3)N$ 种，但这里存在一个问题，即在同一余度条件下，可能存在 P_s 不同而最终确定的调度方案相同的情况，这样的重复情况需要排除。通过分析计算调度方案的重复概率，得到目标算法调度周期。

对于给定的余度 r 和种子冗余体 P_s，调度方案的可能情况有 C_{N-1}^{r-1} 种，每一种都是等可能的，而确定最终调度方案 f_d 后，表示 f_d 的整体相似度大于其他包含 P_s 的方案。以 f_d 中除 P_s 外的某个冗余体作为种子冗余体 P_s' 的可能调度方案有也 C_{N-1}^{r-1} 种，而其中包含 P_s' 和 P_s 的调度方案有 C_{N-2}^{r-2} 种，也是等可能的。若发生重复情况，说明以 P_s' 为种子冗余体的调度方案 $f_d'\propto f_d$，即 $S\big|_{f_d}$ 大于其他 $(\mathrm{C}_{N-1}^{r-1}-1)$ 种可能的情况，由于 $S\big|_{f_d}$ 已经大于包含 P_s' 和 P_s 的调度方案的相似度，故根据条件概率公式，$S\big|_{f_d}$ 大于其他包含 P_s' 的调度方案的概率为

$$P_r=\frac{P(S\big|_{f_d}\text{大于其他包含}P_s'\text{的调度方案})}{P(S\big|_{f_d}\text{大于其他包含}P_s'\text{和}P_s\text{的调度方案})}=\frac{\mathrm{C}_{N-2}^{r-2}}{\mathrm{C}_{N-1}^{r-1}} \tag{8}$$

调度方案最多有 $r-1$ 个与 f_d 发生重复，若 f_d 中除 P_s 外有 n 个冗余体作为种子冗余体的调度方案与 f_d 相同，则调度周期为 $\dfrac{N}{n+1}$，其概率为 $\mathrm{C}_{r-1}^{n}(P_r)^n(1-P_r)^{r-n-1}$，故 r 余度 RSMS 算法的平均调度周期为

$$T_{\mathrm{RSMS}}^{r}=\sum_{n=0}^{r-1}\frac{N}{n+1}\mathrm{C}_{r-1}^{n}(P_r)^n(1-P_r)^{r-n-1} \tag{9}$$

故 RSMS 的总体调度周期为

$$T_{\mathrm{RSMS}}=\sum_{r=3}^{k}T_{\mathrm{RSMS}}^{r}=\sum_{r=3}^{k}\sum_{n=0}^{r-1}\frac{N}{n+1}\mathrm{C}_{r-1}^{n}(P_r)^n(1-P_r)^{r-n-1} \tag{10}$$

易知 $T_{\mathrm{R}}>T_{\mathrm{RSMS}}$，即 RSMS 算法的平均调度周期小于随机调度算法周期，而大于 MD 算法和 OMD 算法。

接下来分析调度方案动态性对系统安全性的影响。

假设攻击者实施攻击是基于先验系统信息的，先验信息越多，则成功概率越高，为便于分析，做出如下假设。

假设 5　攻击者在一个任务执行周期中可进行一次探测。

假设 6　若一次探测所得信息与先验信息矛盾，则先验信息失效，反之则先验信息累积。

假设 7　若当前系统信息与先验信息一致，且攻击前等效进行了 ε ($\varepsilon=1,2,3\cdots$)次有效探测，此时攻击成功概率为

$$P_{\varepsilon}=\mathrm{A}(1-\mathrm{e}^{-\varepsilon}),0<A<1 \tag{11}$$

假设 8　不同调度方案的特征信息不相同。

假设 7 中攻击成功率与连续有效探测次数呈负指数关系，不难理解，随着有效探测次数的增加，先验信息不断积累，依赖先验信息的攻击成功概率随之不断提高并趋近于极限值 A。令 $P_0=0$，即没有先验信息的攻击无法成功。

对于动态变化的调度方案，认为先验信息与当前系统信息相互独立，即攻击成功需要调度算法选中与先验信息一致的调度方案，而只有连续探测一致的先验信息才能累积，则基于 ε 次探测后攻击成功说明当前系统信息与最后一次探测一致，而前 $\varepsilon-1$ 次探测并不确定，基于假设 1，则攻击成功概率为

$$P_{\varepsilon}=\sum_{n=0}^{\varepsilon-1}\left(1-\frac{1}{T}\right)\frac{1}{T^n}A(1-\mathrm{e}^{-n})+\frac{1}{T^{\varepsilon}}A(1-\mathrm{e}^{-\varepsilon}),0<A<1 \tag{12}$$

其中，T 为调度方案的平均周期。对于余度为 9 的冗余池，MD、OMD、RSMS 算法下的攻击成功率与探测次数的关系如图 3 所示（令 A=0.6，RSMS 算法的 $k=3$）。从图 3 可以看出，动态性的导入使 RSMS 算法的攻击成功率远低于 MD 和 OMD 算法，且随着 k 的增加，攻击成功率还会更低，系统安全性明显提高。

5.2　可靠性分析

系统能否在偶然或恶意的失效情况下连续、可靠、正常地执行是拟态防御首要考虑的问题。在拟态调度环节，各冗余体的可靠性是系统可靠性的基础，但不同的调度策略决定了系统可靠性的构成。

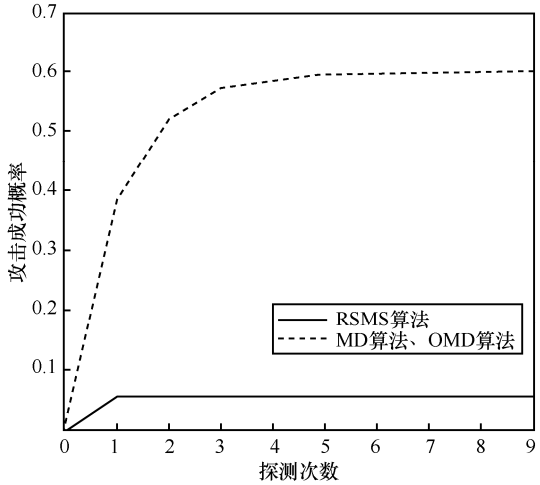


图 3　不同算法下攻击成功概率与探测次数的关系

将调度算法的可靠性定义为系统执行任务后得到正确输出的概率，通过计算随机调度算法、MD 算法、OMD 算法、RSMS 算法下系统正常工作的概率，分析各种算法理论上的可靠性。

为简化分析，只考虑失效发生在异构冗余体内的情况，并做出以下假设。

假设 9　异构体发生失效的情况服从 0-1 分布。异构冗余体调度模型的失效率向量 $\boldsymbol{L}=[\lambda_1 \quad \lambda_2 \quad \cdots \quad \lambda_N]$，$\lambda_i$ 为冗余体 P_i 的失效率，包含随机失效和受到攻击而产生的错误输出。

假设 10　由于系统的整体可靠性还与拟态仲裁环节相关，以最具代表性的多数一致性表决作为仲裁策略，即选择超过半数的一致结果作为系统输出，当所有执行体的结果不能达到多数一致时，则输出异常。

假设 11　相似度的定义是为了体现不同冗余体发生共因失效的情况可能性，这里假设 r 余度调度方案 f_x^r 中的 n 个错误结果相同的概率与其整体相似度有如下关系

$$P_n'\big|_{f_x^r}=A(S\big|_{f_x^r})^n,\ k\leqslant r \tag{13}$$

其中，A 为常系数，同一余度条件下，若方案整体相似度越高，则出现相同错误结果的可能性越大。出现相同错误结果的可能性与相同错误结果数量呈指数关系。

假设 12　输出异常不认为失效，异常后系统重新执行当前任务。

对于调度方案 $f_x^r=\{q_{1x},q_{2x},\cdots,q_{rx}\}$，失效率向量 $\boldsymbol{L}_x=(\lambda_{1x},\lambda_{2x},\cdots,\lambda_{rx})$，发生 n 个错误结果的情况有 C_r^n 种，则至少出现 n 个相同错误结果的概率为

$$P_{\geqslant n}^{\text{wrong}}\Big|_{f_x^r}=\sum_{m=n}^{r}\left\{A(S\big|_{f_x^r})^m\sum_{h=m}^{r}\sum_{i_1=1}^{r-h+1}\sum_{i_2=2}^{r-h+2}\cdots\sum_{i_h=h}^{r}\left[\prod_{j=1}^{h}\lambda_{i_j x}\prod_{z\neq i_1,i_2,\cdots,i_n}(1-\lambda_{zx})\right]\right\} \tag{14}$$

系统出现错误输出说明至少有半数的冗余体失效且错误结果相同，令 $\tau=\left\lceil\dfrac{r+1}{2}\right\rceil$，表示 $\dfrac{r+1}{2}$ 向上取整，则半数以上冗余体失效且错误结果相同的概率为 $P_{\geqslant\tau}^{\text{wrong}}\Big|_{f_x^r}$。

以随机调度算法为例，所有符合余度要求的调度方案都是等可能的，由于执行余度共有 $k-3+1=k-2$ 种，r 余度条件下，调度方案共有 C_N^r 种，故其平均失效率为

$$P_{\text{wR}}=\frac{1}{k-2}\sum_{r=3}^{k}\left(\frac{1}{\mathrm{C}_N^r}\sum_{f_x^r\in\Omega}P_{\geqslant\tau}^{\text{wrong}}\Big|_{f_x^r}\right) \tag{15}$$

进而推导出 4 种算法下系统正常工作的概率如表 2 所示。

表 2　**4 种算法下系统正常工作的概率**

算法	正常工作概率
随机调度	$P_{\text{wR}}=\dfrac{1}{k-2}\sum_{r=3}^{k}\left(\dfrac{1}{\mathrm{C}}\sum_{f_x^r\in\Omega}P_{\geqslant\tau}^{\text{wrong}}\Big\vert_{f_x^r}\right)$
MD	$P_{\text{wL}}=\dfrac{1}{k-2}\sum_{r=3}^{k}P_{\geqslant\tau}^{\text{wrong}}\Big\vert_{\min(S\vert_{f_x^r})}$
OMD	$P_{\text{wB}}=\dfrac{1}{k-2}\sum_{r=3}^{k}P_{\geqslant\tau}^{\text{wrong}}\Big\vert_{\min\left[\sigma(\forall h_{ij}\vert_{h_{ij}\in f_x^r})\right]}$
RSMS	$P_{\text{wRSMS}}=\dfrac{1}{k-2}\sum_{r=3}^{k}\left[\dfrac{1}{N}\sum_{f_x^r\in\Omega\&P_x\in f_x^r}P_{\geqslant\tau}^{\text{wrong}}\Big\vert_{\min(S\vert_{f_x^r})}\right]$

6　仿真实验

本节将 RSMS 算法和随机调度算法、MD 算法和 OMD 算法的性能进行比较，在不同异构体余度和执行余度条件下比较算法的动态性和可靠性，并通过仿真数据对算法总体性能进行分析验证。

6.1　动态性比较

给定模拟冗余池，余度为 9，冗余体间的相似度以参数为(5,15)的 β 分布随机生成[15]，如图 4 所示，得到如下相似度矩阵。

$$\begin{bmatrix}
1 & 0.277 & 0.342 & 0.276 & 0.185 & 0.200 & 0.259 & 0.334 & 0.490 \\
0.277 & 1 & 0.228 & 0.442 & 0.215 & 0.350 & 0.080 & 0.120 & 0.426 \\
0.342 & 0.228 & 1 & 0.433 & 0.212 & 0.224 & 0.047 & 0.301 & 0.436 \\
0.276 & 0.442 & 0.433 & 1 & 0.111 & 0.393 & 0.263 & 0.158 & 0.175 \\
0.185 & 0.215 & 0.212 & 0.111 & 1 & 0.376 & 0.366 & 0.241 & 0.252 \\
0.200 & 0.350 & 0.224 & 0.393 & 0.376 & 1 & 0.401 & 0.258 & 0.289 \\
0.259 & 0.080 & 0.047 & 0.263 & 0.366 & 0.401 & 1 & 0.143 & 0.208 \\
0.334 & 0.120 & 0.301 & 0.158 & 0.241 & 0.258 & 0.143 & 1 & 0.282 \\
0.490 & 0.426 & 0.436 & 0.175 & 0.252 & 0.289 & 0.208 & 0.282 & 1
\end{bmatrix}$$

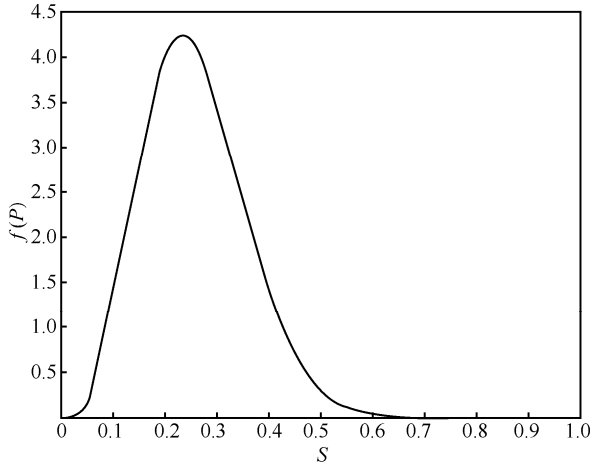


图 4　参数为(5,15)的 β 分布概率密度

文献[16]分析了拟态处理机执行余度与安全增益的关系，结论为：余度为 3 的系统可以达到最佳的折中效果，故本次实验只考虑执行余度范围 *range*=(3,4,5)的情况。根据第 5 节的分析，对于不同的执行余度，4 种算法的平均调度周期、算法确定的调度方案的平均相似度如表 3 和表 4 所示。

表 3　　4 种算法的平均调度周期

算法	r =3	r =4	r =5
RSMS 算法	7.31	5.08	3.49
随机调度算法	84.00	126.00	126.00
MD 算法	1.00	1.00	1.00
OMD 算法	1.00	1.00	1.00

表 4　　算法确定的调度方案的平均相似度

算法	r =3	r =4	r =5
RSMS 算法	0.155 0	0.181 5	0.204 4
随机调度算法	0.272 3	0.272 3	0.272 3
MD 算法	0.114 3	0.153 2	0.195 3
OMD 算法	0.218 3	0.283 0	0.267 3

下面针对RSMS算法和随机调度算法以蒙特卡洛法进行 100 次模拟实验，观察各算法的调度周期。在不断的模拟调度中，若产生与第一次调度结果相同的结果，则一次实验结束，调度周期为总的调度次数减 1，得到实验结果如图 5 所示。

图 5 中虚线为各次实验结果的平均值。从图 5 可以看出，实验结果与理论分析十分吻合，随机调度算法的平均调度周期最大，RSMS 算法次之，而

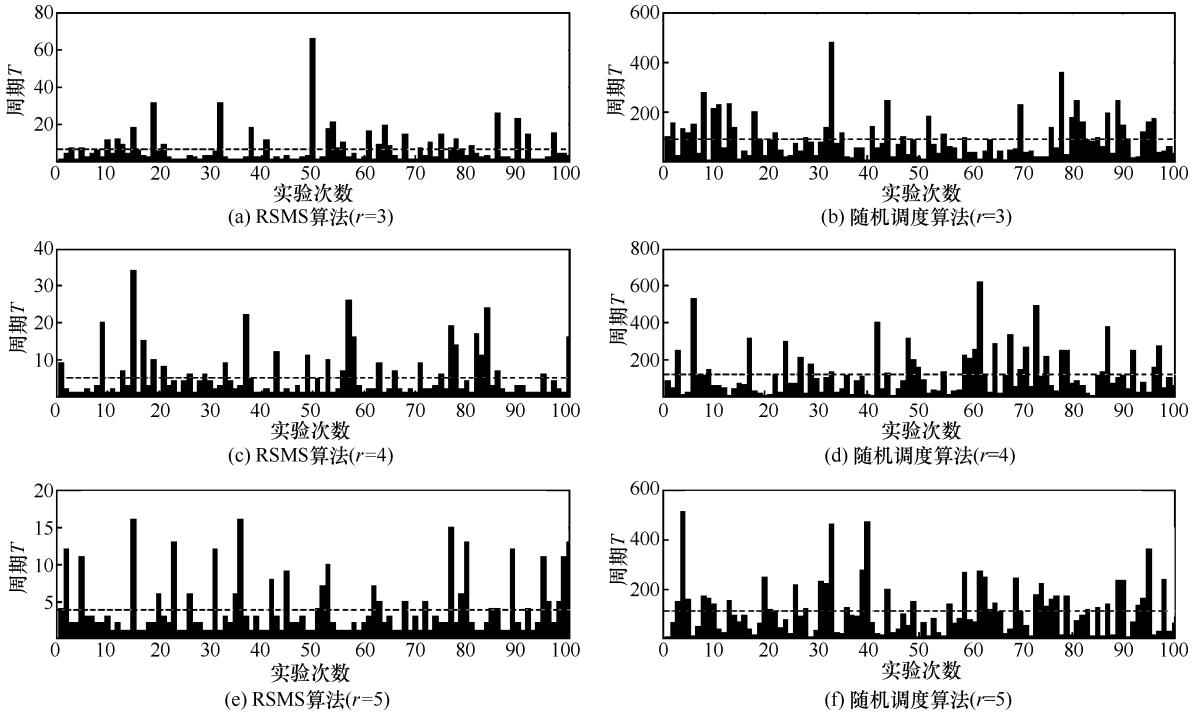


图 5　RSMS 算法和随机调度算法的调度周期仿真结果

MD 算法和 OMD 算法基本没有动态性； RSMS 算法的平均调度周期为 15.884 5，是 MD 算法的 5.3 倍。RSMS 算法虽然在动态性上不如随机调度算法，但在余度为 3、4、5 时，其平均相似度分别降低了 43.08%、33.35%、24.94%。

6.2 可靠性比较

仍考虑上述冗余池，在没有先验知识的条件下，首先假设各异构功能等价体的失效率是相同的，为便于观察，不妨把失效率设置得大一点。令失效率 $\lambda_i = 0.1$，A=1，由于系统正常工作的概率与整体失效率之和为 1，下面，比较 4 种算法的平均失效率，结果如表 5 所示。

表 5 $\lambda_i = 0.1$ 时 4 种算法的平均失效率

算法	$r=3$	$r=4$	$r=5$
RSMS 算法	$7.190\,9\times10^{-4}$	$2.365\,6\times10^{-5}$	$9.210\,3\times10^{-5}$
随机调度算法	0.002 2	$8.022\,7\times10^{-5}$	$1.813\,4\times10^{-4}$
MD 算法	$3.675\,1\times10^{-4}$	$1.335\,0\times10^{-5}$	$6.443\,7\times10^{-5}$
OMD 算法	0.001 3	$8.450\,3\times10^{-5}$	$1.677\,2\times10^{-4}$

由表 5 可以看出，RSMS 算法的调度方案平均失效率在 4 种算法中只略高于 MD 算法，且远比随机调度算法低，其可靠性相对较高。

接下来考虑失效率不相等的情况，令各异构功能等价体的失效率服从(0,0.1)之间的均匀分布，得到失效率向量 $\boldsymbol{L}=[0.081\,5\ \ 0.090\,6\ \ 0.012\,7\ \ 0.091\,3\ \ 0.063\,2\ \ 0.009\,8\ \ 0.027\,8\ \ 0.054\,7\ \ 0.095\,8]$，此时，4 种算法的平均失效率如表 6 和图 6 所示。

表 6 失效率向量为 $\boldsymbol{L}$ 时 4 种算法的平均失效率

算法	$r=3$	$r=4$	$r=5$
RSMS 算法	$2.766\,4\times10^{-4}$	$3.806\,4\times10^{-6}$	$2.574\,5\times10^{-5}$
随机调度算法	$7.664\,7\times10^{-4}$	$1.515\,2\times10^{-5}$	$3.518\,6\times10^{-5}$
MD 算法	$1.141\,9\times10^{-4}$	$8.876\,5\times10^{-7}$	$6.782\,0\times10^{-6}$
OMD 算法	$3.598\,9\times10^{-4}$	$1.058\,3\times10^{-5}$	$1.118\,2\times10^{-5}$

由表 6 可以看出，RSMS 算法的调度方案平均失效率仍然远低于随机调度算法，改变失效率向量后，计算结果仍有相同的规律。

由于MD算法和OMD算法确定的调度方案是唯一的，故对 RSMS 算法和随机调度算法进行 100 次模拟调度实验，计算各次调度结果的失效率，得到的仿真结果如图 7 所示，可见实验结果与理论分析十分吻合。

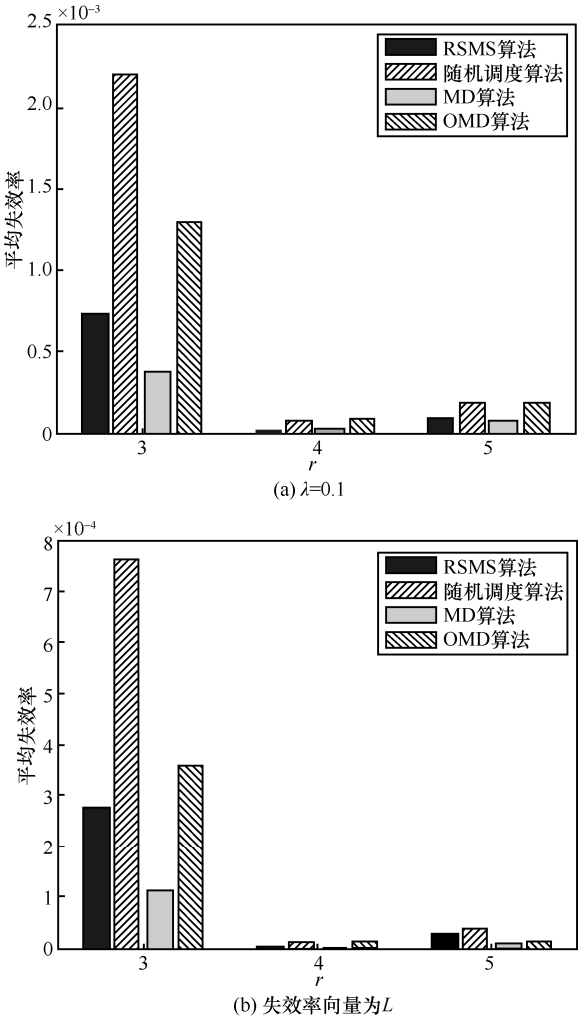


图 6 不同失效率向量下 4 种算法的平均失效率

从表 6 和图 6 可以看出，MD 算法的可靠性最高，RSMS 算法次之，随机调度算法和 OMD 算法最低。其中，当余度为 3、4、5 时，RSMS 算法调度方案的平均失效率比随机调度算法分别低 67.32%、70.51%、49.21%。

通过分析和仿真实验对比 4 种算法的动态性和可靠性这 2 个方面的性能，得到 4 种算法的性能排序，如表 7 所示。

表 7 4 种算法的性能排序

性能	动态性	可靠性
高	随机调度算法	MD 算法
↓	RSMS 算法	RSMS 算法
低	MD 算法/OMD 算法	随机调度算法/OMD 算法

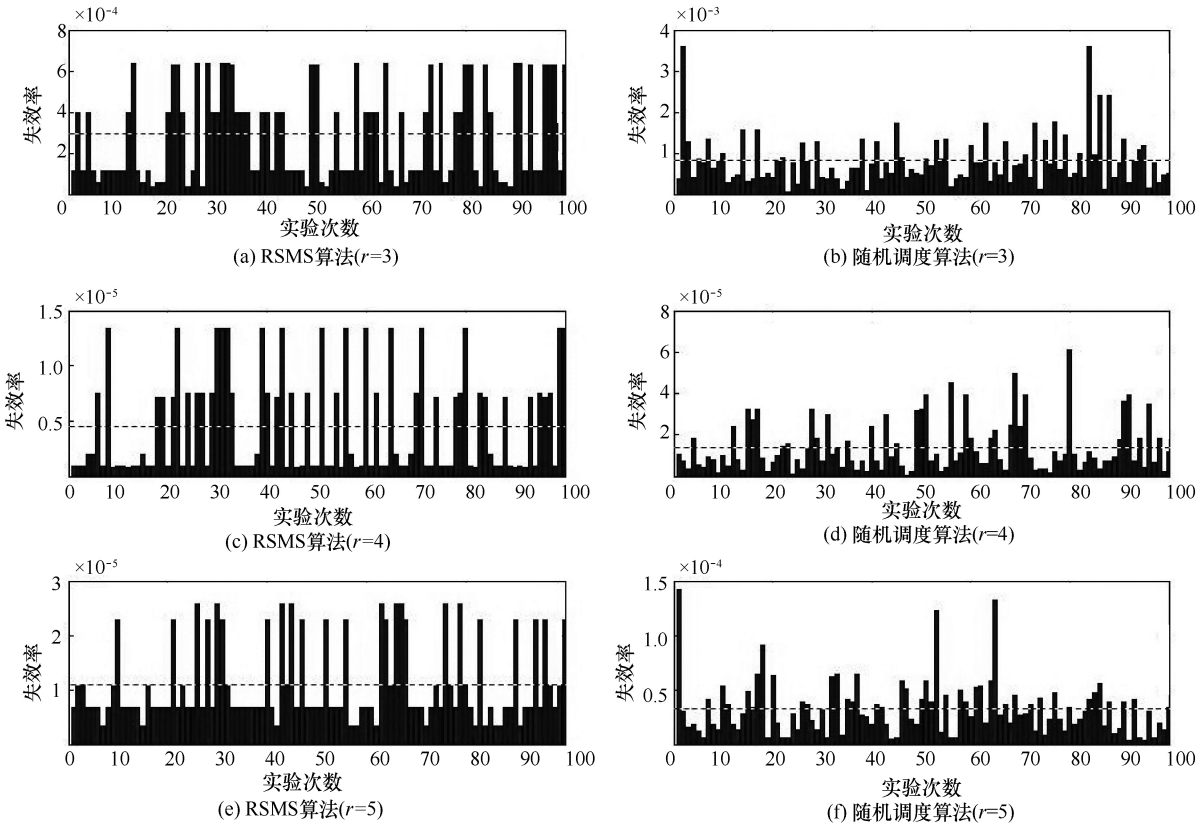


图 7　RSMS 算法和随机调度算法的失效率仿真结果

从图 7 可以看出，随机调度算法和 MD 算法各有侧重，而 RSMS 算法兼顾了两者的优点。RSMS 算法通过对异构体之间相似度的考虑，降低了调度方案因发生共因失效而造成错误输出的概率，同时兼具随机调度的动态性优点，在动态性和可靠性之间达到了较好的平衡。

7　结束语

拟态防御是改变现有网络安全格局的新思想，而异构功能等价体的调度是其中的关键环节，本文针对拟态防御系统的调度环节，提出了异构功能等价体的调度模型以及调度方案的相似度概念和计算式，并针对调度方案的相似度提出了一种拟态调度算法——随机种子最小相似度算法。通过对算法调度周期和失效率的理论分析和仿真实验，证明 RSMS 算法兼顾了动态性和可靠性。冗余体的可靠性有多种定义方式，并且可能在进程中不断变化（例如，引用冗余体的历史记录），而这种情况下 RSMS 算法也可适应。

理论上，RSMS 算法在动态性和可靠性这 2 个方面的权重是可以调整的，这与安全性的定义有关，有待后续研究。

参考文献：

[1] 邬江兴. 网络空间拟态防御研究[J]. 信息安全学报, 2016, 1(4): 1-10.
WU J X. Research on cyber mimic defense[J]. Journal of Cyber Security, 2016, 1(4): 1-10.

[2] 扈红超, 陈福才, 王禎鹏. 拟态防御 DHR 模型若干问题探讨和性能评估[J]. 信息安全学报, 2016, 1(4): 40-51.
HU H C, CHEN F C, WANG Z P. Performance evaluations on DHR for cyberspace mimic defense[J]. Journal of Cyber Security, 2016, 1(4): 1-10.

[3] 仝青, 张铮, 邬江兴. 基于软硬件多样性的主动防御技术[J]. 信息安全学报, 2017, 2(1): 1-12.
TONG Q, ZHANG Z, WU J X. The active defense technology based on the software/hardware diversity[J]. Journal of Cyber Security, 2017, 2 (1):1-12.

[4] REIS G A, CHANG J, VACHHARAJANI N, et al. SWIFT: software implemented fault tolerance[C]//International Symposium on Code Generation and Optimization. 2005:243-254.

[5] WANG J, BAO W, ZHU X, et al. FESTAL: fault-tolerant elastic scheduling algorithm for real-time tasks in virtualized clouds[J]. IEEE Transactions on Computers, 2015, 64(9):2545-2558.

[6] 殷进勇, 顾国昌. 允许多处理机故障的实时任务容错调度算法[J]. 电子与信息学报, 2010, 32(2):444-448.
YIN J Y, GU G C. A real-time fault-tolerant scheduling algorithm for multiple processor faults[J]. Journal of Electronics & Information

Technology, 2010, 32(2):444-448.

[7] 彭浩, 陆阳, 孙峰,等. 副版本不可抢占的全局容错调度算法[J]. 软件学报, 2016, 27(12):3158-3171.
PENG H, LU Y, SUN F, et al. Faulttolerant global scheduling with non-preemptive backups[J]. Journal of Software, 2016, 27(12): 3158-3171.

[8] DAS A, KUMAR A, VEERAVALLI B. Reliability and energy-aware mapping and scheduling of multimedia applications on multiprocessor systems[J]. IEEE Transactions on Parallel & Distributed Systems, 2016, 27(3):869-884.

[9] 吉萌, 余少华, 詹翊春. 双冗余结构路由器故障恢复模型与方案研究[J]. 通信学报, 2006, 27(6):21-28.
JI M, YU S H, ZHAN Y C. Research on fault-recovery model and scheme in dual-system redundancy router[J]. Journal on Communications, 2006, 27(6):21-28.

[10] CHEN C Y. Task scheduling for maximizing performance and reliability considering fault recovery in heterogeneous distributed systems[J]. IEEE Transactions on Parallel & Distributed Systems, 2016, 27(2):521-532.

[11] 邬江兴, 李军飞, 等. 一种异构功能等价体调度装置及方法[P]. 中国, CN106161417A, 2016-11-23.
WU J X, LI J F, et al. A heterogeneous redundancies scheduling equipment and method[P]. China, CN106161417A, 2016-11-23.

[12] 马海龙, 伊鹏, 江逸茗, 等. 基于动态异构冗余机制的路由器拟态防御体系结构[J]. 信息安全学报, 2017, 2(1):29-42.
MA H L, YI P, JIANG Y M, et al. Dynamic heterogeneous redundancy based router architecture with mimic defenses[J]. Journal of Cyber Security, 2017, 2(1):29-42.

[13] 姚文斌, 杨孝宗. 相异性软件组件选择算法设计[J]. 哈尔滨工业大学学报, 2003, 35(3):261-264.
YAO W B, YANG X Z. Design of selective algorithm for diverse software components[J]. Journal of Harbin Institute of Technology, 2003, 35(3):261-264.

[14] 曾国荪, 陈闳中. 探索计算系统异构性的描述[J]. 计算机科学, 2003, 30(12):16-18.
ZENG G S, CHEN H Z. Inquiring the description of heterogeneity among computational systems[J]. Computer Science, 2003, 30(12): 16-18.

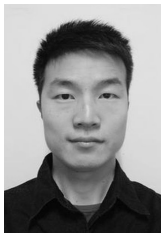
[15] 吴奎, 周献中, 王建宇,等. 基于贝叶斯估计的概念语义相似度算法[J]. 中文信息学报, 2010, 24(2):52-57.
WU K, ZHOU X Z, WANG J Y, et al. A concept semantic similarity algorithm based on bayesian estimation [J]. Journal of Chinese Information Processing, 2010, 24(2):52-57.

[16] 魏帅, 于洪, 顾泽宇, 等. 面向工控领域的拟态安全处理机架构[J]. 信息安全学报, 2017, 2(1):54-73.
WEI S, YU H, GU Z Y, et al. Architecture of mimic security processor for industry control system[J]. Journal of Cyber Security, 2017, 2 (1): 1-12.

[作者简介]

刘勤让（1975−），男，博士，国家数字交换系统工程技术研究中心研究员，主要研究方向为宽带信息网络及芯片设计。

林森杰（1993−），男，广东汕头人，国家数字交换系统工程技术研究中心硕士生，主要研究方向为网络主动防御。



顾泽宇（1993−），男，辽宁沈阳人，国家数字交换系统工程技术研究中心硕士生，主要研究方向为网络主动防御。