软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2020,31(2):564−577 [doi: 10.13328/j.cnki.jos.005615]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

拟态构造的 Web 服务器异构性量化方法*

张杰鑫, 庞建民, 张 铮

(数学工程与先进计算国家重点实验室,河南 郑州 450001)

通讯作者: 庞建民, E-mail: jianmin_pang@hotmail

摘 要: 拟态构造的 Web 服务器是一种基于拟态防御原理的新型 Web 安全防御系统,其利用异构性、动态性、冗余性等特性阻断或扰乱网络攻击,以实现系统安全风险可控.在分析拟态防御技术原理的基础上,论证异构性如何提高拟态构造的 Web 服务器的安全性,并指出对异构性进行量化的重要性.在借鉴生物多样性的量化方法基础上,将拟态构造的 Web 服务器的异构性定义为其执行体集的复杂性与差异性,提出了一种适用于量化异构性的量化方法,通过该方法分析了影响拟态构造的 Web 服务器异构性的因素.在理论上为拟态防御量化评估提供了一种新方法,工程实践上为选择冗余度、构件和执行体提供了指导.实验结果表明,该方法比香浓维纳指数和辛普森指数更适合于量化拟态构造的 Web 服务器的异构性.

关键词: 网络空间安全;拟态防御;Web 服务器;异构性;量化方法

中图法分类号: TP302

中文引用格式: 张杰鑫,庞建民,张铮.拟态构造的 Web 服务器异构性量化方法.软件学报,2020,31(2):564−577. http://www.jos.org.cn/1000-9825/5615.htm

英文引用格式: Zhang JX, Pang JM, Zhang Z. Quantification method for heterogeneity on Web server with mimic construction. Ruan Jian Xue Bao/Journal of Software, 2020,31(2):564−577 (in Chinese). http://www.jos.org.cn/1000-9825/5615.htm

Quantification Method for Heterogeneity on Web Server with Mimic Construction

ZHANG Jie-Xin, PANG Jian-Min, ZHANG Zheng

(State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China)

Abstract: The Web server with mimic construction is a new Web security defense system based on the principle of mimic defence. It uses the heterogeneity, dynamics, redundancy, and other characteristics to block or disrupt network attacks to control the security risk of the system. This study analyzes how heterogeneity can improve the security of the Web server with mimic construction and points out the importance of quantification of heterogeneity. Based on the quantification methods of biodiversity, this study defines the heterogeneity of the Web servers with mimic construction as the complexity and disparity of its execution set, proposes a quantification method that is suitable for quantitative heterogeneity, and analyzes the factors that influence heterogeneity of the Web servers with mimic construction. This study provides a new method for quantitative assessment of mimic defence in theory, and provides guidance for choosing the redundancy, components, and execution in practice. The experimental results show that the proposed method is more suitable for quantifying the heterogeneity of Web server with mimic construction than the Shannon-Wiener index and Simpson index.

Key words: cyberspace security; micmic defence; Web server; heterogeneity; quantification method

网络空间广泛存在着漏洞和后门,加之网络空间软/硬件的一元化,导致安全事件不断发生[1].为了解决这一问题,有很多基于软/硬件多样性的新型防御技术不断出现,但是利用多样性的具体实现手段与目标不尽相同.

* 基金项目: 国家自然科学基金(61472447); 国家重点研发计划(2016YFB0800104); 上海市科学技术委员会科研计划(16DZ1120502)

Foundation item: National Natural Science Foundation of China(61472447); National Key Research and Development Program of China (2016YFB0800104); Research Project of Shanghai Municipal Science and Technology Commission (16DZ1120502)

收稿时间: 2017-12-13; 修改时间: 2018-05-09; 采用时间: 2018-06-21

移动目标防御通过采用多样性技术增强系统的不确定性、多态性,尽可能地减少系统脆弱点的暴露时间,增加攻击困难程度,进而保证系统安全性[2−4].拟态防御技术基于多样化技术,利用“动态异构冗余(dynamic heterogeneous redundancy,简称 DHR)”架构的动态、异构和冗余等特性,使得系统具有内生的安全机理,从而进一步提高系统的安全性[5,6].拟态构造的Web服务器是拟态防御技术的一种典型实例,并且通过测实验证了拟态防御技术的有效性[7,8].虽然拟态防御技术在工程实践上验证了其有效性,但是正是因为其难以量化评估,使其在能够带来多少安全增益等方面存在争议.

现有的量化异构性方法主要通过量化相似性、复杂性等方法来实现.传统的距离函数有欧几里得距离、余弦量化、马氏距离等.通常情况下,这些距离函数用于维数较低的场景会取得较好的描述效果,然而针对高维数据,这些距离函数很难得出准确结论[9].拟态构造的 Web 服务器的执行体的属性维度会比较高,显然不适用于该种方法.文献[10]提出使用最大信息压缩指数的方法来计算特征之间的相似性.这种方法需要定义属性值之间的关系,而这种定义会导致量化不准确.文献[11]提出使用近似熵的方法确定系统复杂度,但是没有很好地解决相对一致性较低的问题,并且无法量化拟态构造的 Web 服务器的执行体间差异性.文献[12]提出了模糊概率模型,对假设和问题进行了形式化描述,并通过推理得出多样性的有效性结论.虽然结论一定程度上佐证了拟态防御技术在一定的方式下寻求多样性的独立性是能够确保提高系统的可靠性的,但是并没有给出具体的量化评估方法.

聚类算法采用量化相似性、差异性等方法,广泛应用于机器学习.文献[13]给出聚类的定义:类簇可以理解为位于多维空间且内部点集密度较高的连通区域,不同类簇之间通过点集密度较低的区域分隔,点集密度越高,实体相似性越高;不同类簇之间实体相似性较低.文献[14,15]分别基于交叉熵、编辑距离计算、余弦相似度等方法来计算聚类集体的相似性.文献[16]采用了7种方法对聚类集体的差异性进行量化,通过实验分析了这些量化方法在不同的对象规模、准确度、数据分布下与各种算法性能之间的关系.以上方法只适用于数据规模较大的情况,而无法用于量化含有数量较少的执行体的拟态构造的 Web 服务器的异构性.

本文第 1 节研究拟态防御技术的内生安全机制,论证异构性与拟态防御安全性的关系.第 2 节结合拟态防御技术的特点提出了拟态构造的 Web 服务器异构性的定义和量化方法.第 3 节在第 2 节的基础上提出影响异构性的因素,并总结提高异构性的方法.第 4 节通过实验对比验证本文提出的量化方法的有效性.

1 拟态构造的 Web 服务器安全性分析

对信息系统安全性、网络态势评估的文献有很多[17−19],但是拟态防御技术因其特殊性,很难直接将这些评估方法进行移植.拟态防御技术基于软/硬件多样性技术和异构性最大化需求,综合了冗余、裁决、主动重构等关键技术,通过 DHR 架构提升安全性.本节论述拟态构造的 Web 服务器的防御原理,阐述异构性对拟态构造的 Web 服务器的重要性.

1.1 拟态构造的Web服务器的架构

典型的 DHR 模型如图 1 所示,模型由输入代理、异构元素池、异构构件集合、调度器、调度算法、执行体集和表决器组成,其中:异构元素池由许多功能模块组成;异构构件集合是从异构元素池中通过组合而得到的 m 个功能等价的异构构件,集合中的元素用 E_i 表示,其中,i=1,2,…,m;执行体集由调度器依据调度算法从异构构件集合中的选取 n 个元素组合构建,执行体集中的元素用 A_j 表示,其中,j=1,2,…,n;输入代理是系统的唯一入口,将输入转发给执行体集中的 n 个执行体,n 个执行体并行地执行任务,然后将各自的执行结果提交到表决器;表决器依据相关表决算法对执行结果进行表决,并将表决结果输出;表决器同时需将表决信息反馈给调度器,后者将根据反馈信息和给定调度算法决定相关执行体清洗恢复或重构等操作.DHR 构造使得系统具有内生的防御机理和先天性免疫的能力,在主动和被动触发条件下,异构软/硬件组件的动态性、伪随机性使得攻击者更加难以基于系统运行情况构建起基于漏洞或后门的攻击通信链路,从而提高系统的安全性[20].

拟态构造的 Web 服务器采用了 DHR 的典型构造,拟态构造的 Web 服务器的系统架构图如图 2 所示.请求分发均衡模块(request dispatching and balancing module,简称 RDB)是用户请求的真实入口,负责将请求复制分

发到功能等价的异构执行体上.非相似 Web 虚拟机池(dissimilar virtual Web server pool,简称 DVSP)是 Web 服务的真实提供者,由异构执行体集里的执行体组成,每个执行体具有一个初始状态.响应多余度表决器(dissimilar redundant responses voter,简称 DRRV),是用户服务的真实出口,负责对执行体的输出进行表决,过滤不一致的输出,保证输出结果的一致性.中心调度器(primary controller,简称 PCON)监测其他功能模块运行状态,保证系统拥有足够的资源,接收 DRRV 的判决信息.动态执行体调度器(dynamicly executing scheduler,简称 DES)是 DVSP 内的控制管理单元,负责接收 PCON 发送的判决结果,依据判决结果和调度策略对执行体进行清洗、回滚等操作.拟态构造的 Web 服务器通过 DHR 架构的动态、异构、冗余等特性提升系统的安全性,本文着重讨论异构性是如何提升系统的安全性的.

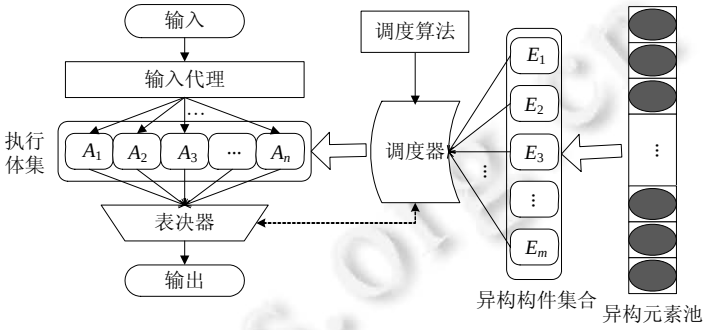


Fig.1 Structure of DHR model

图 1 DHR 模型结构

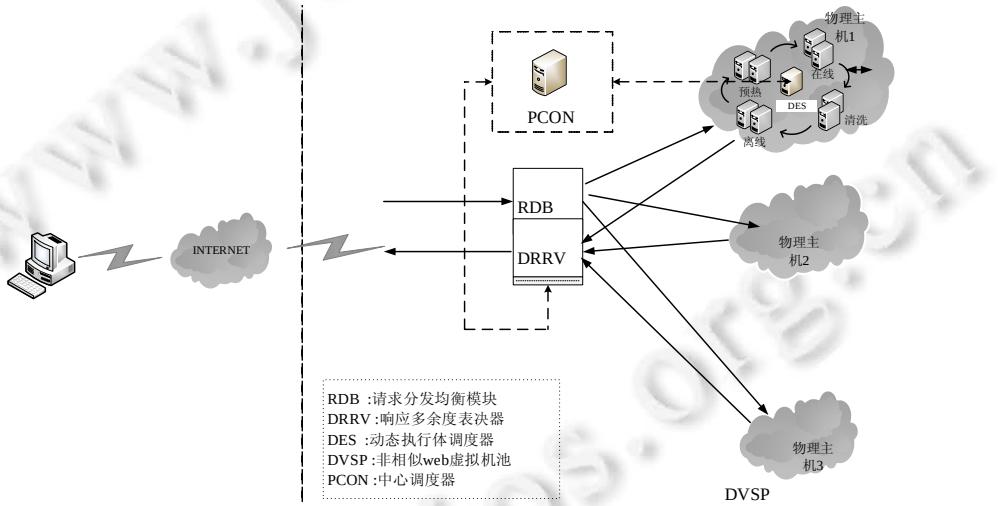


Fig.2 System structure of Web server with mimic construction

图 2 拟态构造的 Web 服务器系统架构

1.2 异构性安全增益分析

定义 1(执行体). 拟态构造的 Web 服务器提供真实 Web 服务的实体,记为 A_i[21].

若执行体 A_i 与执行体 A_j 相同,则记为 $A_i=A_j$;反之,若不同,则记为 $A_i\neq A_j$.

定义 2(执行体集). 拟态构造的 Web 服务器同一时刻上线提供服务,且一起参与响应裁决的执行体组成的集合,记为 $A=\{A_i|A_i$ 是一个执行体,且 $i=1,2,\ldots,n\}$[21].

异构执行体集作为 Web 服务的真实载体,直接承受用户、攻击者的服务请求与攻击行为.而执行体集自身的异构性就可以显著提高拟态构造的 Web 服务器的可靠性和安全性,本小节将阐述异构性是如何提高系统的

安全性的.

定义 3(漏洞集). 执行体 A_i 上漏洞的集合,记为 $V_i=\{V_{ij}|V_{ij}$ 是执行体 A_i 的一个漏洞,且 $j=1,2,\ldots,m\}$[22].

若执行体 A_i 的漏洞集 V_i 与执行体 A_q 的漏洞集 V_q 相同,记为 $V_i=V_q$;若两者完全不同,记为 $V_i\neq V_q$;若两者之间只有部分漏洞相同,即 $V_i\cap V_q\neq\varnothing$,记为 $V_i\approx V_q$.

定义 4(攻击效果). 攻击者利用执行体 A_i 的漏洞集 V_i 中的漏洞 V_{ij} 对执行体 A_i 进行攻击所得到的输出结果,包括异常输出、宕机等,攻击效果记为 EFF_{ij},攻击失败且没有得到攻击效果记为 NULL.

若攻击者利用漏洞 V_{ij} 对执行体 A_i 和 A_q 进行攻击,其中 $i\neq q$,得到的攻击效果完全相同,则记为 $EFF_{ij}=EFF_{qj}$;若攻击者利用漏洞 V_{ij} 对执行体 A_i 和 A_q 进行攻击,其中 $i\neq q$,得到的攻击效果不同,则记为 $EFF_{ij}\neq EFF_{qj}$.

定义 5(攻击过程). 攻击者利用漏洞 V_{ij} 对执行体 A_i 进行的攻击到得到攻击效果的过程,记为

$$f(V_{ij}\rightarrow A_i)=EFF_{ij}.$$

定义 6(攻击表决). DRRV 对攻击者利用漏洞 V_{ij} 对执行体集为 A 进行攻击所得到的攻击效果进行的表决过程记为 $fr(EFF_{1j},EFF_{2j},\ldots,EFF_{nj})$,表决结果记为 R,其中,$j=1,2,\ldots,m$,fr 是表决算法,表决算法可以为相似度表决、大数表决、一致性表决等,本文基于一致性表决进行讨论.

1. 假设一个同构冗余系统,其执行体集 A 中所有执行体都相同,即 $A_1=A_2=\ldots=A_n$,则所有执行体的漏洞集都相同,即 $V_1=V_2=\ldots=V_n$,攻击者利用任一漏洞 V_{ij} 对执行体集 A 进行攻击,则所有执行体都能被攻击成功,且攻击者得到的攻击效果相同,即 $f(V_{1j}\rightarrow A_1)=f(V_{2j}\rightarrow A_2)=\ldots=f(V_{nj}\rightarrow A_n),EFF_{1j}=EFF_{2j}=\ldots=EFF_{nj},j=1,2,\ldots,m$, DRRV 对攻击效果进行表决,表决结果为

$$R=fr(EFF_{1j},EFF_{2j},\ldots,EFF_{nj})=EFF_{ij},i=1,2,\ldots,n.$$

性质 1. 对一个同构冗余系统的任意漏洞进行的攻击都会成功,并造成整个系统的故障或者错误,且会造成一致性的攻击效果,进而导致 DRRV 失效.

2. 假设一个理想的异构冗余系统,其执行体集 A 中所有执行体都完全不相同,即 $A_1\neq A_2\neq\ldots\neq A_n$,则所有执行体的漏洞集都完全不相同,即 $V_1\neq V_2\neq\ldots\neq V_n$,攻击者利用任意执行体 A_i 的漏洞 V_{ij} 对执行体集 A 进行攻击,则攻击只在执行体 A_i 上成功,而在其余执行体均无法成功,即

$$f(V_{ij}\rightarrow A_i)=EFF_{ij},f(V_{ij}\rightarrow A_1)=\ldots=f(V_{ij}\rightarrow A_{(i-1)})=f(V_{ij}\rightarrow A_{(i+1)})=\ldots=f(V_{ij}\rightarrow A_n)=\text{NULL}.$$

DRRV 对攻击效果进行表决,表决结果为

$$R=fr(EFF_{1j},EFF_{2j},\ldots,EFF_{nj})\neq EFF_{ij},i=1,2,\ldots,n.$$

性质 2. 对一个理想的异构冗余系统的任意漏洞的攻击都不会成功,也不会造成整个系统的故障或者错误,且不会造成一致性的攻击效果, DRRV 可以成功阻断攻击.

3. 在实际环境中,不存在理想的异构冗余系统,假设执行体 A_i 的漏洞集为 V_i,执行体 A_q 的漏洞集为 V_q,其中,$i\neq q$,则

$\exists V_{ip}\in V_i$,且 $V_{ip}\in V_1,V_{ip}\in V_2,\ldots,V_{ip}\in V_{i-1},V_{ip}\in V_{i+1},\ldots,V_{ip}\in V_n$,攻击者利用漏洞 V_{ip} 对执行体集 A 进行攻击,则所有执行体都能被攻击成功,且攻击者得到的攻击效果相同,即

$$f(V_{1j}\rightarrow A_1)=f(V_{2j}\rightarrow A_2)=\ldots=f(V_{nj}\rightarrow A_n),EFF_{1j}=EFF_{2j}=\ldots=EFF_{nj},j=1,2,\ldots,m.$$

DRRV 对攻击效果进行表决,表决结果为

$$R=fr(EFF_{1j},EFF_{2j},\ldots,EFF_{nj})=EFF_{ip},i=1,2,\ldots,n.$$

性质 3. 对一个真实的异构冗余系统的共生漏洞进行攻击,可能会造成整个系统的故障或者错误,且会造成一致性的攻击效果,进而导致 DRRV 失效.

$\exists V_{ip}\in V_i$,且 $V_{ip}\in V_1,V_{ip}\in V_2,\ldots,V_{ip}\in V_{i-1},V_{ip}\in V_{i+1},\ldots,V_{ip}\in V_n$,攻击者利用漏洞 V_{ip} 对执行体集 A 进行攻击,则可能至少存在一个执行体 A_q 使得攻击失败,即 $f(V_{ip}\rightarrow A_q)=\text{NULL},q\neq i$,而在其余执行体均成功且得到相同的攻击效果 $f(V_{ip}\rightarrow A_1)=\ldots=f(V_{ip}\rightarrow A_{(q-1)})=f(V_{ip}\rightarrow A_{(q+1)})=\ldots=f(V_{ip}\rightarrow A_n)=EFF_{ip}$,表决器对攻击效果进行表决,表决结果为

$$R=fr(EFF_{1j},EFF_{2j},\ldots,EFF_{nj})\neq EFF_{ip},i=1,2,\ldots\ldots,n.$$

性质 4. 对一个真实的异构冗余系统的共生漏洞进行攻击,可能不会造成整个系统的故障或者错误,且不会

造成一致性的攻击效果,DRRV 可以成功阻断攻击.

$\exists V_{ip}\in V_i$,且至少存在 1 个执行体 A_q,使得 $V_{ip}\notin V_q$,而 $V_{ip}\in V_1,V_{ip}\in V_2,\ldots,V_{ip}\in V_{q-1},V_{ip}\in V_{q+1},\ldots,V_{ip}\in V_{i-1},V_{ip}\in V_{i+1},\ldots,V_{ip}\in V_n$,攻击者利用漏洞 V_{ip} 对执行体集 A 进行攻击,则对执行体 A_q 使得攻击失败,即 $f(V_{ip}\rightarrow A_q)$=NULL,$q\neq i$,而在其余执行体均成功且得到相同的攻击效果 $f(V_{ip}\rightarrow A_1)=\ldots=f(V_{ip}\rightarrow A_{(q-1)})=f(V_{ip}\rightarrow A_{(q+1)})=\ldots=f(V_{ip}\rightarrow A_n)=EFF_{ip}$,DRRV 对攻击效果进行表决,表决结果为

$$R=fr(EFF_{1j},EFF_{2j},\ldots,EFF_{nj})\neq EFF_{ij},i=1,2,\ldots,n.$$

性质 5. 对一个真实的异构冗余系统的某一漏洞进行攻击,不会造成整个系统的故障或者错误,且不会造成一致性的攻击效果,DRRV 可以成功阻断攻击.

针对一个拟态防御系统,我们无法挖掘出系统中每一个执行体的所有漏洞,更无法一一列举出针对每个执行体的漏洞的攻击效果.性质 2 建立的基础是一个绝对异构冗余的环境中,而实际中很难建立起绝对异构冗余的环境,因此性质 3~性质 5 的情况是可能发生的.系统中执行体集中的执行体间的异构性越大,共生的漏洞的存在机率就越低,造成系统一致的攻击效果的机率就越低,系统的安全性就越高.因此,在评价拟态防御系统的安全性时,可以将系统中执行体集的异构性作为评价安全性的指标之一.

多样性是拟态防御技术的基础,拟态构造的 Web 服务器在操作系统层、应用程序层等采用了各种可以替换的功能等价的软件.由于采用了异构的软件,攻击只能导致部分目标组件失效,而其他非目标组件仍能正常工作.而在异构冗余的系统中,执行体集中执行体间的异构性越大,出现共生漏洞的可能性就越低,因此失效组件的数量也就会越低.在拟态构造的 Web 服务器工程实践中,不会选取完全相同的执行体组成执行体集,因此在本文中讨论执行体集异构性时,不包括含有执行体集中含有相同执行体的情况.

1.3 小 结

动态异构冗余的架构不但可以有效地抵御攻击,而且还可以通过比较执行结果,能够检测异常行为.拟态构造的 Web 服务器的异构执行体集不但提供了较高的安全性,而且还提供了较好的可靠性.因此,异构性是拟态构造的 Web 服务器的关键特性之一,如何量化异构性对分析其安全性具有重要作用.

2 执行体集异构性定义量化

上一节分析了异构性为何能够提高拟态构造的 Web 服务器的安全性,并指出,异构性可以作为量化其安全性的指标之一.本节给出异构性的定义,并在分析现有的量化方法的基础上,提出一种适用于拟态构造的 Web 服务器的异构性量化方法.

2.1 异构性定义

通过第 1.2 节的分析可知,拟态构造的 Web 服务器通过异构性来保证同一攻击难以在异构的系统上成功.异构性增加了攻击难度,也就提高了安全性.但是针对异构性却缺少统一的定义.首先,由于拟态构造的 Web 服务器的执行体集采用了异构、冗余的方法,执行体集内执行体种类不唯一,就如同一个生态系统中物种的多样性,因此在定义拟态构造的 Web 服务器的异构性时,要体现出执行体种类的复杂程度[23];其次,执行体种类的复杂程度无法说明执行体间是否存在相同漏洞,因此在定义拟态构造的 Web 服务器的异构性时,更需要体现出执行体间的漏洞集的差异程度[23].由于难以挖掘出所有执行体中已知和未知漏洞,因此将我们将执行体间漏洞集的差异程度定义为执行体间的差异程度.事实证明:两个软件越不相同,存在共生漏洞的可能性就越低[24].

Philip 对异构性的定义满足本文对拟态构造的 Web 服务器的异构性的定义.Philip 将复杂性与差异性作为异构性的两个特征,并详细论述了这种定义的合理性和有效性[25].如图 3 所示,在 a,b,c,d 这 4 个集合中,集合 a 元素种类单一,没有复杂性和差异性;集合 b 具有复杂性,但是差异性很小;集合 c 具有较低的复杂性和较高的差异性;集合 d 具有较高的复杂性和差异性.拟态构造的 Web 服务器的异构性记为 H,拟态构造的 Web 服务器的复杂性记为 C,拟态构造的 Web 服务器的差异性记为 FD,其异构性与复杂性、差异性有如下关系[23].

$$H=C\times FD \tag{1}$$

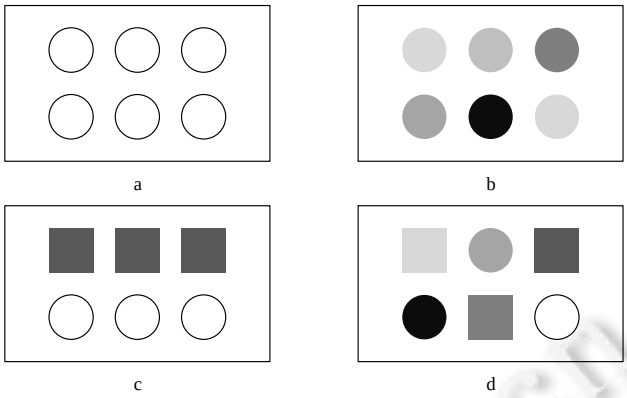


Fig.3 Heterogeneity description

图 3 异构性描述

2.2 执行体集形式化描述

拟态构造的 Web 服务器的执行体集由多个异构执行体组成,其中异构执行体结构不同但功能等价,是整个系统Web服务的主要载体,而执行体集的异构性是拟态构造的Web服务器安全性的衡量指标之一.首先,对拟态构造的 Web 服务器的执行体集进行形式化描述.

定义 7(系统构件集). 拟态构造的Web服务器的所有执行体(包括上线与下线的执行体)的某一类功能等价的软件或者硬件的构件集合,记为 $E_k=\{E_{kj}|E_{kj}$ 是系统的一种组件,$j=1,2,\ldots,m\}$.E_k 中任意两个元素满足功能等价条件,如以操作系统作为一类功能等价的软件,则操作系统的集合为{Centos 6.5,Debian 7.0,Ubuntu 14.10, Ubuntu 16.10,Windows Server 2008,Windows Server 2012}.

定义 8(执行体的特征向量). 拟态构造的 Web 服务器的执行体的特征向量为 $P=(c_{1j},\ldots,c_{mj})^T$,其中,$c_{ij}$ 为执行体 A_j 的第 i 个特征值,m 为需要描述的执行体的特征数量[22].

拟态构造的 Web 服务器是在不同层次上采用了异构的软/硬件,因此在描述执行体时,可以采用系统软/硬件作为其特征.因此,可以将一个 Web 执行体的特征向量简单地描述为不同构件集软/硬件的属性值,如(*ARM*7, *CentOS*7,*nginx*1.12.1,*SQL*3)^T,*ARM*7 属于处理器构件集,*Centos*7 属于操作系统构件集,*nginx*1.12.1 属于 Web 应用构件集,*SQL*3 属于 SQL 指令构件集.为便于描述,将各类的软/硬件进行编号,每一种软件或者硬件对应唯一编号,因此,上面的特征向量可以简化为$(2,3,1,3)^T$,向量中不同位置的特征值代表不同构件集中软/硬件的编号,即"*ARM*7,*Centos*7,*nginx*1.12.1,*SQL*3"的编号,而不同执行体的特征向量之间,相同位置的特征值代表同一构件集.

定义 9(执行体集的特征矩阵). 依据定义 8,拟态构造的 Web 服务器的执行体集 A 的特征矩阵可以进一步形式化描述为

$$C=\begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{pmatrix}.$$

$c_{i1},c_{i2},\ldots,c_{in}$ 表示功能相同的一类构件,$c_{1j},c_{2j},\ldots,c_{mj}$ 表示某执行体的特征属性,其中,n 为执行体集内执行体的数量.由于特征矩阵 C 的每一个特征值仅表示一种软/硬件编号,无法进行数学运算,因此与传统的矩阵相比,特征矩阵 C 有如下性质.

性质 6. 拟态构造的 Web 服务器执行体集的特征矩阵 C 的任意特征值不可进行数学运算,如加、减、乘、除等.

性质 7. 拟态构造的 Web 服务器执行体集的特征矩阵 C 不可进行矩阵运算,如矩阵加、矩阵减、矩阵乘、转置等.

性质 8. 拟态构造的 Web 服务器执行体集的特征矩阵 C 的任意列互换后得到的特征矩阵 C',两个特征矩阵

表示的执行体集相同.

性质 9. 拟态构造的 Web 服务器执行体集的特征矩阵 C 的任意行互换后得到的特征矩阵 C',两个特征矩阵表示的执行体集不同.

定义 10(执行体集的构件集). 执行体集 A 的第 k 类功能等价的软/硬件集合,记为 $EC_k=\{EC_{kj}|EC_{kj}$ 是执行体集 A 的一种软/硬件,$j=1,2,\ldots,m\}$.其中,集合中任意两个元素满足功能等价条件.

这里要注意定义 7 和定义 10 的区别,依据定义 2 对执行体集的定义,执行体集 A 中执行体只包含上线的执行体,并不包含下线的执行体,而定义 10 中的构件集只针对上线的执行体,因此 $EC_k \subseteq E_k$.

定义 11(相对丰度特征向量). 构件集 EC_k 的每一种构件在该类构件总数的丰富度所组成的向量,记为 $p_k=(p_{k1},p_{k2},\ldots,p_{ks})^T$,$p_{ki}$ 为 c_{ki} 及与其相同的构件的丰富度,且[22]

$$\sum_{i=1}^{S} p_{ki} = 1 \tag{2}$$

例如执行体集 A 的特征矩阵 $C=\begin{pmatrix}1 & 2 & 3\\ 2 & 1 & 2\\ 1 & 1 & 2\end{pmatrix}$,其 3 个构件集的相对丰度特征向量分别为 $p_1=\begin{pmatrix}\frac{1}{3} & \frac{1}{3} & \frac{1}{3}\end{pmatrix}^T$,$p_2=\begin{pmatrix}\frac{1}{3} & \frac{2}{3}\end{pmatrix}^T$,$p_3=\begin{pmatrix}\frac{1}{3} & \frac{2}{3}\end{pmatrix}^T$.相对丰度特征向量是量化拟态构造的 Web 服务器的关键数据.

2.3 执行体集复杂性量化

执行体集的复杂性可以用量化生物多样性的方法进行量化,本文采用香农多样性指数用来估算执行体集复杂性的高低,以下简称香农指数.构件集 EC_k 的复杂性的计算公式如下:

$$C_k = -\sum_{i=1}^{S} p_{ki} \ln p_{ki} \tag{3}$$

对于拟态构造的 Web 服务器而言,S 表示构件集 EC_k 所含的构件种类数,p_{ki} 表示软/硬件 i 在构件集 EC_k 中所占比例.当构件集 EC_k 中只存在一种软/硬件时,香农指数达最小值 0;当构件集 EC_k 中存在多种软/硬件构件,且每种构件所包含的成员唯一时,香农指数达到最大值 lnS.

2.4 执行体集差异性量化

执行体集的差异性量化采用二次熵的方法,Rao 提出了一种融合物种之间的差异二次熵的方法去衡量生物多样性[26].二次熵可以计算任何两个随机选择的群体成员之间的平方距离.很多研究者基于 Rao 的二次熵,提出了量化物种的相对丰度和物种间相对差异度的方法[27,28].执行体集的构件集 EC_k 的差异性的计算公式如下:

$$FD_{Qk} = \sum_{i=1}^{S}\sum_{j=1}^{S} d_{kij}^2 p_{ki} p_{kj} \tag{4}$$

其中,d_{kij} 是拟态构造的 Web 服务器执行体集差异性量化的关键参数,代表了构件集 EC_k 中 i,j 两种软/硬件之间的差异($d_{kij}=d_{kji}$,$d_{kii}=0$).d_{kij} 的值可以任意定义,只要满足 $d_{kij}=d_{kji}$ 和 $d_{kii}=0$ 的限制即可.本文通过计算两种构件含有不同漏洞的概率来确定 d_{kij}.如果对于所有 $d_{kij}=1,i\neq j$,满足构件集 EC_k 中存在两种以上软/硬件,且每种构件仅有 1 个成员时,则二次熵转化成辛普森指数,FD_{Qk} 有最大值$(1-1/S)$[29].若构件集 EC_k 中只存在 1 种软/硬件,则对于所有 $d_{kij}=0,i\neq j$,FD_{Qk} 达到最小值 0.当 i,j 两种软/硬件所占比例固定时,d_{kij} 越大,执行体集的差异性就越大.d_{kij} 的取值公式(5)所示.

$$d_{kij} = 1 - \frac{v_{kij}^2}{v_{ki} \times v_{kj}} \tag{5}$$

其中,v_{ki} 和 v_{kj} 分别表示构件 c_{ki} 和 c_{kj} 所含漏洞的数量,v_{kij} 表示构件 c_{ki} 和 c_{kj} 所含相同漏洞的数量,且 $0\leqslant d_{kij}\leqslant 1$.

2.5 执行体集异构性量化

拟态构造的 Web 服务器的执行体集的异构性可以通过计算 M 类执行体集的构件集 EC_k 的异构性来计算,

公式如下:

$$H_A=\sum_{k=1}^{M}C_k\times FD_{Qk} \tag{6}$$

依据上式,执行体集的异构性有如下性质.

性质 10. 若执行体集 A 的执行体满足 $A_1=A_2=\ldots=A_n$,则执行体集的异构性达到最小值 $H_{\min}=0$.

性质 11. 若执行体集 A 的执行体满足 $A_1\neq A_2\neq\ldots\neq A_n$,且任意执行体集的同类构件集中任意两种构件满足 $c_{ki}\neq c_{kj},i\neq j$,则执行体集的异构性达到最大值,公式如下:

$$H_{\max}=M\left(1-\frac{1}{S}\right)\ln S \tag{7}$$

性质 12. 可归约性:若执行体集 A 中存在 x 类构件集,每类构件集中任意两种构件满足 $c_{ki}=c_{kj},i\neq j$,则 $H_A=H_{A'}$,其中,A'为 A 归约 x 类构件集后的执行体集.

证明:执行体集 A 的构件集的集合为$\{EC_1,EC_2,\ldots,EC_m\}$,若存在一个构件集的 EC_q 只含有 1 个元素,则执行体集的第 q 类构件集的相对丰度特征向量为 $p=(1)^T,d_{qij}=0$,则

$$C_q=\sum_{i=1}^{1}p_i\ln p_i=1\times\ln 1=0,$$

$$FD_{Qq}=0,$$

$$H_q=C_q\times FD_{Qq}=0.$$

因此,有执行体集$\{A_i\}$的的异构性:

$$H_A=\sum_{k=1}^{M-1}C_k\times FD_{Qk}+H_q=\sum_{k=1}^{M-1}C_k\times FD_{Qk}.$$

通过归纳,则有:

$$H_A=H_{A'}=\sum_{k=1}^{M-x}C_k\times FD_{Qk}.$$

即执行体集 A 的异构性与 A'的异构性相同,其中,A'为 A 归约 x 类构件集后的执行体集. □

依据性质 12,显然有:执行体集 A 的异构性等于扩充 x 类构件集后的执行体集 A',扩充的每类构件集中任意两种构件满足 $c_{ki}=c_{kj},i\neq j$,即可扩充性.可归约性和可扩充性解决了包含不同类型构件集的执行体集异构性的比较困难问题.从性质 12 可以得出结论,在计算执行体集的不同构件集异构性时是相互独立的.

2.6 小 结

本节通过对拟态构造的 Web 服务器的执行体集异构性进行定义和形式化描述,提出了一种异构性的量化方法.这种方法具有一定的通用性,不仅可以应用于拟态构造的 Web 服务器,而且也可以应用于类似拟态架构的安全系统.

3 提高异构性方法

上一节提出了拟态构造的 Web 服务器的执行体集异构性的量化方法,本节将会分析该种方法对拟态构造的 Web 服务器的作用和现实意义.

3.1 构件差异性对异构性的影响

由公式(7)可以得出,执行体集的异构性 H 与其构件集种类 M、每一类构件集所含构件种类 S 和同一构件集中构件的差异性相关.由公式(6)可知,执行体集内的构件存在差异性的情况下,即使 S 和 M 相同,执行体集的异构性也可能不同;同理,即使执行体集的异构性相同,S 和 M 也可能不同.如图 4 所示,曲面代表了异构性 H 最大值的变化趋势,异构性 H 最小值为 0.因此在选取构件时,要选取差异性较大的构件组成构件集,进而增大使构件集的异构性.

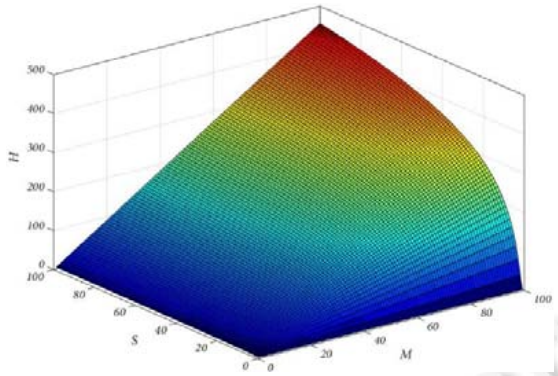


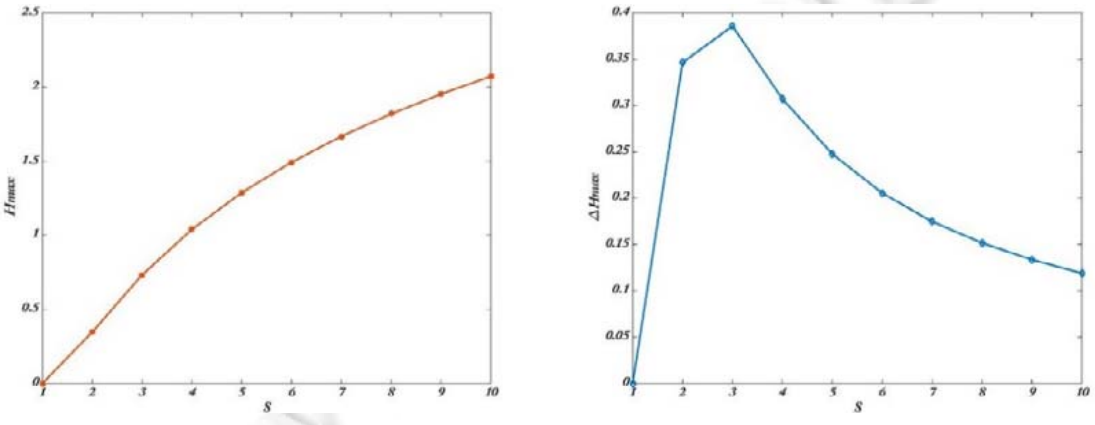
Fig.4 Change of the maximum heterogeneity

图 4 最大异构性变化

3.2 冗余度的选择方法

执行体集的构件集的异构性直接决定了执行体集的异构性,因此可以分开讨论执行体集的每一类构件集异构性.当执行体集的任意构件集 EC_k 所含的构件中任意两种构件的 d_{kij}=1,且构件种类 S 与执行体集中所含的执行体的数量 n 相同时,由公式(7)可知,执行体集的某类构件集 EC_k 异构性达到最大.当 S=N 时,$H_{\max}$ 与 S 正相关,执行体集的某类构件集 EC_k 异构性与执行体数量的关系,如图 5(a)所示.虽然构件集的构件种类增加会增加异构性,但是一味地增加构件集的种类并不会显著增加异构性,由图 5(b)可知,当与构件种类 S=3 时,构件集异构性增益达到最大值$\Delta H_{\max}$=0.3858;而当 $S\geqslant 3$ 时,$\Delta H_{\max}$ 随 S 的增加而减小.因此可以得出结论:当 S=N 时,执行体集的异构性会随着执行体集的构件集所含构件种类的增加而增加;但是,随着执行体集的构件集所含构件种类的增加,执行体集异构性的增益会显著降低.因此,构件集内使用构件种类的增加并不能显著提升系统的异构性.而且实际应用中,能够等价替换的构件的数量是有限的,并且随着使用的构件的增多,会使系统的复杂性、成本开销等增加.

当执行体集的任意构件集 EC_k 所含的构件中任意两种构件的 d_{kij}=1,且构件种类 S 小于执行体集中所含的执行体的数量 N 时,执行体集的某类构件集 EC_k 异构性与执行体数量的关系如图 6 所示.可以得出结论:当 S<N 时,随着执行体集内执行体数量的增加,执行体集的构件集的异构性并不会增加;当 N=xS(x 为正整数)时,执行体集的最大异构性相同.理论上,含有 2 个完全异构的执行体的系统就可以实现容错,含有 3 个完全异构的执行体的系统就可以实现入侵检测[30,31].因此在工程实践上,一般选取 3 作为执行体集的冗余度.



(a) (b)

Fig.5 Relationship between the maximum heterogeneity and number of element species

图 5 最大异构性与构件种类数量关系

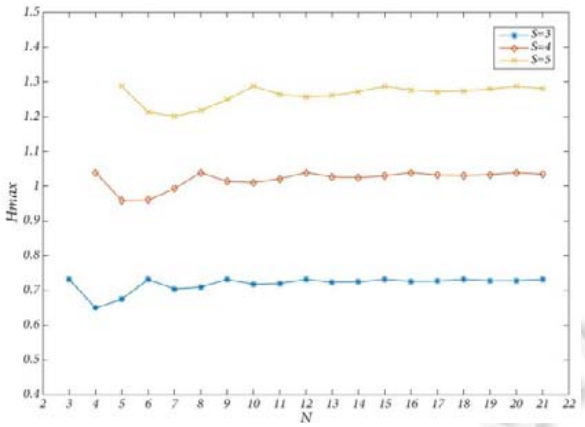


Fig.6 Relationship between the maximum heterogeneity and number of executor

图 6 最大异构性与执行体数量关系

3.3 执行体的选择方法

通过公式(7)可知,执行体集的异构性的最大值和执行体集的构件集 EC_k 的种类数 M 与执行体集的构件集 EC_k 所含的构件种类数 S 相关.由性质 12 可知,当执行体集的构件集的种类越多时,执行体集的异构性越大.但是一般情况下,系统的构件集种类少于系统所含执行体数量,因此也可能造成执行体集的构件集所含构件的种类数小于执行体集所含执行体的数量的情况.图 7(a)~图 7(c)展示了系统构件集种类分别为 3,5,7 的 3 个拟态构造的 Web 服务器系统,图 7(d)是这 3 个拟态构造的 Web 服务器系统在不同执行体数量时,执行体集的异构性最大值.从图中可以看出,在执行体数量相同的情况下,构件集的种类越多,异构性最大值就越大.因此,选择执行体组成执行体集时,需要选择能够构成异构构件集种类较多的执行体.

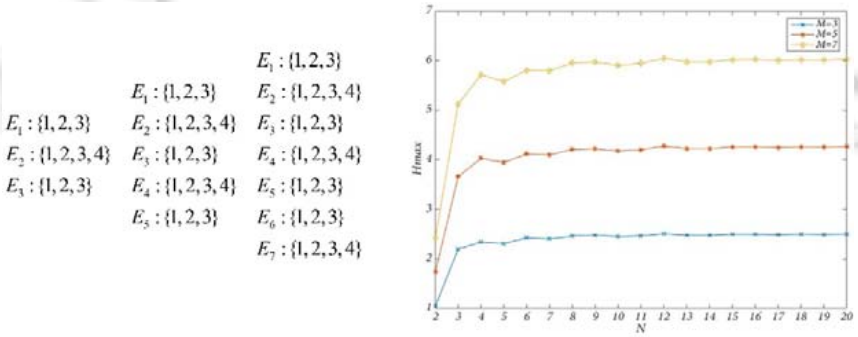


Fig.7 Relationship between the maximum heterogeneity and number of element set

图 7 最大异构性与构件集种类关系

3.4 小 结

由上可知,在选取构件时,要选取异构性较大的构件组成构件集;确定执行体集的冗余度时,并不是冗余度越大越好,而是需要考虑每一种构件集的构件种类和执行体数量之间的关系,当执行体数量为 3 时,异构性的增益最大;在选择执行体组成执行体集时,需要选择能够组成异构构件集种类较多的执行体.

4 实验分析

我们选取一些软件组成了 10 个不同的拟态构造的 Web 服务器系统执行体集,其中,应用程序层都使用 PHP,

操作系统层和 Web 服务器层使用不同软件,见表 1.因为应用程序层相同,在计算异构性时,可以将此层的异构性直接记作为 0.表 2 是按照第 2.4 节的方法,在构件漏洞统计分析结果的基础上得出的构件之间的差异性.此外,假定未在表中的构件之间的差异性均为 1,其中,漏洞信息来源于 Common Vulnerabilities & Exposures(CVE).

Table 1 Heterogeneity of element set quantification results contrast

表 1 执行体集异构性量化结果对比

编号	软件栈	香农指数	辛普森指数	本文方法
1	Ubuntu 12.04+Apache 2.4.0+PHP Windows Server 2003+IIS 6.0+PHP RedHat 7+Nginx 1.12.0+PHP	2.197 2	1.333	1.458 4
2	Ubuntu 12.04+Apache 2.4.0+PHP Windows Server 2012+IIS 7.0+PHP RedHat 7+Nginx 1.12.0+PHP	2.197 2	1.333	1.458 4
3	Debian 7.0+Nginx 1.12.0+PHP Windows Server 2016+Lighttpd 1.4.48+PHP Windows 7+Apache2.4+PHP	2.197 2	1.333	1.390 0
4	Ubuntu 12.04+Nginx 1.12.0+PHP Windows Server 2003+IIS 6.0+PHP Windows 7+Apache2.4+PHP	2.197 2	1.333	1.379 1
5	Windows Server 2003+IIS 6.0+PHP Windows Server 2012+IIS 7.0+PHP Windows 7+Nginx 1.12.0+PHP	2.197 2	1.333	1.082 1
6	Ubuntu 12.04+Apache 2.4.0+PHP Windows Server 2008+Apache 2.4.0+PHP RedHat 7+Nginx 1.12.0+PHP	1.734 1	1.111	1.009 0
7	Ubuntu 12.04+Apache 2.4.0+PHP Debian 7.0+Apache 2.4.0+PHP RedHat 7+Nginx 1.12.0+PHP	1.734 1	1.111	0.977 6
8	Ubuntu 12.04+Nginx 1.12.0+PHP Windows Server 2016+Lighttpd 1.4.48+PHP Windows 7+Nginx 1.12.0+PHP	1.734 1	1.111	0.949 5
9	Ubuntu 12.04+Apache 2.4.0+PHP Windows Server 2008+Apache 2.4.0+PHP RedHat 7+Apache 2.4.0+PHP	1.098 6	0.666 7	0.726 1
10	Ubuntu 12.04+Apache 2.4.0+PHP Windows Server 2008+Apache 2.4.0+PHP Windows 7+Apache2.4+PHP	1.098 6	0.666 7	0.497 8

Table 2 Table of disparity parameter

表 2 差异性参数表

构件 1	构件 2	*d*
Ubuntu 12.04	RedHat 7	0.986 9
Windows Server 2003	Windows 7	0.805 6
Windows Server 2016	Windows 7	0.832 8
Ubuntu 12.04	Debian 7.0	0.938 5
RedHat 7	Debian 7.0	0.995 4
Windows Server 2003	Windows Server 2012	0.974 8
Windows Server 2012	Windows 7	0.513 2
Windows Server 2008	Windows 7	0.197 9
IIS 6.0	IIS 7.0	0.754 9

(1) 执行体集 1、执行体集 2 与执行体集 3、执行体集 4 相比,虽然在 Web 服务器层其异构性相同,但是执行体集 1、执行体集 2 在操作系统层使用了两种 Linux 的操作系统,执行体集 3、执行体集 4 使用了两种 Windows 操作系统,所以执行体集 1、执行体集 2 的异构性高于执行体集 3、执行体集 4.

(2) 执行体集 3 与执行体集 4 相比,虽然两者在 Web 服务器层异构性相同,但是在操作系统层前者的异构性要高于后者,所以执行体集 3 的异构性高于执行体集 4.

(3) 执行体集 4 与执行体集 5 相比,执行体集 4 的一个执行体在操作系统层使用了 Linux 操作系统,而执行体集 5 在操作系统层都是用了 Windows 操作系统,所以执行体集 4 在操作系统层的异构性要高于

执行体集5;且执行体集5在Web服务器层使用IIS 6.0和IIS 7.0,两种构件的异构性较低,所以执行体集4在Web服务器层的异构性也高于执行体集5,因此,执行体集4的异构性高于执行体集5.

(4) 执行体集5与执行体集6相比,执行体集5在操作系统层都使用Windows操作系统,所以执行体集6在操作系统层的异构性要高于执行体集5,但是执行体集6的两个执行体在Web服务器层使用相同的Apache 2.4.0,所以执行体集5在Web服务器层的异构性也高于执行体集6.综合两层的异构性可知,执行体集5的异构性高于执行体集6.

(5) 执行体集6与执行体集7相比,虽然在Web服务器层其异构性相同,但是执行体集7在操作系统层使用了3种Linux操作系统,因此执行体集6的异构性高于执行体集7.

(6) 执行体集7与执行体集8相比,虽然在Web服务器层其异构性相同,但是执行体集8的两个执行体在操作系统层使用两种异构性较低的Windows操作系统,因此,执行体集7的异构性高于执行体集8.

(7) 执行体集8与执行体集9相比,虽然在操作系统层,执行体集9的异构性高于执行体集8,但是在Web服务器层,执行体集9的异构性为0,因此,执行体集8的异构性高于执行体集9.

(8) 执行体集9与执行体集10相比,执行体集9和执行体集10在Web服务器层的异构性均为0,但是在操作系统层,执行体集9使用了两种Linux操作系统,而执行体集10使用了两种异构性很低的Windows操作系统,因此.执行体集9的异构性高于执行体集10.

由表1可知,使用传统的量化生物多样性方法,如香浓指数和辛普森指数,不能有效地量化执行体集的差异性;而本文所使用的方法可以很好地解决该问题,而且本文方法对执行体集异构性的区分度明显高于香浓指数和辛普森指数.

5 结束语

异构Web服务器构成了拟态构造的Web服务器系统的核心.硬件、操作系统、应用程序等存在差异性,降低了异构服务器发生共模故障的可能性,同时也增加了其中存在的漏洞后门、病毒木马等的不一致性,进而提高了可靠性与安全性.本文针对拟态构造的Web服务器异构性难以量化的问题,将复杂的执行体集的异构性层层分解为每一类构件集的异构性,引入香农指数和二次熵量化构件集的异构性.在该方法的基础上,分类并讨论了影响异构性的因素.与其他量化方法相比,该方法对于拟态构造的Web服务器的异构性量化效果更好,更能反映实际情况.

下一步将会利用软件同源性分析[32,33]等方法进一步改进和完善差异性量化方法.本文只从异构性的角度分析了拟态防御技术的有效性,而在负反馈机制基础上的动态性是拟态防御技术另一个重要特性.今后我们将在本文量化方法的基础上,综合考虑成本和收益等因素,改进和完善拟态构造的Web服务器的执行体动态调度策略.

References:

[1] Birman KP, Schneider FB. The monoculture risk put into context. Security & Privacy, 2009,7(1):14−17.

[2] Kewley DL, Bouchard JF. DARPA information assurance program dynamic defense experiment summary. IEEE Trans. on Systems, Man, and Cybernetics—Part A: Systems and Humans, 2001,31(4):331−336.

[3] Jajodia S, Ghosh AK, Swarup V, *et al*. Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. New York: Springer-Verlag, 2011.

[4] Cai GL, Wang BS, Wang TZ, *et al*. Research and developmentof moving target defense technology. Journal of Computer Research and Development, 2016,53(5):968−987 (in Chinese with English abstract).

[5] Wu JX. Meaning and vision of mimic computing and mimic security defense. Telecommunications Science, 2014,30(7):1−7 (in Chinese with English abstract).

[6] Wu JX, Zhang F, Luo XG. Mimic computing and mimic security defense. Communications of the CCF, 2015,11(1):8−14 (in Chinese with English abstract).

[7] Tong Q, Zhang Z, Zhang WH, Wu JX. Design and implementation of mimic defense Web server. Ruan Jian Xue Bao/Journal of Software, 2017,28(4):883−897 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5192.htm [doi: 10.13328/j.cnki.jos.005192]

[8] Zhang Z, Ma BL, Wu JX. The test and analysis of prototype of mimic defense in Web servers. Journal of Cyber Security, 2017,2(1): 13−28 (in Chinese with English abstract).

[9] Shao CS, Lou W, Yan LM. Optimization of algorithm of similarity measurement in high-dimensional data. Computer Technology and Development, 2011,21(2):1−4 (in Chinese with English abstract).

[10] Mitra P, Murthy CA, Pal SK. Unsupervised feature selection using feature similarity. IEEE Trans. on Pattern Analysis & Machine Intelligence, 2002,24(3):301−312.

[11] Pincus SM. Approximate entropy as a measure of system complexity. Proc. of the National Academy of Sciences of the United States of America, 1991,88(6):2297−2301.

[12] Salako K, Strigini L. When does “diversity” in development reduce common failures? Insights from probabilistic modeling. IEEE Trans. on Dependable and Secure Computing, 2014,11(2):193−206.

[13] Jain AK, Dubes RC. Algorithms for Clustering Data. Englewood Cliffs: Prentice Hall, 1988.

[14] Yu H, Zhao YL, Cui K, *et al*. Community detection algorithm based on cross-entropy method. Chinese Journal of Computers, 2015, 38(8):1574−1581 (in Chinese with English abstract).

[15] Xu ZM, Li D, Liu T, *et al*. Measuring similarity between microblog users and its application. Chinese Journal of Computers, 2014, 37(1):207−218 (in Chinese with English abstract).

[16] Luo HL, Kong FS, Li YX. An analysis of diversity measures in clustering ensembles. Chinese Journal of Computers, 2007,30(8): 1315−1324 (in Chinese with English abstract).

[17] Xi RR, Yun XC, Zhang YZ, *at el*. An improved quantitative evaluation method for network security. Chinese Journal of Computers, 2015,38(4):749−758 (in Chinese with English abstract).

[18] Ram M. On system reliability approaches: A brief survey. Int’l Journal of System Assurance Engineering and Management, 2013, 4(2):101−117.

[19] Avritzer A, Czekster RM, Distefano S, *et al*. Software aging and rejuvenation for increased resilience: modeling, analysis and applications. In: Proc. of the Resilience Assessment and Evaluation of Computing Systems. Berlin, Heidelberg: Springer-Verlag, 2012. 167−183.

[20] Wu JX. Research on cyber mimic defense. Journal of Cyber Security, 2016,1(4):1−10 (in Chinese with English abstract).

[21] Zhang JX, Pang JM, Zhang Z, Tai M, Liu H. QoS quantification method for Web server with mimic construction. Computer Science, 2019,46(11):109−118 (in Chinese with English abstract).

[22] Zhang JX, Pang JM, Zhang Z, Tai M, Zhang H, Nie GL. Executors scheduling algorithm for Web server with mimic structure. Computer Engineering, 2019,45(8):14−21 (in Chinese with English abstract).

[23] Zhang JX, Pang JM, Zhang Z, Tai M, Liu H. Heterogeneity quantization method of cyberspace security system based on dissimilar redundancy structure. Journal of Electronics and Information Technology, 2019,41(7):1594−1600 (in Chinese with English abstract).

[24] Han J, Zang BY. Analyzing the effectiveness of software diversity for system security. Computer Applications and Software, 2010, 27(9):273−275 (in Chinese with English abstract).

[25] Twu P, Mostofi Y, Egerstedt M. A measure of heterogeneity in multi-agent systems. In: Proc. of the American Control Conf. IEEE, 2014. 3972−3977.

[26] Rao CR. Diversity and dissimilarity coefficients: A unified approach. Theoretical Population Biology, 1982,21(1):24−43.

[27] Ding N, Yang W, Zhou Y, *et al*. Different responses of functional traits and diversity of stream macroinvertebrates to environmental and spatial factors in the Xishuangbanna watershed of the upper Mekong River Basin, China. Science of the Total Environment, 2017,574:288−299.

[28] Liu ZJ. Bootstrapping one way analysis of Rao’s quadratic entropy. Communication in Statistics-Theory and Methods, 2007,20(20): 1683−1703.

[29] Botta-Dukát Z. Rao's quadratic entropy as a measure of functional diversity based on multiple traits. Journal of Vegetation Science, 2010,16(5):533−540.

[30] Chen L, Avizienis A. *N*-version programming: A fault-tolerance approach to reliability of software operation. In: Proc. of the 8th Int'l Conf. on Fault Tolerant Computing. 1978. 3−9.

[31] Gashi I, Popov P. Rephrasing rules for off-the-shelf SQL database servers. In: Proc. of the European Dependable Computing Conf. IEEE Computer Society, 2006. 139−148.

[32] Luo L, Ming J, Wu D, *et al*. Semantics-based obfuscation-resilient binary code similarity comparison with applications to software plagiarism detection. In: Proc. of the ACM Sigsoft Int'l Symp. on Foundations of Software Engineering. ACM, 2014. 389−400.

[33] Jhi YC, Jia X, Wang X, *et al*. Program characterization using runtime values and its application to software plagiarism detection. IEEE Trans. on Software Engineering, 2015,41(9):925−943.

附中文参考文献:

[4] 蔡桂林,王宝生,王天佐,等.移动目标防御技术研究进展.计算机研究与发展,2016,53(5):968−987.

[5] 邬江兴.专题导读——拟态计算与拟态防御的原意和愿景.电信科学,2014,30(7):1−7.

[6] 邬江兴,张帆,罗兴国.拟态计算与拟态安全防御,中国计算机学会通讯,2015,11(1):8−14.

[7] 仝青,张铮,张为华,邬江兴.拟态防御 Web 服务器设计与实现.软件学报,2017,28(4):883−897. http://www.jos.org.cn/1000-9825/5192.htm [doi: 10.13328/j.cnki.jos.005192]

[8] 张铮,马博林,邬江兴.Web 服务器拟态防御原理验证系统测试与分析.信息安全学报,2017,2(1):13−28.

[9] 邵昌昇,楼巍,严利民.高维数据中的相似性度量算法的改进.计算机技术与发展,2011,21(2):1−4.

[14] 于海,赵玉丽,崔坤,等.一种基于交叉熵的社区发现算法.计算机学报,2015,38(8):1574−1581.

[15] 徐志明,李栋,刘挺,等.微博用户的相似性度量及其应用.计算机学报,2014,37(1):207−218.

[16] 罗会兰,孔繁胜,李一啸.聚类集成中的差异性度量研究.计算机学报,2007,30(8):1315−1324.

[17] 席荣荣,云晓春,张永铮,等.一种改进的网络安全态势量化评估方法.计算机学报,2015,38(4):749−758.

[20] 邬江兴.网络空间拟态防御研究.信息安全学报,2016,1(4):1−10.

[21] 张杰鑫,庞建民,张铮,郜铭,刘浩.拟态构造 Web 服务器的服务质量量化方法.计算机科学,2019,46(11):109−118.

[22] 张杰鑫,庞建民,张铮,郜铭,张浩,聂广来.面向拟态构造 Web 服务器的执行体调度算法.计算机工程,2019,45(8):14−21.

[23] 张杰鑫,庞建民,张铮,郜铭,刘浩.基于非相似余度架构的网络空间安全系统异构性量化方法.电子与信息学报,2019,41(7):1594−1600.

[24] 韩进,臧斌宇.软件相异性对于系统安全的有效性分析.计算机应用与软件,2010,27(9):273−275.

张杰鑫(1989－),男,天津人,博士,主要研究领域为网络空间安全,高性能计算.

张铮(1976－),男,博士,副教授,主要研究领域为网络空间安全,先进计算.

庞建民(1964－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为高性能计算,信息安全.