一种基于 Bio-PEPA 的分布式虚拟化系统脆弱性扩散模型

吕宏武 王慧强 林俊宇 冯光升 郭方方

(哈尔滨工程大学计算机科学与技术学院 哈尔滨 150001)

摘 要 脆弱点类型差异和脆弱性演化对脆弱性扩散过程具有显著影响,而现有脆弱性扩散模型对此还缺少深入研究.该文提出一种基于分簇思想的分布式虚拟化系统脆弱性扩散模型,首先按照节点包含脆弱点类型的不同进行分簇,其次利用 Bio-PEPA 静态分层特性,对脆弱性在簇内、簇间传播,以及簇间迁移演化过程进行建模.最后,将 Bio-PEPA 模型转化为常微分方程求解,分析分布式虚拟化系统脆弱性扩散的特点和规律,避免了传统分析方法的状态空间爆炸问题.实验结果显示,可以通过提升系统修复能力、降低簇间传播速率、减小簇间变迁速率,抑制分布式虚拟化系统的脆弱性扩散.

关键词 脆弱性分析;脆弱性扩散;分布式虚拟化系统;Bio-PEPA;云计算

中图法分类号 TP309 **DOI 号** 10.11897/SP.J.1016.2016.00391

A Vulnerability Propagation Model of Distributed Virtualized Systems Based on Bio-PEPA

LV Hong-Wu WANG Hui-Qiang LIN Jun-Yu FENG Guang-Sheng GUO Fang-Fang

(*College of Computer Science and Technology, Harbin Engineering University, Harbin* 150001)

Abstract Vulnerability is usually the essential reason of security and dependability. Recently, enormous amounts of third-party applications appear on distributed virtualized systems, which bring out a lot of additional vulnerabilities even more than the inherent vulnerabilities in the systems. Meanwhile, the vulnerabilities are propagated rapidly by frequent interactions and unreasonable trust relationship among nodes. Vulnerability propagation has grown up to be a serious problem. Different types of vulnerabilities and vulnerability evolution have a significant impact on the process of vulnerability propagation, but the existing vulnerability propagation models have not considered these issues. In order to make the model more reasonable, we propose a new vulnerability propagation model for distributed virtualized systems based on clustering. In this model, the same kind of vulnerabilities is regarded as in a single cluster, and then the vulnerability propagation in/between clusters as well as vulnerability migration between clusters is modeled by Bio-PEPA (Performance Evaluation Process Algebra) in a static hierarchy manner. Besides, the Bio-PEPA model we have proposed is converted into ODEs (Original Differential Equations) to discover the law of vulnerability propagation, avoiding the state space

收稿日期:2015-03-12;在线出版日期:2015-07-23.本课题得到国家自然科学基金(61402127,61370212)和黑龙江省自然科学基金(F2015029)资助.**吕宏武**,男,1983年生,博士,讲师,中国计算机学会(CCF)会员,主要研究方向为可用性、云计算、性能评估.E-mail:lvhongwu@hrbeu.edu.cn.**王慧强**,男,1960年生,博士,教授,中国计算机学会(CCF)会员,主要研究领域为网络安全、未来网络、可信性.**林俊宇**,男,1981年生,博士,助理研究员,中国计算机学会(CCF)会员,主要研究方向为网络安全、脆弱性分析.**冯光升**,男,1980年生,博士,讲师,中国计算机学会(CCF)会员,主要研究方向为网络安全、认知网络.**郭方方**,男,1974年生,博士,副教授,主要研究方向为网络安全态势感知、云监控.

explosion existing in traditional analysis methods. The experimental results show that the vulnerability propagation progress can be retained by enhancing the recovery capability, decreasing the rate of vulnerability propagation and reducing the rate of vulnerability migration between clusters. Our works provide an insight into the nature of the vulnerability propagation of distributed virtualized systems, and it is useful to improve the security of the systems.

Keywords vulnerability analysis; vulnerability propagation; distributed virtualized system; Bio-PEPA; cloud computing

1 引 言

脆弱性通常是引发系统安全性、可信性等问题的内在原因[1],一直以来就是软件开发和系统测试领域关注的研究热点.但是随着以覆盖网、未来网络和云计算为代表的分布式虚拟化系统的广泛应用,系统脆弱性变得尤为严峻.一方面,新兴业务的不断涌现,带来了第三方软件的爆发式增长,然而它们通常缺少严苛的编码规范和系统化测试,存在脆弱性漏洞的概率远超系统本身.包括著名安全机构Secunia在内的大量研究显示①,因第三方应用而引入的脆弱点在数量上已远远超过系统固有的脆弱点.另一方面,作为云计算、覆盖网络技术基础的虚拟化技术,从根本上变革了资源的利用模式.服务使用的资源往往来自于远程"云"中的不同位置,它们通过动态化的组合产生满足用户需求的个性化服务[2],而这些资源自身的脆弱性往往难以断定,并将通过频繁的动态连接而快速扩散,为脆弱性的防护带来了巨大挑战.

脆弱性分析作为一个热点领域已经在脆弱性检测、系统脆弱性评估和脆弱性风险分析等方面进行了大量研究[3],但是针对脆弱性扩散的研究尚处于初期阶段.(1)最初的脆弱性扩散研究主要关注于特定系统的脆弱性风险,如OpenBSD操作系统漏洞[4]、Mozilla系统漏洞[5]和数据错误在软件中的传播[6],通常以多年运行数据为基础来挖掘脆弱性的传播规律.但是这些结果往往具有很强的系统相关性,而且很难用于系统设计阶段;(2)近来,Feng等人[7]建立了一种安全性风险模型(SRAM),通过贝叶斯网络来模拟风险因子及它们之间的因果关系,估计最大概率的脆弱性传播路径以及带来的风险值.而De等人[8]针对无线传感器网络环境下由于多跳广播协议带来的脆弱性扩散问题进行了研究.文献[7-8]仅在系统整体层面研究脆弱性的扩散,均未考虑脆弱性在同种类型脆弱点和不同类型之间扩散的差异,以及脆弱性的演化问题;(3)为了在系统设计阶段减少脆弱性,文献[9]研究了由于代码重用造成的脆弱点在旧有代码与新代码之间传播的问题,并对每一类脆弱点建立单独的脆弱性树(Vulnerable trees).文献[10]则研究了在面向对象设计阶段由类的耦合而引入的脆弱性扩散,并采用脆弱性树和自动柜员机(Automated Teller Machine, ATM)方法进行了分析.但是这些研究虽然对不同脆弱点类型进行了区分,却仅考虑了脆弱性在同种类型脆弱点之间的传播,而没有考虑在不同类型脆弱点之间传播的问题.此外,当系统规模巨大时,组件数量及其相互关联可能达到数以百万计,此时利用脆弱性树或者ATM进行脆弱性扩散分析,将面临状态空间爆炸的问题.

综上所述,虽然现有研究为脆弱性扩散分析和预测提供了重要基础,但是对于分布式虚拟化系统中由于动态连接而引入的脆弱性扩散仍具有一定的局限性,主要表现在:(1)由于虚拟化技术,在分布式虚拟化系统中服务所用资源处于远程"模糊化"的云中,无法事先预测连接的对象,因而脆弱性扩散具有很强的概率性,基于事后日志的分析方法往往难以准确描述这种随机选择;(2)虽然现有部分模型已将不同的脆弱点分类,但通常仅仅研究脆弱性在同种类型脆弱点之间的扩散,即假定扩散是均匀地,而忽略了脆弱性在不同类型脆弱点之间传播的差异;(3)此外,随着节点规模的扩大(甚至数以亿计),传统的基于状态空间的分析方法如马尔可夫过程、脆弱性树等面临严重的状态空间爆炸问题[11].

针对现有模型的不足,考虑分布式虚拟化系统

① Secunia. Vulnerability Update, August, September, October 2014. http://secunia.com/?action=fetch&filename=secunia-vulnerability-update-no1_2014.pdf, 2014

节点脆弱性的差异，本文把具有相同类型脆弱点的节点抽象为同一个簇，研究脆弱性在簇内和簇间传播的过程，以及脆弱点由一种类型演化为其他类型时脆弱性的簇间迁移问题. 在此基础上，利用生物性能评价进程代数 Bio-PEPA 对分布式虚拟化系统脆弱性扩散过程进行描述和分析. 文章组织结构如下，首先对 Bio-PEPA 进行简单介绍，然后对分布式虚拟化系统的脆弱性扩散过程建模，并就模型关键参数对于脆弱性扩散趋势的影响进行分析，最后做出总结.

2　Bio-PEPA 的基础语义

Bio-PEPA 是在性能评价进程代数 PEPA(Performance Evaluation Process Algebra)基础之上发展而来的一种新型形式化语言. 由于引入了生物网络(Biological Network)的特点，其具有静态分层(static hierarchy)特性[12]，能够抽象出一系列独立的位置(location)，因而适宜于本文描述脆弱性在不同类型脆弱点之间扩散的过程. 同时，由于 Bio-PEPA 可以转化为 ODEs(Ordinary Differential Equations)求解，能够克服传统的基于状态空间方法面临的状态空间爆炸问题，所以更加适用于组件数量众多的大型分布式虚拟化环境.

Bio-PEPA 基本组成元素包括两个种类，分别是种类组件 S(species components)和模型组件 P(model components)，前者用于描述每一种类的行为，后者用于描述种类之间的交互及初始化. Bio-PEPA 基本语意表达式如下[12-13]：

$$S ::= (\alpha, r) op\ S \mid S + S \mid Constant \mid S@L \quad (1)$$

$$op ::= \downarrow \mid \uparrow \mid \odot \quad (2)$$

$$P ::= P \underset{G}{\bowtie} P \mid S(x) \quad (3)$$

其中各表达式的含义如下：

(1) 在前缀(α, r)中，α 代表动作，r 代表执行动作 α 时种类 S 的计量系数，又称为变迁速率. 当 S 为反应物时 r 为正值，当 S 为生成物时 r 为负值. 操作符“+”代表 S 之间的选择. *Constant* 代表常量. $S@L$代表在位置 L 中的组件 S，通常一个位置可以描述为 L:*s unit*, *kind*, 其中 *s* 代表大小，*unit* 是与之相关的度量单位，*kind* 代表种类. 当仅有一个位置 L 时，则可以省略不写，在本文中位置 L 代表独立的簇.

(2) 操作 *op* 代表 S 在反应中的作用，包括 S 为反应物($\downarrow$)、生成物($\uparrow$)和通用修改($\odot$)等情况.

(3) $\underset{*}{\bowtie}$ 代表合作操作符，其中 G 是在合作过程中必须同步的动作集合. 若记为 $\underset{*}{\bowtie}$ 则代表在合作过程中所涉及的动作必须同步执行. $S(x)$中的 $x \in \boldsymbol{N}^+$，表示初始时刻组件的数量.

(4) 对于任意的反应$\langle Reaction \rangle S_1 + S_2 \rightarrow S_3 + S_4$，反应速率 f_k 有两种基本种类 mass-action 和 Michealis-Menten. 不同于生物系统，分布式虚拟化系统作为一种信息系统，仅包含第 1 种反应，即组件的浓度越高则发生反应的概率越大. 同时，允许非最简反应式的存在[12]，即可以存在$\langle Reaction1 \rangle S_1 + S_2 \rightarrow S_1 + S_4$或$\langle Reaction2 \rangle S_1 + S_2 \rightarrow 2S_3$的情况. 此时用表达式$(\alpha, (r_1, r_2)) \odot$来表示组件 S_1 的变迁速率的改变，其中 r_1、r_2 分别是反应前后的变迁速率，$r = r_2 - r_1$. 当 $r=1$时，$(\alpha, k)\ op\ S$ 可以省略为 $\alpha\ op\ S$. 限于篇幅，详细语义逻辑可参见文献[13].

求解 Bio-PEPA 主要是为了获得各组件的近似稳态概率，继续对系统做进一步的分析. 所谓近似稳态概率是指模型稳定后，各种类组件的数量(population level)占所有组件总量的比例. 若 $\boldsymbol{X} = (x_1, x_2, \cdots, x_n)$ 代表系统中各种类组件的数量，N_{total} 为系统中所有组件的数量，那么 $\forall i \in \boldsymbol{N}^+$，$0 < i < n$，种类 i 组件的近似稳态概率

$$\pi_i = \frac{x_i}{N_{\text{total}}} \quad (4)$$

若 Bio-PEPA 对应的隐含标记过程为$\{M(t), t \geq 0\}$，其中每一个状态称为一个序列组件 C_i，i 代表种类，C_{ij} 是 C_i 经过一个反应 j 得到的派生，则所有的 C_{ij} 构成状态空间的状态集合. 在状态空间中，根据状态之间的转移关系可列出变迁速率矩阵 $\boldsymbol{D} = \{d_{ij}\}_{n \times m}$，其中 n 是种类组件的数量，m 是反应的个数，d_{ij} 对应于 C_{ij} 中种类 i 对反应 j 的贡献. 所有反应速率 f_r 构成动能法向量(kinetic law vector)$\boldsymbol{V}_{KL}$，其中 $f_r = r^* \prod_i N_{reaction_i}$，而 $N_{reaction_i}$ 代表本次反应中反应物 $reaction_i$ 的数量. 在此条件下，前述向量 $\boldsymbol{X}$ 满足 ODEs：

$$\left.\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}t}\right|_{X=X_0} = D \times \boldsymbol{V}_{KL} \quad (5)$$

由于篇幅的限制，详细推导和求解过程见文献[12-13]. 此外，Bio-PEPA 不但可以通过式(5)的微分方程形式求解，还可以通过随机模拟或离散马尔可夫链求解.

3 分布式虚拟化系统脆弱性扩散模型

3.1 脆弱点的分簇

系统脆弱性研究是一个广泛的领域,并且由于应用范围和侧重点的不同,脆弱性的定义也不尽相同.其中,最具代表性的是 Bishop 和 Bailey[14] 提出的"computer vulnerability"概念.

定义 1. 脆弱性.脆弱状态是指一个已授权状态,且由该状态经过已授权的转移方式可以到达未授权状态,而脆弱性是指脆弱状态区别于非脆弱状态的特征.

通常将这种具有脆弱性的组件称为脆弱点.造成系统脆弱性的原因具有多种类型,传统上主要包括操作系统设计缺陷、软件 bug 等.此外,连接性(connectivity)、可疑用户输入、不合理信任关系、间谍或恶意软件等因素也会引入额外的脆弱点[15].

分布式虚拟化系统是一类以分布式虚拟化技术为核心技术的通用计算环境[16],典型计算形态包括覆盖网络、云计算系统、未来网络等.分布式虚拟化系统的一项重要特征是虚拟组件与物理资源相分离,构成服务的组件均来自于远程"模糊化"的云中.这些组件将通过动态组合产生满足用户需求的个性化服务,而远程被连接的组件的脆弱性往往很难断定,这就导致脆弱性通过频繁的动态连接而快速扩散.并且随着互联网业务的增长和第三方软件的广泛应用,这类由于连接而产生的脆弱性快速增长,已成为脆弱性诱发的主要原因.因此,在分布式虚拟化环境下,系统脆弱性面临着更为严峻的挑战.

由于虚拟化技术的支撑,分布式虚拟化系统中的所有组件都可以被抽象为"对等"的实体,它们之间通过广义上的连接关系构成系统各类服务.于是,脆弱性就可以是视为在这些对等实体之间的扩散.在本文中我们把这些抽象的对等实体称之为节点.相对于脆弱点产生而言,系统脆弱性扩散仅依赖于节点之间的联系,具有良好的数学统计规律[17].本文将不再讨论已经大量研究的脆弱性产生过程,仅考虑脆弱性在节点间的扩散.

目前许多学者已经采用基于状态的方法对脆弱性扩散规律进行了研究.然而传统脆弱性模型往往在整个系统层面研究脆弱性的扩散,没有区分脆弱性类型造成的扩散差异,或者仅关注脆弱性在同种类型脆弱点之间的扩散,而忽视了不同脆弱点类型之间的扩散.但是对于一个分布式虚拟化系统而言,系统包含的脆弱点种类数以千计,脆弱性在包含不同类型脆弱点的组件之间普遍存在,且传播趋势差别较大,扩散过程往往是不均匀的.为了更加精确的描述和分析这一扩散过程,本文提出把包含同一类脆弱点的节点划分为一个类,称之为一个簇.为研究简便,假设每个节点在确定时刻其中一类脆弱点占主要地位,则按照占主要地位的脆弱点类型决定节点的归属.同时随着时间的演进,由于系统升级、安全策略更改和内外条件变化,节点对应的脆弱点类型发生改变,此时称之为迁移到其他簇.

据已有研究显示,由于安全策略和所处环境的不同,在某一确定时刻,在一个簇内节点可能具备不同的状态.借鉴软件脆弱点状态模型[6]和经典病毒传播 SIR 模型[18],把节点抽象成如下 5 种状态:

(1)W.该类节点虽然目前没有表现出脆弱性,但是具有转化为某类脆弱点的潜在可能,当与处于 V_E 和 V_S 状态的节点连接时可能被渗透而具有脆弱性;

(2)V_E.包含已知脆弱点但尚未被检测出的节点;

(3)V_S.包含已知脆弱点且已被检测出的节点;

(4)F.未及时修复脆弱点,已造成故障/失效的节点;

(5)R.脆弱点修复后的节点.其中,修复方式可以包括简单的断开连接、拒绝服务,以及更加有效的在线升级、打补丁、重配置等方式.

记节点的全部状态集合 $S_0=\{W,V_E,V_S,F,R\}$,而其中 V_E,V_S,F 构成脆弱态集合 $V_{set}=\{V_E,V_S,F\}$.

在上述条件下,可以把节点抽象为一系列 Bio-PEPA 种类组件.为了描述的简便,在建立的 Bio-PEPA 模型中,将处于状态为 $Y\in S_0$ 的节点,称为 Y 组件.下面将分别研究脆弱性在簇内、簇间和迁移条件下的扩散过程.

3.2 由簇内传播导致的脆弱性扩散

相对于在不同脆弱点类型之间的传播,脆弱性通常更易于在具有相同缺陷的节点之间扩散,因此首先考虑最为简单的情况,即脆弱性在同一个簇内的扩散过程.

在任意的簇 location 中,通过连接的脆弱性扩散只发生在 W 组件和 V_S 组件,或者 W 组件和 V_E 组件之间,不考虑对扩散范围无影响的 V_S 和 V_E 组件之间的传播,那么簇内的脆弱性扩散可概括为如下规则.

[Propagation 1]：

(1)$\langle link1\rangle W+V_E\rightarrow 2V_E$

V_E组件通过连接把脆弱性扩散到 W 组件；

(2)$\langle link2\rangle W+V_S\rightarrow V_E+V_S$

V_S组件通过连接把脆弱性扩散到 W 组件；

(3)$\langle discovery\rangle V_E\rightarrow V_S$

V_E组件由于被发现或披露而转化为 V_S组件；

(4)$\langle fail1\rangle V_E\rightarrow F$

V_E组件演化出故障或者错误；

(5)$\langle fail2\rangle V_S\rightarrow F$

V_S组件演化出故障或者错误；

(6)$\langle recovery1\rangle V_E\rightarrow R$

V_E组件被修复；

(7)$\langle recovery2\rangle V_S\rightarrow R$

V_S组件被修复；

(8)$\langle recovery3\rangle F\rightarrow R$

F 组件被修复；

(9)$\langle insecure\rangle R\rightarrow W$

R 组件由于可能仍存在潜在缺陷而转化为 W 组件.

在规则$\langle link1\rangle$与$\langle link2\rangle$描述的脆弱性扩散中，已表现出脆弱性的节点使得与之连接的其他节点表现出脆弱性，但是这种连接可能随时由于连接中断、交互中止、服务取消，或者第三方安全机构的临时补丁等原因而中止，本文把这些连接的中止也归结到修复规则$\langle recovery1\rangle$与$\langle recovery2\rangle$中. 在规则$\langle link2\rangle$中，由于脆弱性扩散的隐蔽性，V_S组件与 W 组件连接默认形成了 V_E组件，V_E可以经过规则$\langle discovery\rangle$再转化为 V_S. 在$\langle insecure\rangle$中，一方面由于连接中止等方式使得部分节点的脆弱点没有得到彻底恢复；另一方面，又如协议的安全性脆弱点不可能在系统局部得到解决，组件可能仍隐含缺陷，所以部分 R 组件将重新转化为 W 组件. 此外，对于 R 组件与 V_S、V_E组件连接进行脆弱性扩散，可以认为先转化为 W 组件，然后进行脆弱性扩散. 需要指出的是，对于修复后 R 组件不能再转化为 W 的组件被视为迁移出该簇，将在 3.4 节中单独讨论.

在簇内扩散中，记扩散集合 $Propagation1=\{link1, link2, discovery, fail1, fail2, recovery1, recovery2, recovery3, insecure\}$. 对于每个反应(扩散规则)，其变迁速率记为 r_α，$\alpha\in Propagation1$. 在此先不考虑节点的加入与退出，即在一定时间段内组件总量是固定的，并记为 N. 在某一时刻 t，组件 W, V_E, V_S, F, R 的个数分别记为 n_W, n_E, n_S, n_F, n_R. 由 Bio-PEPA 的 Mass-Action 语义规则[12]，则反应速率分别满足

$$f_{link1}=r_{link1}n_Wn_E,\ f_{link2}=r_{link2}n_Wn_S,$$
$$f_{discovery}=r_{discovery}n_E,\ f_{fail1}=r_{fail1}n_E,$$
$$f_{fail2}=r_{fail2}n_S,\ f_{recovery1}=r_{recovery1}n_E,$$
$$f_{recovery2}=r_{recovery2}n_S,\ f_{recovery3}=r_{recovery3}n_F,$$
$$f_{insecure}=r_{insecure}n_R.$$

3.3　由簇间传播导致的脆弱性扩散

在具有脆弱点的分布式虚拟化系统中，脆弱性不但会在同种类型的脆弱点之间扩散，有时也会在不同类型脆弱点之间传播. 例如组件 C_A具有一个潜在的访问权限缺陷，但是当前并未表现. 而组件 C_B具有另一种脆弱点类型且已感染恶意软件，当 C_A连接 C_B时，其他组件就可以通过 C_B而获得组件 C_A的高优先级权限，使 C_A表现出脆弱性，就发生了一次脆弱性簇间扩散. 当然，在其他实例中组件 C_A也可能更多地表现出 C_B的脆弱性类型，按照 3.1 节的分簇原则，此时本文认为发生了组件的簇间迁移，将在 3.4 节中进行阐述. 下面将对脆弱性簇间传播的情况进行讨论.

与簇内传播相类似，簇间脆弱性的传播也主要发生在 W 和 V_E、V_S之间. 此外，如果一个脆弱点类型为 A 的簇 $location_A$内 V_E或 V_S组件连接了另一个脆弱点类型为 B 的簇 $location_B$中 V_E或 V_S组件，此时将判断何种类型脆弱点占主导地位，若原来的脆弱点类型 A 占主导，从整体上来看对脆弱性扩散的范围没有产生重大影响，本文中将不再建立单独的规则；如果脆弱点类型 B 占主导将视为已迁移到其他的簇，将在 3.4 节中讨论.

若系统中包含 l 个簇，记簇的集合为 $C=\{location_1, location_2, \cdots, location_l\}$，$|C|=l$，而 $location_i, location_j\in C$，且 $location_i\neq location_j$代表包含不同的脆弱点类型. 则脆弱性簇间传播规则如下.

[Propagation 2]：

(10)$\langle link1_{ij}\rangle$

$W@location_i+V_E@location_j\rightarrow V_E@location_i+V_E@location_j$簇 $location_j$中 V_E组件通过连接把脆弱性扩散到簇 $location_i$中 W 组件；

(11)$\langle link2_{ij}\rangle$

$W@location_i+V_S@location_j\rightarrow V_E@location_i+V_E@location_j$簇 $location_j$中 V_S组件通过连接把脆

弱性扩散到簇 $location_i$ 中 W 组件.

记簇 $location_i$ 到 $location_j$ 的脆弱性扩散的变迁速率分别是 r_{link1_ij}，r_{link2_ij}. $location_i$ 中 W 组件的数量为 $n_{W@location\_i}$，$location_j$ 中 V_E 组件的数量为 $n_{V_E@location\_j}$，根据 Bio-PEPA 的反应规则，其反应速率满足

$$f_{link1\_ij} = r_{link1\_ij} n_{W@location\_i} n_{V_E@location\_j},$$

$$f_{link2\_ij} = r_{link2\_ij} n_{W@location\_i} n_{V_S@location\_j}.$$

若某两个簇之间不存在扩散关系，则令 r_{link1_ij}，$r_{link2\_ij}=0$.

3.4 由簇间迁移导致的脆弱性扩散

除了不同类型的脆弱点之间存在脆弱性传播，由于系统升级、漏洞补丁或者连接其他脆弱点类型等情况，使得占主导地位的脆弱点类型发生改变，可以抽象为节点在簇间的迁移. 例如，系统打补丁后，组件由原来的主要包含 A 种类脆弱点，改为主要包含 B 种类脆弱点. 为了避免问题的复杂化，本文将脆弱性的传播和演化进行区分，以此拆分某些复杂的脆弱性扩散，而簇间迁移导致的脆弱性扩散将仅涉及演化规则. 以图 1 中簇内传播与簇间迁移划分的实例来做进行进一步说明.

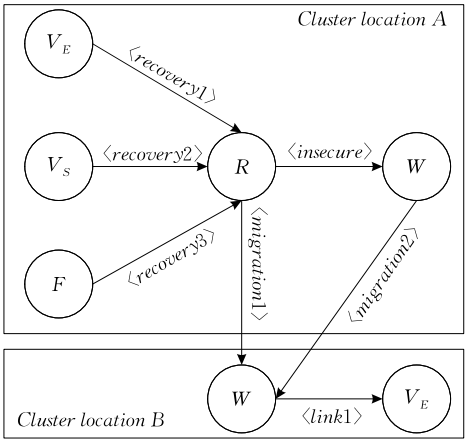

图 1 脆弱性的簇内传播与簇间迁移划分示意图

迁出簇 $location_A$ 中处于 V_S 状态的节点与迁入簇 $location_B$ 中的另一个处于 V_E 状态的节点连接后，占主要地位的脆弱点类型变为 B，这其实是一个复合后的脆弱性扩散，包含簇内传播和簇间迁移. 对于迁入的目的簇 $location_B$ 而言，当簇 $location_A$ 中节点与本簇内节点相连之前，这些节点与本簇内节点完全不同，甚至不是包含潜在脆弱点的 W 组件，因此如果不能发生 3.3 节所述簇间传播，那么这些节点首先必须经过演化转化为本簇的 W 组件才能被扩散到脆弱性. 于是，这个复合的脆弱性扩散过程将包含 3 个阶段. 第 1 阶段，簇 $location_A$ 中 V_S 组件由于环境的改变被暂时修复，脆弱点类型 A 在该节点中将不再占主导地位，即发生规则〈$recovery2$〉；第 2 阶段，修复后的节点状态为 R，发生簇间迁移转变为迁入簇 $location_B$ 的 W 组件；第 3 阶段，在簇 $location_B$ 内发生簇内传播，节点状态由 W 转化为 V_E，即发生规则〈$link1$〉. 因而，仅相当于发生了由迁出簇 $location_A$ 的 R 组件到迁入簇 $location_B$ 的 W 组件的演化. 同理，对于迁出簇 $location_A$ 的 W 组件而言，脆弱点演化发生时从迁出簇的 W 组件转化为迁入簇的 W 组件. 因此，将脆弱性的簇内传播、簇间传播和簇间迁移区分后，簇间迁移的规则主要包含以下两条.

[Propagation 3]：

(12)〈$migration1_{ij}$〉

$W@location_i \rightarrow W@location_j$

簇 $location\_i$ 中 W 组件迁移为簇 $location\_j$ 中 W 组件；

(13)〈$migration2_{ij}$〉

$R@location_i \rightarrow W@location_j$

簇 $location\_i$ 中 R 组件迁移为簇 $location\_j$ 中 W 组件.

若组件 W 和 R 迁移的变迁速率分别为 r_{m1_ij}，$r_{m2\_ij} \in [0,1]$，其中 i 代表迁出的簇，j 代表迁入的簇. 记 $location_i$ 中 W 组件的数量为 $n_{W@location\_i}$，R 组件的数量为 $n_{R@location\_i}$，则簇间迁移的反应速率满足

$$f_{migration1_{ij}} = r_{m1\_ij} n_{W@location\_i},$$

$$f_{migration2_{ij}} = r_{m2\_ij} n_{R@location\_i}.$$

若某两个簇之间不存在簇间迁移关系，则令 r_{m1_ij}，$r_{m2\_ij}=0$.

3.5 脆弱性扩散模型的描述

在系统中，两个簇之间的脆弱性扩散关系包括两种基本类型：簇间传播和簇间迁移，而这两种最基本的类型通过组合又可以构成更为复杂的关系. 若假设任意选取的两种脆弱点类型 A 和 B 是对等的，那么这些组合关系如表 1 所示. 更为一般化，当系统中具有 0 至可列个节点类型的时候，这些脆弱性扩散关系均成立.

在此情况下，假设组件的水平分级 $step\text{-}size=1$，在簇 k 中组件的最大数量为 N_k. 若 $j \neq i$，在 $[0,T)$ 时间段内，$0<T<+\infty$，由确定初始数量的脆弱点诱发的分布式虚拟化系统脆弱性扩散用 Bio-PEPA 描述如下.

表 1　簇间脆弱性扩散关系的种类

序号	关系表示	含义
1	$A\longrightarrow B$	脆弱性由簇 A 传播到簇 B
2	$A\dashrightarrow B$	簇 A 的组件可迁移到簇 B
3	$A\longleftrightarrow B$	脆弱性在簇 A 与簇 B 中可双向传播
4	$A\dashleftarrow\dashrightarrow B$	簇 A 与簇 B 中组件可双向迁移
5	$A\longrightarrow$ $\dashrightarrow B$	脆弱性由簇 A 传播到簇 B，且簇 A 的组件可迁移到簇 B
6	$A\longleftrightarrow$ $\dashrightarrow B$	脆弱性在簇 A 与簇 B 中双向传播，且簇 A 的组件可迁移到簇 B
7	$A\longrightarrow$ $\dashleftarrow\dashrightarrow B$	脆弱性由簇 A 传播到簇 B，且簇 A 与簇 B 中组件可以双向迁移
8	$A\longleftrightarrow$ $\dashleftarrow\dashrightarrow B$	脆弱性簇在 A 与簇 B 中可双向传播，且簇 A 与簇 B 中组件可双向迁移

$$W@location_i \overset{\text{def}}{=} (link1_i,1)\downarrow W@location_i + (link2_i,1)\downarrow W@location_i + (insecure_i,1)\uparrow W@location_i + \sum_j (link1_{ij},1)\downarrow W@location_i + \sum_j (link2_{ij},1)\downarrow W@location_i + \sum_j (migration1_{ij},1)\downarrow W@location_i + \sum_j (migration1_{ji},1)\uparrow W@location_i + \sum_j (migration2_{ji},1)\uparrow W@location_i;$$

$$V_E@location_i \overset{\text{def}}{=} (link1_i,(1,2))\odot V_E@location_i + (link2_i,1)\uparrow V_E@location_i + (discovery_i,1)\downarrow V_E@location_i + (fail1_i,1)\downarrow V_E@location_i + (recovery1_i,1)\downarrow V_E@location_i + \sum_j (link1_{ji},1)\uparrow V_E@location_i + \sum_j (link2_{ji},1)\uparrow V_E@location_i;$$

$$V_S@location_i \overset{\text{def}}{=} (link2_i,(1,1))\odot V_S@location_i + (discovery_i,1)\uparrow V_S@location_i + (fail2_i,1)\downarrow V_S@location_i + (recovery2_i,1)\downarrow V_S@location_i;$$

$$F@location_i \overset{\text{def}}{=} (fail1_i,1)\uparrow F@location_i + (fail2_i,1)\uparrow F@location_i + (recovery3_i,1)\downarrow F@location_i;$$

$$R@location_i \overset{\text{def}}{=} (recovery1_i,1)\uparrow R@location_i + (recovery2_i,1)\uparrow R@location_i + (recovery3_i,1)\uparrow R@location_i + (insecure_i,1)\downarrow R@location_i + \sum_j (migration2_{ij},1)\downarrow R@location_i.$$

在模型中如果某两个簇之间不存在簇间传播或簇间迁移时，则令相应的变迁速率为零. 若任意第 b 个簇中，$1\leq b\leq M$，令 N^0_{Wb}，N^0_{Eb}，N^0_{Sb}，N^0_{Fb}，N^0_{Rb} 分别为初始时刻组件 W、V_E、V_S、F 和 R 组件的数量，且任意 U 组件的初始数量满足 $N^0_{Ub}\leq N_b$，其中 N_b 为第 b 个簇中全部组件的数量. 则模型可以表述为

$$W[N^0_{W1}]\underset{*}{\bowtie}V_E[N^0_{E1}]\underset{*}{\bowtie}V_S[N^0_{S1}]\underset{*}{\bowtie}F[N^0_{F1}]\underset{*}{\bowtie}R[N^0_{R1}]\underset{*}{\bowtie}W[N^0_{W2}]\underset{*}{\bowtie}V_E[N^0_{E2}]\underset{*}{\bowtie}V_S[N^0_{S2}]\underset{*}{\bowtie}F[N^0_{F2}]\underset{*}{\bowtie}R[N^0_{R2}]\underset{*}{\bowtie}\cdots\underset{*}{\bowtie}W[N^0_{WM}]\underset{*}{\bowtie}V_E[N^0_{EM}]\underset{*}{\bowtie}V_S[N^0_{SM}]\underset{*}{\bowtie}F[N^0_{FM}]\underset{*}{\bowtie}R[N^0_{RM}].$$

利用该模型可以对包含多种类型脆弱点，且考虑脆弱点演化的分布式虚拟化系统脆弱性扩散过程进行描述和分析.

4　脆弱性扩散的模拟与分析

分布式虚拟化系统脆弱性扩散模型包含诸多参数，它们对模型的稳定性和合理性存在一定的影响. 本节将选取量化指标，简要分析脆弱性簇间传播速率、簇间变迁速率以及节点恢复能力等关键参数取值对脆弱性扩散过程的影响，并与实际模拟实验进行比较，为模型进一步改进提供参考.

4.1　脆弱性扩散的指标

常见的系统脆弱性指标包括脆弱点存在的概率和范围，例如文献[4]中脆弱点的数量和分布趋势、文献[8]中脆弱性扩散范围. 为了更加精确地刻画脆弱性扩散的趋势，本文参考脆弱性分析和病毒传播领域已有研究成果[8,17-18]，提出脆弱性扩散的两项指标包括：脆弱性峰值指数 P_V，以及脆弱性稳态指数 π_V. 若令 $q\in S_0$ 代表组件种类，$location$ 代表簇，得到定义如下.

定义 2.　脆弱性峰值指数. 所谓脆弱性峰值指数是指在 $0<t<+\infty$ 时刻，脆弱态集合 V_{set} 中各种类组件数量的最大比值，即

$$P_V = \text{Max}\left\{\frac{1}{N_{\text{total}}} * \sum_{location}\sum_{q\in V_{\text{set}}} n_{q@location}(t)\right\} \quad (6)$$

其中，$n_{q@location}$ 代表在簇 $location$ 中组件 q 的数量；N_{total} 是指系统中所有组件数量的总和.

定义 3.　脆弱性稳态指数. 所谓脆弱性稳态指数是指当系统达到稳态后，脆弱态集合 V_{set} 中各种类组件的近似稳态概率之和，即

$$\pi_V = \sum_{location}\sum_{q\in V_{\text{set}}} \pi_{q@location} \quad (7)$$

其中，$\pi_{q@location}$ 代表在簇 $location$ 中组件 q 的近似稳态概率.

P_V 主要用来表征脆弱性扩散的最大范围，而脆弱性稳态指数 π_V 用来表征脆弱性对系统长期的潜在影响. 下面将结合提出的指标，就脆弱性的扩散范围与趋势分析其内在扩散规律.

4.2　脆弱性扩散实例的选取

由于分布式虚拟化系统中任意两个簇之间都可能存在如表 1 所示的 8 种关系，为了抓住问题的核心，减少随机因素干扰和系统特异性带来的额外复

杂性，本文选取一种最为简单的实例，如图 2 所示. 其中 $location_A$、$location_B$、$location_C$ 分别代表包括 3 种脆弱点类型 A、B、C 的组件集合，即 Bio-PEPA 模型的 3 个簇. 其中 $location_A$ 中节点的脆弱性能够传播给 $location_B$ 中节点，$location_B$ 中节点的脆弱性能够传播给 $location_C$ 中节点；同时在一定条件下，脆弱点类型 A 可以演化为脆弱点类型 C，即 $location_A$ 中节点可迁移到 $location_C$. 该实例同时包含簇间扩散、簇间迁移，且有节点同时受到簇间扩散和簇间迁移影响，具有典型性. 需要说明的是，系统可以包含更多的簇，且它们之间的关联关系可以更加复杂，但核心机理与本文选取的实例是相似的.

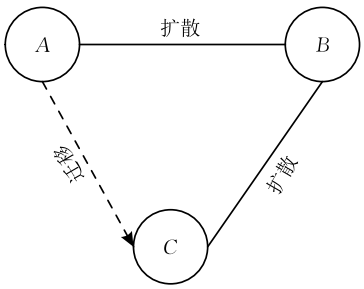

图 2 一个简单的脆弱性扩散实例

在分布式虚拟化环境下，由于虚拟化技术的运用，所有计算资源被组织成资源池的形式. 在理论上，池中的任意资源被选中的概率是等同的，并不会受到具体地理位置和实现形式的影响. 参照文献[8]的假设，不妨设共有 N 个节点，那么选取任意一个节点的概率是 N，即 $r_{link1}=1/N$；同时由于已经表现出脆弱性的节点 V_S 可能感染诸如蠕虫或间谍软件，因此可能会主动发起扫描或连接，不妨假设它的连接机率增加一个数量级 $r_{link2}=10\times 1/N$. 相对于簇内扩散，脆弱性簇间扩散发生的概率更小，簇间扩散的变迁速率取值为簇内扩散变迁速率的 1/10. 为了突出簇间扩散和簇间迁移的影响，减少对比试验的随机干扰，假设 $location_A$、$location_B$、$location_C$ 簇内的扩散参数是相同的，其余各参数的取值如表 2 所示.

表 2 模型中参数的取值

参数	取值	参数	取值
$r_{discovery}$	0.50	r_{link1_12}	$1/10r_{link1}$
$r_{recovery1}$	0.10	r_{link2_12}	$1/10r_{link2}$
$r_{recovery2}$	1.00	r_{link1_23}	$1/10r_{link1}$
$r_{recovery3}$	1.00	r_{link2_23}	$1/10r_{link2}$
r_{fail1}	0.05	r_{m1_13}	1/10
r_{fail2}	0.05	r_{m2_13}	1/10
$r_{insecure}$	0.10		

需要说明的是，由于模型采用了形式化描述的方法，对于任意给定的目标系统(或实例)，均可通过推导得到脆弱性扩散的分析，因而实例的选取不是唯一的. 为了提高求解的效率，本文利用爱丁堡大学开发的 Eclipse Bio-PEPA Plugin 工具包辅助 ODEs 求解[19].

4.3 关键参数对扩散结果的影响

(1) 模型与仿真结果的对比

为了检验模型的合理性，本文首先将提出的模型与采用随机算法模拟的现实系统相比较. 令系统所有组件的总量为 300，设定 3 个簇内组件数量 (n_W,n_E,n_S,n_F,n_R) 的初始值分别为(70,20,10,0,0)、(100,0,0,0,0)、(100,0,0,0,0)，随机算法采用的是 Gillespire's Tau-Leep 随机算法，选取 10 000 个随

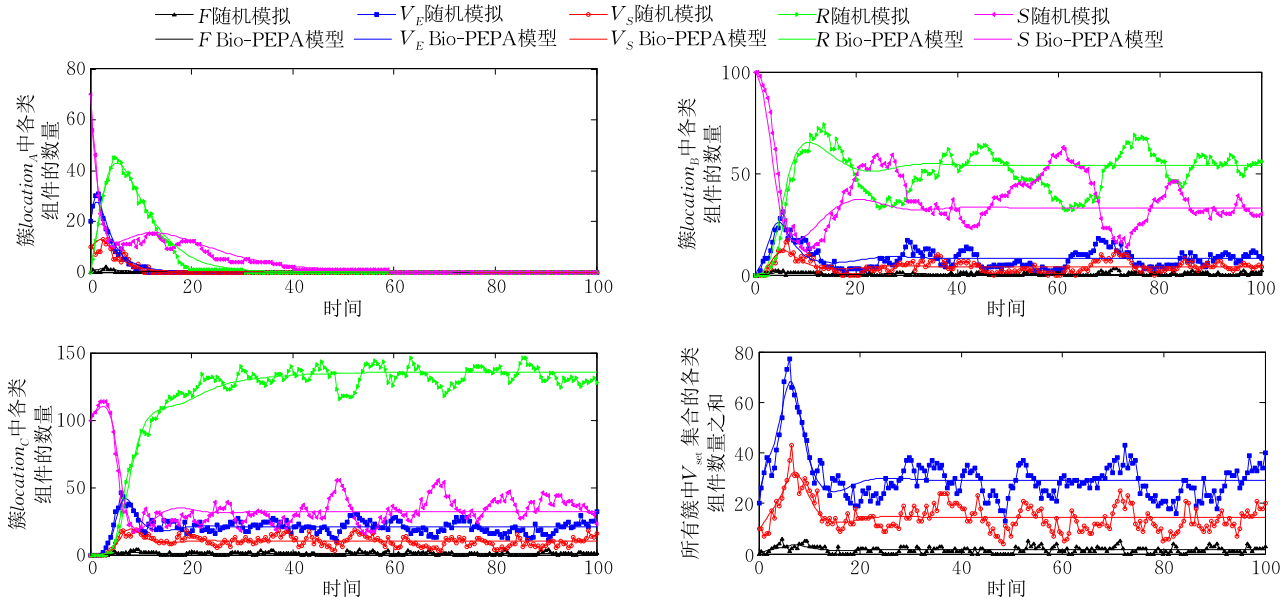

图 3 Bio-PEPA 模型与随机模拟结果的对比

机点，误差设置为 1×10^{-5}. 据此，得到簇 $location_A$、$location_B$、$location_C$ 中的组件，以及系统全局上 V_{set} 集合中组件数量的对比如图 3 所示.

其中图 3(a)、(b)和(c)分别代表在 0 到 100 个时间单位内，在簇 $location_A$、$location_B$、$location_C$ 三个簇中各种组件数量与仿真实验结果的对比. 图 3(a)显示在初始时刻由于脆弱点的存在，脆弱性在簇内迅速扩散，V_E 和 V_S 组件的数量逐渐上升. 此后随着时间推移，由于簇间迁移的存在，$location_A$ 中的节点将逐渐迁移到 $location_C$ 中，V_E 和 V_S 的数量逐渐下降并趋近于 0. 在图 3(b)中，受簇 $location_A$ 脆弱性簇间传播的影响，簇 $location_B$ 中 W 组件逐渐转化为 V_E 和 V_S 组件，随后由于节点修复能力的存在，脆弱性被逐渐修复，大量 V_E 和 V_S 组件转化为 R 组件. 然而因为 V_E 不能被全部发现，所以 V_E 和 V_S 组件将逐渐稳定在一定的水平. 在图 3(c)中，簇 $location_C$ 中的节点受到簇间传播和簇间迁移的双重影响. 由于由簇 $location_A$ 中节点逐步迁移到簇 $location_C$，因而 $location_C$ 中 W 组件逐渐增多，同时受 $location_B$ 的脆弱性簇间传播影响，V_E 和 V_S 组件逐渐增多，此后当这两种组件被修复后转化为 R 组件，于是 R 将逐渐增多. 而图 3(d)则为整个系统范围内脆弱性集合中 V_E、V_S 和 F 组件数量的变化，由于节点修复能力的存在，也是一个先增多而后减少到一个稳定值的过程. 在图 3 中，由于 V_E 和 V_S 组件演化出故障或者错误的概率较小，且系统修复能力较强，组件 F 数量一直处于较低水平.

由图 3 的结果可见，本文模型与模拟结果基本一致，能够反映系统中脆弱性扩散的趋势.

(2) 簇间传播与簇间迁移对脆弱性扩散的影响.

簇间传播是脆弱性扩散的一种重要方式，在本文的模型中主要由簇间传播速率 r_{link1_ij} 和 r_{link2_ij} 进行控制，分别代表了某个簇中处于 V_E 和 V_S 状态的节点对其他簇中连接节点的影响. 在本实例中，r_{link1_ij} 包含 r_{link1_12} 和 r_{link1_23}，而 r_{link2_ij} 包含 r_{link2_12} 和 r_{link2_23}. 为了分析簇间传播对脆弱性扩散的影响，下面将以 P_V 和 π_V 为指标，在整个分布式虚拟化系统层面，考察参数 r_{link1_ij} 和 r_{link2_ij} 取值水平对脆弱性扩散范围和趋势的影响. 当连接速率 r_{link1_ij} 和 r_{link2_ij} 分别扩大 5 倍、扩大 2 倍、缩小为原来的 1/2 和 1/5 时，整个系统范围内 V_{Set} 中组件数量的变化如图 4 所示. 由图可见，随着 r_{link1_ij} 和 r_{link2_ij} 的增大，V_{Set} 中组件数量呈逐渐增长的趋势，并且达到扩散峰值的时间减小.

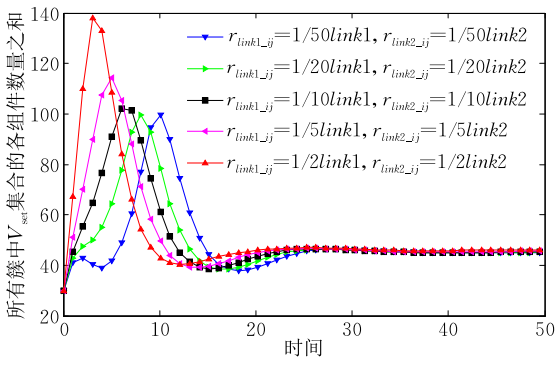

图 4　簇间传播速率对 V_{set} 中组件数量的影响

根据图 4 中数据，由式(6)和(7)分别得到脆弱性扩散的峰值指数 P_V 和稳态指数 π_V，如图 5 所示. 随着连接速率的增长，P_V 的对应值分别为 0.3321、0.3323、0.3402、0.3811、0.4605，即随着连接速率的增大，脆弱性扩散的峰值逐渐变大. 同理，得到 π_V 分别为 0.1502、0.1505、0.1509、0.1516 和 0.1535，可见最终脆弱性扩散的范围基本相同. 发生该现象的根本原因在于，随着连接速率的增大，簇 $location_A$ 与簇 $location_B$ 节点交互更为紧密，发生脆弱性传播的概率增大，因而扩散的峰值增大且时间缩短. 但是由于节点数量一定，而节点修复能力不变，簇中感染脆弱性的节点基本保持在一定的水平. 由此说明簇间传播速率与扩散峰值正向相关，所以为了抑制系统的脆弱性需要降低簇间传播速率.

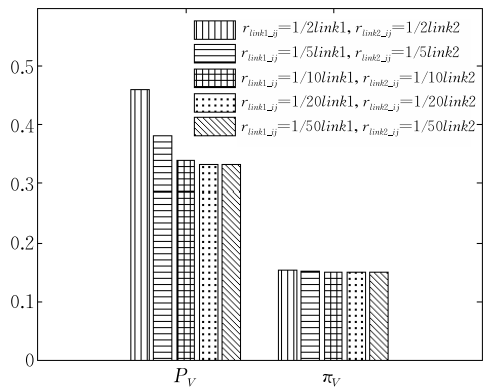

图 5　簇间传播速率对脆弱性峰值指数 P_V 和稳态指数 π_V 的影响

与此相对应，簇间迁移是脆弱性扩散的另一种重要方式，在本文的模型中主要由簇间变迁速率 r_{m1_13} 和 r_{m2_13} 进行控制，分别代表了由原簇中处于 W 和 R 状态的节点经过演化转化为新脆弱点类型的概率. 为了分析簇间迁移对脆弱性扩散的影响，考察 r_{m1_13} 和 r_{m2_13} 的取值水平对簇 $location_C$ 内脆弱性扩散范围和趋势的影响. 当把 r_{m1_13} 和 r_{m2_13} 分别改变为 1/50、1/10、1/2 时，在簇 $location_C$ 内 V_{set} 中组

件数量如图 6 所示. 由图中可见随着簇间变迁速率的增大,V_E 和 V_S 组件达到同一水平所需要的时间缩短,加速了扩散的过程.

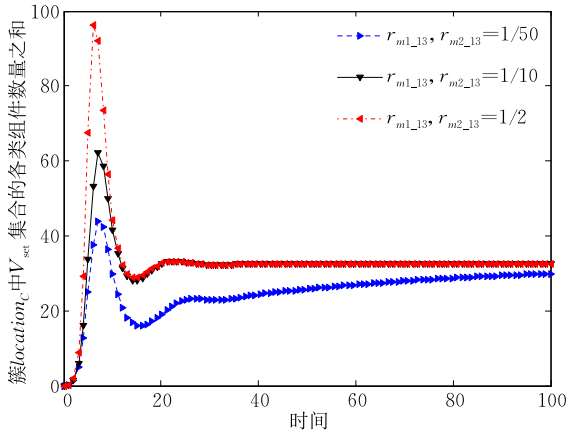

图 6 簇间变迁速率对簇 $location_C$ 中 V_{set} 组件总数的影响

同理,可以得到在整个分布式虚拟化系统层面的脆弱性扩散的峰值指数 P_V 和稳态指数 π_V,如图 7 所示. 随着变迁速率的增大,脆弱性扩散的峰值逐渐变大,而最终脆弱性扩散的范围基本相同. 究其原因,变迁速率的增大使得簇 $location_A$ 中的节点,甚至是已经被修复的节点迁移到了簇 $location_C$ 内,因而簇 $location_C$ 内的潜在脆弱性节点增多,更加容易产生脆弱性簇内扩散. 但是由于整体上节点数量一定,而节点修复能力不变,簇中具备脆弱性的节点基本保持在某个确定水平. 由此说明簇间变迁速率与扩散峰值正向相关,所以为了抑制系统的脆弱性需要降低簇间变迁速率.

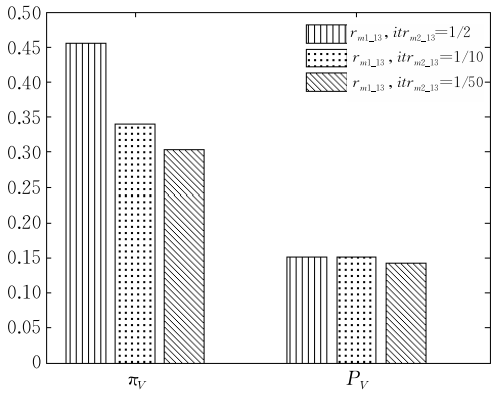

图 7 簇间变迁速率对脆弱性峰值指数P_V和稳态指数π_V的影响

(3) 修复能力对脆弱性扩散的影响

对于目标系统而言,修复能力的存在可以使失效节点恢复,有效地降低 V_E 和 V_S 的水平,对脆弱性的抑制具有重要意义. 下面将以 P_V 和 π_V 为指标,通过令 $r_{recovery1}$ 和 $r_{recovery2}$ 分别在区间[0.1,1.0]之间 9 等分取值来分析修复能力对脆弱性扩散的影响,得到结果分别如图 8 和图 9 所示.

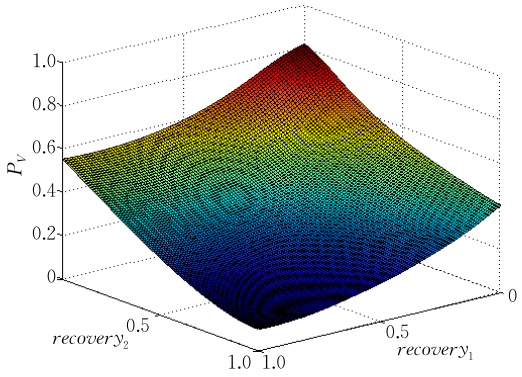

图 8 修复能力对脆弱性峰值指数 P_V 的影响

由图 8 和图 9 可见,随着 $r_{recovery1}$ 和 $r_{recovery2}$ 取值的增大,P_V 和 π_V 逐渐减小,并且最终趋势变得平缓. 发生该现象的原因主要是,当 $r_{recovery1}$ 和 $r_{recovery2}$ 增大时,意味着系统具有更强大的修复能力,V_E 和 V_S 组件将以更大概率被修复,因而脆弱性扩散的峰值和范围都有所减小. 此外,通过对比来看,当 $r_{recovery2}$ 和 $r_{recovery1}$ 减小同等幅度时,前者带来的 P_V 和 π_V 的增长更为剧烈. 这主要是由于 V_E 最终也将转化为 V_S,V_S 修复能力的增加可以更加有效地抑制脆弱性. 因此,为了降低扩散的最大范围,尤其是脆弱性最终的稳态概率,需要增强节点的修复能力.

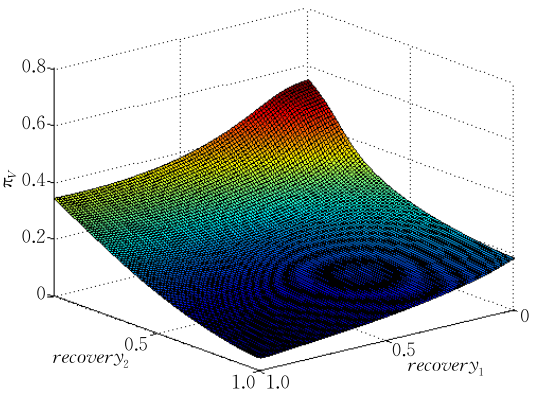

图 9 修复能力对脆弱性稳态指数 π_V 的影响

(4) 模型求解开销分析

由于分布式虚拟化系统往往包含大量的节点并分别属于众多种类,因此脆弱扩散性模型能够准确快速地求解具有重要现实意义.

根据本文模型,对于一个包含 l 种脆弱点类型的分布式虚拟化系统而言,任意的簇 $location_k$ 包含的节点数最多为 N_k,若采用文献[9,10]的方法都等价于求解一个隐含的马尔可夫过程,则根据传统的基于状态的求解方法,其状态空间未化简前包含的

状态数量为

$$\prod_{k=1}^{l} 5^{N_k} = 5^{\sum_{k=1}^{l} N_k}.$$

即使在状态空间化简后，组件数量也将达到，

$$\prod_{k=1}^{l} \begin{pmatrix} N_k - 1 \\ 5 + N_k - 1 \end{pmatrix}.$$

求解时将面临状态空间爆炸的问题. 与之相对，本文采用了 Bio-PEPA 转化为 ODEs 的途径，方程个数仅与脆弱点类型相关而与每种类型包含的组件数量无关[13]. 以本文选取的实例为例子，采用马尔可夫过程、随机模拟和 ODEs 途径时的求解时间分别如表 3 所示. 其中，初始条件下各个组件所占比例与本文 4.2 节中实例遵循相同比例，随机算法仍采用 Gillespire's Tau-Leep 随机算法，取 10 000 个随机点，迭代 1000 步. 而运行环境为 Windows XP，CPU 为 4 核 2.0 GHz，内存为 4 GB.

表 3　模型求解开销对比

组件总共数量	马尔可夫过程方法				随机模拟方法耗时/s	ODEs 方法耗时/s
	化简前的状态空间包含状态数量	求解时间/s	化简后的状态空间包含状态数量	求解时间/s		
30	9.3×10^{20}	—	8.0×10^{9}	32	0.12	0.20
60	8.6×10^{41}	—	7.7×10^{13}	—	0.28	0.23
300	4.9×10^{209}	—	7.8×10^{23}	—	0.56	0.23
3000	8.0×10^{2096}	—	6.0×10^{38}	—	3.30	0.23
30 000	1.0×10^{4202}	—	5.8×10^{53}	—	23.00	0.25

注："—"代表所需计算资源已超出了本文实验环境的能力.

因此，采用本文的方法可以有效地克服分布式虚拟化系统脆弱性扩散模型求解的状态空间爆炸问题.

综合上述分析，不难发现本文提出的模型与仿真结果基本一致，可以提供定量的分布式虚拟化系统脆弱性扩散分析，能够合理地解释脆弱性变化的趋势. 而通过参数效应的分析，可以得到各关键参数对本文提出模型的影响，具体包括：

(1) 当系统中节点的修复能力较强时，将有效地减少脆弱点的数量；

(2) 簇间传播速率与脆弱性扩散峰值正向相关，能够减少扩散需要的时间，但对最终稳定后的扩散范围影响不大；

(3) 簇间变迁速率的增大将加快脆弱点在簇间的迁移(演化)过程，进而间接加快脆弱性的扩散.

综上所述，需要提升系统的脆弱性修复能力，遏制和阻止不同类型脆弱点之间的扩散，理清脆弱点之间的演化关系，并针对性地进行预防.

5　国内外相关研究工作

随着覆盖网络、云计算以及未来网络技术的快速发展，分布式虚拟化系统的应用范围日益广泛，而由于虚拟化技术的支撑，其不但规模巨大、而且具有显著地动态性、移动性和开放性. 这些特点导致了分布式虚拟化系统中由连接而引入的脆弱性大大增加，在数量上已远远超过了系统固有脆弱性缺陷，且随时可能发生并快速扩散，已成为亟待解决的挑战[20]. 而对脆弱性扩散规律的认识有助于预防、抑制和阻止系统脆弱性的漫延，以较小的代价提高系统的安全性，是目前脆弱性分析的发展方向.

目前许多学者已开始研究脆弱性的演化、传播和扩散规律，为脆弱性的防护和修复提供了参考. 这些研究主要集中在如下 3 个方面：第 1 方面，主要是基于运行日志对特定系统脆弱性扩散的事后分析. 英国剑桥大学的 Ozment[4] 在对现有脆弱性发现模型分析的基础上，利用 OpenBSD 操作系统八年的脆弱性数据，对该系统脆弱性的演化规律进行了总结. Neuhaus 等人[5] 则通过自动化的软件挖掘 Mozilla 系统的现有脆弱点数据库，把所有已知脆弱点映射到组件上，并在此基础上对 Mozilla 脆弱性的发展进行了预测. Hiller 等人[6] 分析了数据错误在软件中的传播，讨论了数据错误在关联软件模块之间的扩散行为，并提出了设定检查点和恢复点的定位方法. 上述研究主要基于已有系统日志的统计和分析，缺少理论化的推理模型，具有很强的系统相关性，并且很难在软件或系统的设计阶段来指导脆弱性的防治问题. 同时，在分布式虚拟化系统中，由于虚拟化技术的存在，计算节点之间的连接是一个概率性事件，已有系统日志难以覆盖所有可能的连接，因而这类研究具有一定局限性. 第 2 方面，主要是系统整体层面上的脆弱性扩散研究. 针对信息系统的风险管理，Feng 等人[7] 建立了一种安全性风险模型(SRAM)，基于贝叶斯网络来模拟风险因子及它们之间的因果关系，并通过蚁群算法和专家知识估计最大概率的脆弱性传播路径以及带来的风险值. De 等人[8] 针对无线传感器网络环境下，由于多跳广播协议带来的脆弱性扩散问题进行研究，并基于该协议通过间谍软件的帮助，讨论了在不同的动作变迁速率、连接性、可恢复性时脆弱性的传播过程，对本文的参数分析具有重要的借鉴意义. 但是这

些工作都没有关注脆弱点类型之间的区分,没有考虑脆弱性在不同脆弱点类型之间扩散的差异,以及脆弱点的演化问题.第 3 方面,主要是可应用于系统设计阶段的脆弱性扩散分析模型.为了在软件设计阶段减少脆弱性,文献[9]提出了一种计算脆弱性传播的算法,在面向对象设计中采用脆弱性树的方式测量属性脆弱比率(Attribute Vulnerability Ratio, AVR),主要解决了由于代码重用造成脆弱点在旧有代码与新代码之间的传播问题.在此基础上,文献[10]则研究了在面向对象设计阶段由类的耦合而引入的脆弱性扩散,并采用脆弱性树和 ATM 进行了分析.虽然在这些研究中区别了脆弱点类型的不同,分别建立了脆弱性树,但是仅仅考虑了脆弱性在同种类型脆弱点之间的传播,而没有考虑在不同脆弱性树之间传播的问题.同时,对于分布式虚拟化系统而言,其巨大的规模导致对象类的数目可能数以百万计,此时利用脆弱性树或者 ATM 进行状态空间搜索,可能将面临状态空间爆炸问题.此外,针对脆弱性分析的模型还包括基于标签转移图、基于着色 Petri 网、攻击图模型、需求/产出和模型检测等多种类型[17],但是当系统规模巨大时也面临着状态空间爆炸的问题.

综上所述,与现有研究相比,本文的主要改进之处如下:

(1)虽然已有部分模型将不同的脆弱点分类[9-10],但通常仅仅研究脆弱性在同种类型脆弱点之间的扩散,即假定扩散是均匀地,而忽略了脆弱性在不同类型脆弱点之间传播的差异,而本文考虑了脆弱性在同种脆弱点类型、不同类型,以及脆弱点演化情况下的脆弱性扩散.

(2)在分布式虚拟化系统中,服务所用资源处于远程"模糊化"的云中,无法事先预测连接的对象,据此本文给出了一种形式化的脆弱性扩散模型,节点之间的连接基于概率化表示,适合于分布式虚拟化系统脆弱性扩散的特点.

(3)采用的 Bio-PEPA 形式化描述可以转化为 ODEs,与传统分析方法相比,具有更低的时间开销,为应用于如云计算等大型分布式虚拟化系统提供了条件.

6 结 论

随着分布式虚拟化系统的广泛应用,由第三方应用引入的脆弱点在数量上已远远超过系统固有的脆弱点,同时服务使用的资源往往来自于远程"云"中的不同位置,通过频繁的动态连接而使脆弱性快速扩散,因此脆弱性扩散已成为亟待解决的挑战.本文提出了一种分布式虚拟化系统的脆弱性扩散模型,利用 Bio-PEPA 的静态分层特性描述了脆弱性在相同脆弱点类型、不同脆弱点类型之间传播以及由脆弱点演化带来的脆弱性扩散过程,为分布式虚拟化系统的脆弱性扩散规律研究提供了参考.实验结果显示,模型与仿真模拟的结果基本相符合,可以通过提升系统的脆弱性修复能力、降低簇间传播速率、减少簇间变迁速率,遏制脆弱性的扩散.此外,通过把 Bio-PEPA 转化为 ODEs,可以避免传统分析方法的状态空间爆炸问题,适用于大规模和高动态分布式虚拟化系统的脆弱性扩散研究.

在下一步的工作中,我们计划研究有脆弱点随机加入退出的扩散模型,以处理更加复杂的情况,并针对实际的分布式虚拟化系统进行验证和改进.

致 谢 在此,我们向对本文的工作给予支持和宝贵建议的各位评审专家、编辑表示衷心感谢!

参 考 文 献

[1] Lowis L, Accorsi R. Vulnerability analysis in SOA-based business processes. IEEE Transactions on Services Computing, 2011, 4(3): 230-241

[2] Jula A, Sundararajan E, Othman Z. Cloud computing service composition: A systematic literature review. Expert Systems with Applications, 2014, 41(8): 3809-3824

[3] Vu H L, Khaw K K, Chen T Y. A new approach for network vulnerability analysis. The Computer Journal, 2015, 58(4): 878-891

[4] Ozment A. Vulnerability Discovery & Software Security [Ph. D. dissertation]. University of Cambridge, London, 2007

[5] Neuhaus S, Zimmermann T, Holler C, et al. Predicting vulnerable software components//Proceedings of the Computer and Communications Security 2007. Alexandria, USA, 2007: 529-540

[6] Hiller M, Jhumka A, Suri N. EPIC: Profiling the propagation and effect of data errors in software. IEEE Transactions on Computers, 2004, 53(5): 1-19

[7] Feng N, Wang H J, Li M. A security risk analysis model for information systems: Causal relationships of risk factors and vulnerability propagation analysis. Information Sciences, 2014, 256: 57-73

[8] De P, Liu Y, Das S. An epidemic theoretic framework for vulnerability analysis of broadcast protocols in wireless sensor networks. IEEE Transactions on Mobile Computing, 2009, 8(3): 413-425

[9] Agrawal A, Khan R A. Impact of inheritance on vulnerability propagation at design phase. ACM SIGSOFT Software Engineering Notes, 2009, 34(4): 1-5

[10] Agrawal A, Khan R A. Role of coupling in vulnerability propagation-object oriented design perspective. Software Engineering: An International Journal (SEIJ), 2012, 2(1): 60-68

[11] Groote J F, Kouters T, Osaiweran A. Specification guidelines to avoid the state space explosion problem. Software Testing, Verification and Reliability, 2015, 25(1): 4-33

[12] Ciocchett F, Hillston J. Bio-PEPA: A framework for the modelling and analysis of biological systems. Theoretical Computer Science, 2009, 410(33-34): 3065-3084

[13] Galpin V. Hybrid semantics for Bio-PEPA. Information and Computation, 2014, 236: 122-145

[14] Bishop M, Bailey D. A critical analysis of vulnerability taxonomies. University of California, Davis: Technical Report CSE296211, 1996

[15] Feng Ping-Hui, Lian Yi-Feng, Dai Ying-Xia, Bao Xu-Hua. A vulnerability model of distributed systems based on reliability theory. Journal of Software, 2006, 17(7): 1633-1640 (in Chinese)
(冯萍慧，连一峰，戴英侠，鲍旭华. 基于可靠性理论的分布式系统脆弱性模型. 软件学报，2006，17(7)：1633-1640)

[16] Chowdhury N M, Boutaba R. A survey of network virtualization. Computer Networks, 2010, 54(5): 862-876

[17] Xing Xu-Jia, Lin Chuang, Jiang Yi-Xin. A survey of computer vulnerability assessment. Chinese Journal of Computers, 2004, 27(1): 1-11(in Chinese)
(邢栩嘉，林闯，蒋屹新. 计算机系统脆弱性评估研究. 计算机学报，2004，27(1)：1-11)

[18] Bradley J T, Gilmore S T, Hillston J. Analysing distributed Internet worm attacks using continuous state-space approximation of process algebra models. Journal of Computer and System Sciences, 2008, 74: 1013-1032

[19] Ciocchetta F, Duguid A, Gilmore S. The Bio-PEPA tool suite//Proceedings of the 6th International Conference on the Quantitative Evaluation of Systems. Eger, Hungary, 2009: 309-310

[20] Ali M, Khan S U, Vasilakos A V. Security in cloud computing: Opportunities and challenges. Information Sciences, 2015, 305(1): 357-383

LV Hong-Wu, born in 1983, Ph. D., lecturer. His major research interests include availability, cloud computing and performance evaluation.

WANG Hui-Qiang, born in 1960, Ph. D., professor. His major research interests include network security, future network and dependability.

LIN Jun-Yu, born in 1981, Ph. D., assistant researcher. His major research interests include network security and vulnerability analysis.

FENG Guang-Sheng, born in 1980, Ph. D., lecturer. His major research interests include network security and cognitive network.

GUO Fang-Fang, born in 1974, Ph. D., associate professor. His major research interests include network security situation awareness and cloud monitor.

Background

Vulnerability is usually the essential reason of security and dependability. And all the researches of vulnerability are generally known as vulnerability analysis, which is always a hot topic in the domain of software development and system test. Recently, for distributed virtualized systems, such as overlay network, cloud computing system and future network, there are enormous amounts of third-party applications now than ever before, which have more defects and holes than the operation system and the original software. Thus plenty of additional vulnerabilities are brought out by interactions and unreasonable trust relationship among nodes in distributed virtualized systems. Moreover, it is reported that the number of these new additional vulnerabilities is even more than the inherent ones. Vulnerability propagation has grown up to be a serious problem.

In the field of vulnerability propagation and forecast, a preliminary study on the process of vulnerability propagation of WSN, object-oriented program and some information systems has been carried out. However, these traditional models ignored too much information about vulnerabilities, and there are mainly the following three aspects of shortcomings.

(1) The vulnerability propagation process is usually

treated as uniform distribution, that is, the differences of vulnerability propagation between different types of vulnerabilities are ignored.

(2) For a distributed virtualized system, all the computing sources stay in the remote cloud owing to the technology of virtualization. It is impossible to predict the vulnerability of the connection object beforehand, and it may choose anyone in the virtual resource pool. Thus it is difficult to accurately describe this random selection based on traditional log analysis methods.

(3) For the huge number of nodes in distributed virtualized systems, the analyzer may suffer from a problem of state space explosion while the traditional vulnerability propagation models based on state-transition are solved.

According to the existing shortcomings, we propose a new vulnerability propagation model for distributed virtualized systems. In this model, the same kind of vulnerabilities are seen as in a single cluster, and then the vulnerability propagation in/between clusters as well as vulnerabilities migration between clusters is modeled by Bio-PEPA (Performance Evaluation Process Algebra) in a static hierarchy manner, avoiding the state space explosion existing in traditional analysis methods.

This work is partially supported by the National Natural Science Foundation of China under Grant Nos. 61402127 and 61370212, and the Natural Science Foundation of Heilongjiang Province under Grant No. F2015029. The aims of these projects are to improve service availability of distributed virtualized system. Until now, the research team has published more than 50 papers about security and dependability of distributed systems. And the purpose of this paper is to study the law of vulnerability propagation, which is useful to prevent and restrain vulnerability and enhance the dependability of distributed virtualized systems.