计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190800051

可抵御内部威胁的角色动态调整算法

潘　恒[1]　　李景峰[2]　　马君虎[3]
1 中原工学院前沿信息技术研究院　郑州 450007
2 战略支援部队信息工程大学　郑州 450001
3 空军 93010 部队　沈阳 110016
(panheng@zut.edu.cn)

摘　要　业务流程、信息基础设施等的变化会造成当前角色定义出现偏差,使得组织易遭受内部威胁。基于定时、合理改变组织内部角色集合的防御思路,提出了一种角色动态调整算法(Role Dynamic Ajusting,RDA)。该算法定义了带有调整参数的目标函数,以平衡考虑用户权限实际使用数据以及系统管理员专家知识;基于启发式搜索策略和子集结对操作得到一组候选角色;使用启发式函数计算角色分值,按照角色分值的高低对候选角色集进行删选,得到符合角色质量要求的调整角色集;以降低角色冗余度为目标,使用调整角色集为系统用户重新分配角色,得到新的系统角色配置。基于某医院管理系统日志的实验结果表明,RDA 算法可有效调节目标组织系统的角色集,为抵御内部威胁打下良好基础。

关键词:内部威胁;基于角色的访问控制;启发式搜索策略;一类支持向量机;动态调整
中图法分类号　TP191

Role Dynamic Adjustment Algorithm for Resisting Insider Threat

PAN Heng[1],LI Jing-feng[2] and MA Jun-hu[3]
1 Research Institute of Advanced Information Technology,Zhongyuan University of Technology,Zhengzhou 450007,China
2 PLA Information Engineering University,Zhengzhou 450001,China
3 PLA Air Force 93010 Unit,Shenyang 110016,China

Abstract　Due to derivations in current role definition from the changes of the bossiness process and information infrastructure, organizations are vulnerable to internal threat. A role dynamic adjustment algorithm is proposed based on the defensive idea of changing the set of roles within the organization regularly and reasonably. The algorithm defines an objective function with adjusting parameters to balance the two elements, which are the user privilege actual use data and the system administrator expert knowledge. Based on heuristic search strategy and sub-set pairing technique, a group of candidate roles are obtained. From these roles, a set of adjusting roles which can achieve a predefined score are obtained, by using a certain heuristic function. Finally, in order to reduce role redundancy, the users of the organization are reassign the roles from the adjusting roles, so getting a new Role-Based Access Control(RBAC) configuration. By using the audit logs from a hospital management system, the performance of the RDA is analyzed. The experimental results show that the proposed algorithm can efficiently adjust the RBAC configuration for the special organization, so it can provide concrete base for resisting the insider threats.

Keywords　Insider threats, Role-based access control, Heuristic search strategy, One-class support vector machine, Dynamic adjustment

1　引言

近年来,内部威胁已经成为了信息社会面临的最严重的安全挑战之一[1]。组织内员工滥用自身权限时很容易发生内部威胁,对组织造成巨大的声誉损失和经济损失,甚至导致战略被动。因此,无论是政府部门,还是私营企业,都迫切需要借助现代信息技术来构建内部威胁防御系统。

在为目标组织设计内部威胁防御系统时,设计者必须重点考虑两个问题。(1)何种解决策略更符合目标组织?(2)选定解决策略后,使用哪些特殊方法才能实现该策略?对于问

到稿日期:2019-08-12　返修日期:2019-10-28　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:河南省高等学校重点基础研究计划项目(19A520047);中原工学院自主创新应用研究项目(K2018YY017)
This work was supported by the Foundation of Priority Fundamental Research Project of Institutions of Higher Education of Henan Province, China (19A520047) and Foundation of Independent Innovation Application Research of Zhongyuan University of Technology, China (K2018YY017).
通信作者:李景峰(lee_jingfeng@163.com)

题(1),内部威胁解决策略可分为两类。第一类是预期性策略,即在用户请求发生时就做出拒绝或者同意的决策。大多数访问控制系统都属于这类,如强制访问控制(Mandatory Access Control,MAC)、自主访问控制(Discretionary Access Control,DAC)和基于角色访问控制(Role-based Access Control,RBAC)等。第二类是追溯性策略(Retrospective Strategy),即允许先进行用户访问,事后再对访问行为进行核查[2]。对于问题(2),现有解决方案多依赖专家知识和专家经验,对海量访问日志数据背后隐藏信息的分析不足,因此不能更好地指导防御系统的精细设计。

RBAC是目前应用最为广泛的预期性解决策略,可对内部威胁防御系统起到重要的支撑作用。RBAC将角色作为用户-权限分配的桥梁。一个完整、准确、有效的角色集,是RBAC能够发挥防御效用的前提。然而,即使使用Coyne等提出的"角色工程"(Role Engineering)方法[3]定义出了自认为符合组织要求的初始角色集,这个角色集也会因为组织业务流程变化、内部人员调动、信息技术进步等因素而必须适时做出调整,否则就会使系统角色存在被内外部攻击者利用的隐患[4-5]。一般而言,角色调整可以在目标组织业务流程发生变化,中高层人员出现变动,信息系统基础设施更新改造等时机进行。

针对上述问题,本文基于定时、合理改变组织内部角色集合的基本思路,提出了一种角色动态调整算法——RDA算法。首先,综合考虑目标系统中用户权限实际使用数据以及系统管理员专家知识,使用动态搜索策略和集合结对技术来获取候选角色集合;然后,使用特定启发式函数计算角色分值,并按照角色分值的高低对候选角色集中的角色进行排序和选择,从而得到符合角色质量要求的调整角色集;最后,以降低角色冗余度为目标,使用调整角色集为用户重新分配角色,从而得到新的RBAC配置,其中包含新的用户角色关系和角色权限关系。使用RBAC配置间距离和RBAC异常用户率等作为评价指标,来评判调整后系统角色集的合理性。RBAC配置间距离基于角色同质性计算获得;RBAC异常用户率由基于RBF内核的一类SVM进行学习和衡量。最后,使用来自某医院EMR系统的访问日志数据对所提算法的性能进行实验分析,结果表明RDA算法能够对目标系统角色集进行有效调节。

2 基本概念

本文将研究的系统角色动态调整问题看作广义角色挖掘问题,本节首先给出该问题的形式化定义。

2.1 广义角色挖掘问题

定义1(广义角色挖掘问题[6]) 设 $t=\langle U,P,\boldsymbol{UPA}\rangle$ 为一个访问控制配置,其中,$U=\{u_1,u_2,\cdots,u_m\}$ 表示用户集合;$P=\{p_1,p_2,\cdots,p_n\}$ 表示权限集合;$\boldsymbol{UPA}$ 是 $m\times n$ 的布尔矩阵,是 U 到 P 的一个映射。

根据给定角色 r_l,可得到对应的用户和权限。符号 $\boldsymbol{P}_l^{(c)}=\{p_x\mid \boldsymbol{RPA}_{lx}=1\}$ 代表分配给 r_l 的权限集合。符号 $\boldsymbol{U}_l^{(c)}=\{u_y\mid \boldsymbol{URA}_{yl}=1\}$ 代表配置 c 中被分配了 r_l 的角色集合。按照经典RBAC文献[7],在不引起混淆的条件下,本文使用权限的集合来表示角色。例如,$\gamma=\{p_1,p_2\}$ 表示角色 γ 当前拥有 p_1 和 p_2 两种权限。

2.2 目标函数的设计

目标函数 $f()$ 很关键[8-9],在设计目标函数时,本文考虑在系统管理员专家经验和用户调用系统权限的实际情况之间进行平衡。为方便用户随时调整这两种因素的比重,最好将目标函数设置为参数可调的。

2.2.1 用户权限调用要素

本文在进行角色的动态调整时,将考虑用户使用系统时的实际权限调用情况,即使用系统访问日志中的用户权限调用数据来构建一个 $m\times n$ 的"用户-权限调用矩阵"***UPIM***。该矩阵的每个元素 ω_{ij} 代表用户 u_i 调用权限 p_j 的次数。

一般来说,多位用户的行为越相似,那么这些用户被分配同一角色的可能性就越大。但是,在初始定义系统角色时被分配了同一角色的多位用户,在实际使用系统时的调用权限行为出现了较大差别。如果其中某个用户的行为与其他用户的行为差别过大,那么就称该用户为该角色的"异常用户"。为检测异常用户,需要使用投影矩阵提供的行向量信息。

定义2(投影矩阵) 给定 c 及其对应的***UPIM***,角色 r_l 的投影矩阵 $\boldsymbol{M}_{r_l}$ 是一个 $p\times q$ 矩阵。其中,$p=|\boldsymbol{U}_l^{(c)}|$,$q=|\boldsymbol{P}_l^{(c)}|$。

$\boldsymbol{M}_{r_l}$ 的行向量代表被分配了角色 r_l 的一位用户;列向量代表分配给角色 r_l 的一种权限。$\boldsymbol{M}_{r_l}$ 中的每个元素 β_{ij} 定义如下:如果 $\boldsymbol{U}_l^{(c)}$ 中第 i 位用户是 $u_f\in U$,并且 $\boldsymbol{P}_l^{(c)}$ 中第 j 种权限是 $p_g\in P$,那么 $\beta_{ij}=\omega_{fg}$。

为刻画被分配了同一角色的多位用户的行为相似程度,本文提出了"角色同质性"概念。

定义3(角色同质性) 给定 c 和用户-权限调用矩阵***UPIM***,则单一 r_l 的角色同质性为:

$$RH(r_l)=m^{-1}\sum_{i=1}^{m}(1-\mathrm{cosine}(\boldsymbol{x}_i,\boldsymbol{c}_l)) \tag{1}$$

其中,m 是 $\boldsymbol{M}_{r_l}$ 中行向量的数量,x_i 是 $\boldsymbol{M}_{r_l}$ 中的第 i 个行向量,$\boldsymbol{c}_l$ 是 $\boldsymbol{M}_{r_l}$ 中全部行向量的平均向量,$\mathrm{cosine}(\boldsymbol{a},\boldsymbol{b})$ 用于计算余弦相似度 $(\boldsymbol{a}\cdot\boldsymbol{b})/|\boldsymbol{a}||\boldsymbol{b}|$。$RH(r_l)$ 越大,说明角色同质性越高,组织设置该角色就越合理。

基于单一角色的角色同质性,就可以定义RBAC配置 c 的"RBAC同质性"为:

$$h(c)=|R|^{-1}\sum_{r_l\in R}RH(r_l) \tag{2}$$

同样,$h(c)$ 越大,说明RBAC配置 c 越符合组织角色结构要求。

2.2.2 管理员专家经验要素

一般而言,目标组织最初的角色集都是系统管理员根据自身经验定义的[5-7],其在很大程度上能够很好地符合组织业务流程的需求。为防止角色动态调整时造成角色集合剧烈变化,从而影响组织业务的平稳运转及信息系统的稳定运行,在设计目标函数时,本文重点考虑了系统管理员的专家经验。具体来说,这是通过将角色动态调整前后的新旧RBAC配置之间的距离限制在一定范围来反映的。

首先,给出任意两个角色之间距离的定义。

定义4(角色间距离) 设 γ_i 和 δ_j 分别是RBAC配置 c_1 和 c_2 中的一个角色,它们之间的角色距离为:

$$jac(\gamma_i,\delta_j)=1-\frac{|(\boldsymbol{P}_i^{(c_1)}\times\boldsymbol{U}_i^{(c_1)})|\bigcap|(\boldsymbol{P}_j^{(c_2)}\times\boldsymbol{U}_j^{(c_2)})|}{|(\boldsymbol{P}_i^{(c_1)}\times\boldsymbol{U}_i^{(c_1)})\bigcup(\boldsymbol{P}_j^{(c_2)}\times\boldsymbol{U}_j^{(c_2)})|} \tag{3}$$

其中，$A\times B$ 是 A 和 B 的笛卡尔乘积。换言之，一个角色可以使用该角色的权限集合与用户集合的笛卡尔乘积来表示。在此基础上，可以进行两个角色的比较。此外，不难看出，由式(3)计算得到的角色间距离实际上是 Jaccard 距离[9]。

下面定义单一角色到一个角色集之间的距离。

定义 5(角色到角色集之间的距离)　设 γ 是一个角色，R 是 c 中的角色集 R，γ 到 R 之间的距离是 γ 到 R 中任意角色的最小距离：

$$minjac(\gamma,R)=\min_{\delta\in R} jac(\gamma,\delta) \tag{4}$$

最后，定义两个 RBAC 配置之间的距离。

定义 6(RBAC 配置之间的距离)　设 c_i 和 c_j 是两个 RBAC 配置，R_i 和 R_j 分别是 c_i 和 c_j 中的角色集，则它们之间的距离为：

$$j(c_i,c_j)=|R_i|^{-1}\sum_{\gamma\in R_i} minjac(\gamma,R_j) \tag{5}$$

2.2.3　目标函数的定义

基于前两节讨论，本节分别为用户权限调用情况要素和系统管理员专家要素定义两个函数 $h()$ 和 $j()$。

新目标函数的定义如下：

$$f(c_{new})=\alpha\cdot h(c_{new})+(1-\alpha)\cdot j(c_{old},c_{new}) \tag{6}$$

其中，c_{new} 是通过角色动态调整算法得到的新 RBAC 配置；c_{old} 是原有 RBAC 配置；$\alpha\in[0,1]$ 是调节参数，用以调节目标函数对 $h()$ 和 $j()$ 这两个要素的偏好程度。

为判定角色 γ 的质量，定义一个启发式函数 rs 来计算该角色的分数：

$$rs(\gamma)=\alpha\cdot RH(\gamma)+(1-\alpha)\cdot minjac(\gamma,R) \tag{7}$$

其中，R 是 c_{old} 包含的角色集合；$RH()$ 和 $minjac()$ 分别由式(1)和式(4)计算得到。

最后，根据定义 1 和式(6)得到角色动态调整问题的定义。

定义 7(角色动态调整问题)　给定原 RBAC 配置 $c=\langle U,P,R,\boldsymbol{URA},\boldsymbol{RPA}\rangle$，$\boldsymbol{UPA}$ 和 $\boldsymbol{UPIM}$，找到新 RBAC 配置 $c^*=\langle U,P,R^*,\boldsymbol{URA}^*,\boldsymbol{RPA}^*\rangle$，能够满足等式 $\boldsymbol{UPA}=\boldsymbol{URA}^*\otimes\boldsymbol{RPA}^*$，并且使目标函数 $f(c^*)=\alpha\cdot h(c^*)+(1-\alpha)\cdot j(c^*,c)$ 的值最小。

文献[10]已经证明了广义角色挖掘问题是 NP-完全的。由于角色动态调整问题可以被规约到广义角色挖掘问题，因此该问题也是 NP-完全的。

3　角色动态调整算法

为解决角色动态调整问题，本文基于启发式搜索策略提出了一种角色动态调整算法，如算法 1 所示。

算法 1　RDA 算法

输入：c=〈U,P,R,**URA**,**RPA**〉,**UPIM**,α,TotalTimes

输出：c^*=〈U,P,R^*,$\boldsymbol{URA}^*$,$\boldsymbol{RPA}^*$〉

1. t←0, DP←∅, CR_{old}←∅, UR←∅, **UPA**=**URA**⊗**RPA**, $\boldsymbol{UPA}_{temp}$=**UPA**
2. for each $p_i\in P$ do
3.　　UR←UR∪{{p_i}} ＃根据权限集合 P 得到单位角色集合
4. end for
5.　CR←UR
6.　while (|CR_{old}−CR|>0)&&(t+1≤TotalTimes) do
7.　　DP←∅ ＃迭代执行角色调整
8.　　for each $\mu_i\in$CR do
9.　　　for each $\mu_j\in$CR do
10.　　　　DP←DP∪{$\mu_i\cup\mu_j$} ＃子集结对操作
11.　　　end for
12.　　end for
13.　　CR_{old}←CR, CR←∅
14.　　DP′=Sort(DP,∈**UPIM**,c,α) ＃依据角色分数对角色排序
15.　　for each $\gamma_i\in$DP do
16.　　　if every element in $\boldsymbol{UPA}_{temp}$ is 0 then
17.　　　　break
18.　　　end if
19.　　　if γ_i cannot cover any 1's in $\boldsymbol{UPA}_{temp}$ then
20.　　　　continue
21.　　　　end if
22.　　　CR←CR∪γ_i
23.　　　change all 1's in $\boldsymbol{UPA}_{temp}$ covered by γ_i to 0's
24.　　end for
25.　end while
26.　{R^*,$\boldsymbol{URA}^*$}=ReAssignment(U,P,**UPA**,CR)
27.　RPA^*←0 ＃ 将布尔矩阵 RPA^* 初始化为全 0
28. for each $\mu_i'\in R^*$ do
29.　　for each $p_j\in P$ do
30.　　　if $p_j\in\mu_i'$ then
31.　　　　$RPA^*_{ij}=1$
32.　　　end if
33.　　end for
34. end for
35. return c^*=〈U,P,R^*,$\boldsymbol{URA}^*$,$\boldsymbol{RPA}^*$〉

3.1　候选角色的生成

这一阶段的工作见算法 1 第 1－25 行伪代码。

首先，RDA 算法依据原有 RBAC 配置中的权限集合 P，分析得到一组“单元角色”(Unit Role，UR)(每个 UR 只拥有一种权限，任意两个 UR 的权限不相同)，再将 UR 复制给候选角色集合 CR。

然后，RDA 算法开始迭代执行，直至满足下列结束条件之一：(1)执行次数到达上限；(2)CR 不再变化。每轮迭代中，先清空缓冲池 DP，再将 CR 中的全部角色进行“子集结对”操作，并复制到 DP。所谓子集结对，就是由两个单元角色{p_x}和{p_y}生成{p_x,p_y}。

接着，使用 Sort()函数(见算法 2)，先计算角色分数，再按照角色分数的高低对 DP 角色进行排序，排序后 DP 队首角色的分值最高。

最后，队首角色被复制给 CR，并检查队首角色在矩阵 $\boldsymbol{UPA}_{temp}$ 中对应的元素，将全部值为 1 的对应元素置 0。

算法 2　排序函数 Sort()

输入：DP,**UPIM**,α,c

输出：DP′

1. 初始化角色分值数组 score[]，数组大小与 DP 相同。
2. for each $\gamma_i\in$DP do
3.　　score[i]=rs(γ_i) ＃使用式(7)计算每个角色的分值

4. end for
5. 依据 score[]中分值的高低，对 DP 重排序。
6. return DP′ ＃DP′是排序结果

3.2 角色重新分配

角色重新分配可形式化描述如下：给定一位用户 i，其权限集合为 $PMS_i=\{p_{i_1},p_{i_2},\cdots,p_{i_k}\}$，角色集合为 $ROLES_i=\{\gamma_1,\gamma_2,\cdots,\gamma_l\}$，每个 γ_j 均为 PMS_i 的子集。所谓角色重新分配，就是从 $ROLES_i$ 中找到最少数量的不同角色，使这些角色拥有的权限覆盖 PMS_i。显然，这类似于集合覆盖问题，而后者已被证明是 NP-完全的[11]。

一般情况下，若用户已拥有角色的权限集合是另一个角色权限集合的超集(superset)，则用户自动获得该角色。例如，设用户已拥有角色 $\gamma=\{p_1,p_2,p_3\}$，而另一角色 $\mu=\{p_1,p_2\}$，则用户将自动获得角色 μ。但这就造成了组织角色权限的冗余，不符合最小特权原则，大大增加了组织遭受内部威胁的可能性。因此，角色动态调整算法必须要解决这一问题。

RDA 算法使用 ReAssignment()算法(见算法 3)迭代实现组织角色的动态调整。每轮迭代中，ReAssignment()算法从 $ROLES_i$ 中选择覆盖 PMS_i 权限数量最多的一个角色添加到 R_i 中，并从 PMS_i 中删除该角色所覆盖的权限。当 PMS_i 为空时，ReAssignment()算法停止迭代，R_i 中所有角色被分配给用户 i。

按照上述过程，当组织为每位用户都重新分配新角色集后，RDA 算法将返回新 RBAC 配置 $c^*=\langle U,P,R^*,\boldsymbol{URA}^*,\boldsymbol{RPA}^*\rangle$。其中，$R^*$ 是调整角色集合；$\boldsymbol{URA}^*$ 是新的用户-角色分配矩阵，表示新的用户权限关系；$\boldsymbol{RPA}^*$ 是新的角色-权限分配矩阵，表示新的角色权限关系。

算法 3 角色重新分配算法 ReAssignment()

输入：CR，**UPIM**，P，U
输出：**URA**，R
1. R←Ø，m=|U|
2. for each $\mu_i\in$U do
3. $PMS_i\leftarrow\{p_j\mid\forall p_j\in P,UPA_{ij}=1\}$
   $ROLES_i\leftarrow\emptyset$
4. while $PMS_i\neq\emptyset$ do
5. 从 CR 中选择角色 μ_k，$\mu_k\subseteq PMS_i$ 且 $|PMS_i\cap\mu_k|$ 最大
6. $PMS_i\leftarrow PMS_i-\mu_k$
7. $ROLES_i\leftarrow ROLES_i\cup\{\mu_k\}$
8. end while
9. R←R∪ROLES$_i$
10. end for
11. 使用整数 1 到 h=|R|来重新索引集合 R 中的角色，即 R=$\{\mu_1',\mu_2',\cdots,\mu_h'\}$
12. 构建布尔矩阵 **URA**，若 $\mu_j'\in ROLES_i$，则 $URA_{ij}=1$；否则 $URA_{ij}=0$。
13. return R，**URA**

3.3 合理性评判

一般来说，只有 RDA 算法输出的新角色集合优于原角色集合，才认为此次动态角色调整合理。因此，本文采用“异常用户率”这一指标对 R^* 的质量进行评判。

为计算异常用户率，本文设计了一种基于一类向量机(one-class Support Vector Machine，SVM)[12]的异常用户检测算法，用于对 R^* 中属于每个角色的异常用户进行检测。

异常用户检测算法采用基于 RBF 内核的一类 SVM，如式(8)所示：

$$K(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp(-g\cdot\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2) \tag{8}$$

为了对 R^* 中的每个角色 r_l 进行评估，首先根据定义 2 得到投影矩阵，再将该投影矩阵中的行向量划分成训练集和测试集两个部分。接下来，使用 EnsembleSVM 从训练集得到合适的 v 和 g[13]。最后，使用这两个参数进行训练以获得符合要求的一类 SVM，该一类 SVM 发现的异常用户率将被用于评判新角色集合的合理性。

4 实验验证与性能分析

4.1 实验数据集

为分析 RDA 算法的性能，特别是该算法根据目标系统日志文件信息获取新 RBAC 配置的合理性，本文使用了某三甲医院电子医疗记录(EMR)系统的访问日志文件作为实验数据集。表 1 给出了该访问日志文件的部分原始字段。

表 1 医院 EMR 系统的原始字段

Table 1 Fields' speciation for the EMR system

字段名称	含义	字段名称	含义
User ID	用户标识	Access Reason	数据访问事由
Encounter ID	治疗标识	Number of Order	用户指令数
Patient ID	病人标识	Patient Location	病人病房位置
User Position	用户职务	Patient Service	病人住院科室
Date and Time Stamp	时间戳		

在表 1 中，用户是为病人直接提供治疗服务的人员，包括医生、护士、药剂师等不同类别。每位用户经医院信息中心审核后，获得访问 EMR 系统所需的“用户标识”，“用户标识”直接和“用户职务”绑定，每个“用户职务”都拥有相应的访问权限。当病人被收入医院科室进行治疗时，系统为该病人分配本次住院治疗事件的“治疗标识”“病人标识”“病人所在科室”和“病人病房位置”。

表 2 给出了 EMR 系统访问日志文件的实例。其中每个实例都记录着用户对 EMR 系统的一次访问。访问日志文件包含 2018 年某季度全部用户的访问信息，表 3 列出了相关统计信息。

表 2 EMR 系统访问日志实例

Table 2 Instances of access logs for EMR system

字段名称	实例 1	实例 2	实例 3
User ID	u_1	u_2	u_3
Patient ID	p_1	p_2	p_3
Time	10/8/18	12/6/18	12/14/18
Patient Service	产科	产科	儿科
User Position	主治医生	护士	护士
Access Reason	治疗病人	护理病人	护理病人
Access Location	2 号楼	2 号楼	2 号楼

表 3 EMR 系统访问日志的统计数据

Table 3 Statistics of EMR access logs

字段名称	数量	字段名称	数量
用户标识	6028	用户职务	155
访问事由	142	访问位置	5
住院科室	30	访问数量	846980

根据表 1－表 3 的信息，本文构造了 RBAC 配置 $c=\langle U, P, R, \boldsymbol{URA}, \boldsymbol{RPA}\rangle$ 和用户-权限调用矩阵 $\boldsymbol{UPIM}$，如表 4 所列。

表 4　RBAC 配置和 $\boldsymbol{UPIM}$ 构造

Table 4　RBAC configuration and $\boldsymbol{UPIM}$ construction

符号	含义	符号	含义
U	由访问日志中全部用户构成的用户集合	R	由访问日志中的全部用户职务构成角色集合
P	R 中每个用户职务可勾选的使用事由构成的权限集合	$\boldsymbol{UPIM}$	一个 $\lvert U\rvert\times\lvert P\rvert$ 的整数矩阵，若第 i 位用户和第 j 个用途同时出现在访问日志中的次数为 t，那么 $UPIM_{ij}=t$
$\boldsymbol{RPA}$	一个 $\lvert R\rvert\times\lvert P\rvert$ 的布尔矩阵，若第 i 个用户职务可构选第 j 个用途，那么 $RPA_{ij}=1$，否则 $RPA_{ij}=0$	$\boldsymbol{URA}$	一个 $\lvert U\rvert\times\lvert R\rvert$ 的布尔矩阵，若某个访问日志项中同时出现第 i 位用户和第 j 个用户职务，则 $URA_{ij}=1$，否则 $URA_{ij}=0$

4.2　评价指标

为评估 RDA 算法所得 RBAC 配置的合理性，本文重点考虑两个指标。

(1)RBAC 配置间距离：用于表征新旧 RBAC 配置之间的距离。

任何一个系统在进行角色动态调整时，都要考虑成本问题。如果新 RBAC 配置和原有 RBAC 配置的差别过大，就很可能导致组织的原有业务流程被中断，原有职责分离约束被破坏。因此，RBAC 配置之间的距离不可过大。

(2)RBAC 异常用户率：用于表征每个角色所含异常用户的数量。

对于角色 r_l，首先得到该角色对应的投影矩阵 $\boldsymbol{M}_{r_l}$，然后使用 3.3 节所述的异常用户检测算法计算每个角色的异常用户率，最后再得到 RBAC 异常用户率。具体来说，$\boldsymbol{M}_{r_l}$ 的行向量被划分为 3 部分：$\{part_1, part_2, part_3\}$。选择其中一部分作为测试集，剩下两部分则作为训练集和验证集。在学习得到符合要求的一类 SVM 模型之后，再对测试集进行异常检测。被分类为负(Negatives)的所有行向量都被认为是异常的，行向量表征的用户就是异常用户。为确保 $\boldsymbol{M}_{r_l}$ 中的所有行向量都被评价，上述过程按三重交叉验证方式进行，即 3 次使用不同的测试集和训练集。最后，计算角色 r_l 的异常用户率：

$$or_l=\frac{\sum_{i=1}^{3}(part_i\text{ 中异常行向量数})}{\boldsymbol{M}_{r_l}\text{ 中行向量总数}} \tag{9}$$

根据式(10)，可计算新 RBAC 配置中全部角色的整体异常用户率：

$$oor=\frac{\sum_i or_i\cdot n_i}{\sum_i n_i} \tag{10}$$

其中，n_i 是被分配了角色 r_i 的用户数量。

4.3　实验结果

实验的软硬件环境为：Intel Core i5 2.53 GHz CPU，4 GB 内存，Windows 7-64 位操作系统。本文在 Virtual Studio 2010 集成开发环境下，使用 C++语言实现了 RDA 算法，基于 libsvm 实现了一类向量机。

4.3.1　运行效率分析

当 α 取不同值时，在 EMR 数据集上多次运行 RDA 算法，该算法输出新 RBAC 配置的时间如表 5 所列。可以看出，RDA 算法的最长运行时间为 19.4 min，说明该算法可在合理时间内运行完毕。

表 5　RDA 算法的运行时间

Table 5　Run time for RDA

α	时间/min	α	时间/min
1	19.4	0.8	10.7
0.95	12.5	0.7	8.5
0.94	12.1	0	4.5
0.9	11.8		

4.3.2　可调参数作用的分析

在新目标函数 $f(c_{\text{new}})$ 中，可调参数 α 的作用是调节用户权限调用情况要素和系统管理员专家要素的相对重要性。因此，考查不同 α 值情况下 RDA 算法获取新角色集合的性能。理想情况下，即使只考虑一种要素，RDA 算法获取的新角色集合与原角色集和的差别也必须在较小的范围之内，才能满足不影响组织业务流程正常开展的需求。

图 1 上的每个点，是 α 取不同值时，在 EMR 数据集上运行 RDA 算法得到的 RBAC 配置。曲线上每个点旁边的数就是生成 RBAC 配置时使用的 α 值。从图中可以看出，α 值越大，说明角色动态调整时更多地考虑了用户权限实际使用情况，新 RBAC 配置的用户异常率就越低；但正因为较少考虑原 RBAC 配置，所以 RBAC 配置间的距离必然较大，对组织业务过程可能产生的干扰也就较大。极端情况下，当 α 为 1 时，新 RBAC 配置的用户异常率为 12%；当 α 为 0 时，新 RBAC 配置的用户异常率为 23%。上述运行结果表明：RDA 算法能够对目标系统的 RBAC 配置进行有效调节。

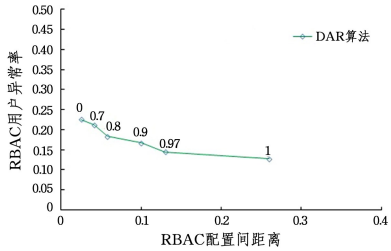


图 1　α 取不同值时 RDA 算法结果的合理性指标

Fig. 1　Indexs of rationality of results of RDA algorithm with different α values

到目前为止，我们还没有找到以动态调整角色为基本思路的内部威胁解决方案，缺乏性能横向比较的基础。注意到，Molloy 等在 2012 年提出的最小扰动角色挖掘(RM-MP)算法也体现了对系统原有 RBAC 角色集进行适时优化的思路[14]。该算法使用参数 $w\in[0,1]$ 来控制新生成角色数量和新旧角色集间距离之间的平衡。本文对该算法进行了初步分析，结果表明其运行时间至多为 16min 左右，这主要是因为该算法没有使用机器学习方法来选择生成新角色；另一方面，极端情

况下，当 $w=0$ 时新旧角色集的距离很大，当 $w=1$ 时又没有新角色生成，这些情况都严重影响了系统的稳定运行。翟志刚等提出的最小扰动混合角色挖掘方法与 RM-MP 算法思路较为接近[6]，在评价角色调整结果质量方面，两者都采用了角色相似性作为主要指标[6,14]，可以推断，该算法的性能也接近 RM-MP 算法。

结束语 RBAC 是目前被广泛使用的一种预期性访问控制机制，能够起到重要的内部威胁抵御作用。然而，只有定时、合理地改变组织内部的角色集合，才能进一步提高抵御效能。本文提出了一种角色动态调整算法。该算法综合考虑管理员专家知识和用户权限实际使用情况，基于启发式搜索和子集结对技术得到候选角色集，使用启发式函数对候选角色集进行删选，得到调整角色集；以降低角色冗余度为目标，使用调整角色集为用户重新分配角色，从而得到新 RBAC 配置。为验证算法性能，本文使用某医院 EMR 系统访问日志数据进行了实验分析，结果表明 RDA 算法能够有效调节组织角色集，为抵御内部威胁提供有力支撑。下一步，我们将深入分析基于最小扰动的角色挖掘算法，探索将其引入内部威胁防御体系的新方法。

参 考 文 献

[1] WANG G F, LIU C Y, PAN H Z, et al. Survey on Insider Threats to Cloud Computing [J]. Chinese Journal of Computers, 2017, 40(2): 296-316.

[2] POVEY D. Optimistic security: A new access control paradigm [C]// Proceedings of the 1999 Workshop on New Security Paradigms. New York: ACM, 1999: 40-45.

[3] COYNE E J. Role engineering [C]// Proceedings of the First ACM Work Shop on Role Based Access Control. New York: ACM, 1996.

[4] ZHOU C, REN Z Y, WU W C. Semantic Roles Mining Algorithms Based on Formal Concept Analysis [J]. Computer Science, 2018, 45(12): 117-122, 129.

[5] ZHANG L, ZHANG H L, HAN D J, et al. The Theory and Algorithm for Roles Minimization Problem in RBAC Based on Concept Lattice [J]. Acta Electronica Sinica, 2014, 42(12): 2371-2378.

[6] ZHAI Z G, WANG J D, CAO Z N, et al. Hybrid Role Mining Methods with Minimal Perturbation [J]. Journal of Computer Research and Development, 2013, 50(5): 951-960.

[7] SANDHU R S, COYNEE J, FEINSTEINH L, et al. Role-based Access Control models [J]. Computer, 1999, 29(2): 38-47.

[8] ZHANG D, EBRINGER T, RAMAMOHANARAO K. Role Engineering Using Graph Optimization [C]// Proceedings of The 10th ACM Symposium on Access Control Models and Technologies. New York: ACM, 2017: 139-144.

[9] HAVELIWALA T H, GIONIS A, KLEIN D D, et al. Evaluating Strategies for Similarity Search on the Web [C]// Proceedings of the 11th International Conference on the World Wide Web. New York: ACM, 2002: 432-442.

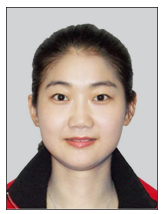
[10] SCHAAD A, MOFFETT J, JACOB J. The Role-based Access Control System of a European Bank: a Case Study and Discussion [C]// Proceedings of the 6th ACM Symposium on Access Control Models and Technologies. New York: ACM, 2001: 3-9.

[11] GAREY M R, DAVID S J. Computers and Intractability: A Guide to the Theory of NP-Completeness [M]. New York: W. H. Freeman and Company, 1990: 320-334.

[12] SANDHU R S. Lattice-based Access Control Models [J]. IEEE Computer, 1993, 26(11): 9-19.

[13] CLAESEN M, DE SMET F, SUYKENS J A K, et al. EnsembleSVM: A Library for Ensemble Learning Using Support Vector Machines [J]. Journal of Machine Learning Research, 2014, 15(1): 141-145.

[14] MOLLOY I, PARK Y, CHARI S. Generative Models for Access Control Policies: Applications to Role Mining Over Logs with Attribution [C]// Proceedings of the 17th ACM SACMAT. New York: ACM, 2012: 45-56.



PAN Heng, born in 1977, Ph.D, associate professor, is a member of China Computer Federation. Her main research interests include Security risk assessment of information System, Network security situation awareness and blockchain technology.

LI Jing-feng, born in 1977, Ph.D, associate professor. His main research interests include information system security technologies and information security guarantee.