第 36 卷 第 4 期 计 算 机 工 程 2010 年 2 月
Vol.36 No.4 Computer Engineering February 2010
·安全技术· 文章编号：1000—3428(2010)04—0121—04 文献标识码：A 中图分类号：N945.12

基于多值逻辑 Petri 网的攻击模型

黄光球，赵阿妮
(西安建筑科技大学管理学院，西安 710055)

摘 要：针对基于 Petri 网攻击模型存在模型繁杂、规模大的缺点，提出从多条相关的命题规则生成多值逻辑 Petri 网(MVPN)的攻击模型。采用逆向推理简化模型，应用 L-M 算法对 MVPN 权值进行学习和训练，根据模型的性质与特点给出模糊逻辑推理算法。实验结果证明，该模型能对网络攻击行为进行描述，该算法能减小空间复杂度，提高计算效率。

关键词：攻击模型；多值逻辑 Petri 网；权值学习

Attack Model Based on MVPN

HUANG Guang-qiu, ZHAO A-ni
(School of Management, Xi'an University of Architecture & Technology, Xi'an 710055)

【Abstract】 Aiming at Petri net-based attack model existing the shortage of complicated model and large scale, this paper proposes a new complicated attack model based on Mutli-Valued Logic Petri-net(MVPN) by using a number of scattered but related rules. The backward reasoning method is used to simplify the complicated model, the back promulgation L-M algorithm is used to train the MVPN weights of the model, a fuzzy logic algorithm is put forward according to the nature and characteristics of the model. Experimental results prove that this model can clearly describe attack activities, its associated algorithm can reduce space complexity and improve the efficiency of calculation.

【Key words】 attack model; Multi-Value logic Petri Net(MVPN); weight learning

基于 Petri 网构建的攻击模型对攻击的描述，可以高度抽象，也可以对攻击过程进行细化，具有很大的灵活性。以往基于模糊 Petri 网的攻击模型存在的缺陷，有色 Petri 网可以解决该问题。本文以文献[1]提出的吸收了模糊 Petri 网和有色 Petri 网优点的多值逻辑 Petri 网(Multi-Value Logic Petri Net, MVPN)为基础，定义一种基于 MVPN 的攻击模型。鉴于多值产生式规则参数难以确定，本文模型中参数的确定采用收敛速度更快的 L-M(Levenberg-Marquardt)算法来确定。

1 基于 MVPN 的攻击模型

1.1 相关定义

定义 1 本文模型是一个 10 元组

本文模型==$(P,T,D,I,O,R,F,\Gamma,W,\Theta)$

其中，$P=\{p_1,p_2,\cdots,p_m\}$ 表示不同的攻击阶段和状态；$T=\{t_1,t_2,\cdots,t_n\}$表示不同的攻击行为；$D=\{d_1,d_2,\cdots,d_m\}$是一个命题的有限集合，是攻击状态 p_i 的命题化描述；$\boldsymbol{I}$: $P\rightarrow T$ 为输入矩阵，$\boldsymbol{I}=\{\delta_{ij}\}$，$\delta_{ij}=\begin{cases}0 & p_i\notin {}^{\bullet}t_j\\ 1 & p_i\in {}^{\bullet}t_j\end{cases}$，$i=1,2,\cdots,m$，$j=1,2,\cdots,n$；$\boldsymbol{O}$: $T\rightarrow P$ 为输出矩阵，$\boldsymbol{O}=\{\gamma_{ij}\}$，$\gamma_{ij}=\begin{cases}0 & p_i\notin t_j^{\bullet}\\ 1 & p_i\in t_j^{\bullet}\end{cases}$，$i=1,2,\cdots,m$，$j=1,2,\cdots,n$；$F=diag(\mu_1,\mu_2,\ldots,\mu_n)$，$\mu_j\in\{0,1,2,\cdots,R\text{-}1\}$，为变迁 t_j 的置信度，即该节点在攻击过程中的满足程度、攻击可能性等；$\Gamma=diag(\lambda_1,\lambda_2,\cdots,\lambda_n)$，$\lambda_j\in\{0,1,2,\cdots,R\text{-}1\}$，$\lambda_j$ 是变迁 t_j 的启动阈值；$W=\{w_{ij}\}$，$w_{ij}\in[0,1]$，w_{ij} 表示从库所 p_i 到变迁 t_j 的连接弧上赋予相应的权重，$w_{ij}=\begin{cases}0 & p_i\notin {}^{\bullet}t_j\\ (0,1] & p_i\in {}^{\bullet}t_j\end{cases}$，$i=1,2,\cdots,m$，$j=1,2,\cdots,n$；$\Theta$ 为一个映象：$P\rightarrow\{0,1,\cdots,R\text{-}1\}$为托肯的颜色，$\Theta(p_i)=\theta_i$ 是该库所对应命题 d_i 的真值；R 为多值逻辑的维度。

定义 2 $\forall t_j\in T$，$\sum_{i=1}^{m}\theta_i\times\delta_{ij}\times w_{ij}\geq\lambda_j$，则称该变迁 t_j 是使能的，$j=1,2,\cdots,n$。

定义 3 使能的变迁可以激发。当变迁 t_j 激发时，它的输入库所中的标记值不改变，而向输出库所 p_k 传送新的标记值为式(1)的值。

$$\theta_k=\left[\frac{\mu_j}{R-1}\times\sum_{i=1}^{m}\theta_i\times\delta_{ij}\times w_{ij}\right] \tag{1}$$

其中，$k=1,2,\cdots,m$；[]为四舍五入法取整运算符。

定义 4 当库所 p_k 是多个变迁 $t_j(j=1,2,\cdots,n)$的输出库所，

$$\theta_k=\left[\max\left(\frac{\mu_1}{R\text{-}1}\times\sum_{i=1}^{m}\theta_i\times\delta_{i1}\times w_{i1},\frac{\mu_2}{R\text{-}1}\times\sum_{i=1}^{m}\theta_i\times\delta_{i2}\times w_{i2},\cdots,\frac{\mu_n}{R\text{-}1}\times\sum_{i=1}^{m}\theta_i\times\delta_{in}\times w_{in}\right)\right] \tag{2}$$

且当这些使能的变迁 t_j 激发时，库所 p_k 得到的标记值为式(2)的值，$k=1,2,\cdots,m$。

1.2 基于 MVPN 的攻击模型的构建

若已知多条相关的攻击规则，生成本文模型的算法如下：

输入 q 条符号化的攻击规则

输出 与其等价的本文模型

Step1 构造一个以“ → ”为根的树，包括左右 2 个孩子“ C(条件)”和“ RS(结论)”，定义 TF 为真实父亲，令 $TF=C$，PF 为临时父亲，Y 为当前遍历到的元素，定义二维虚节点数组 V_{ij}，$i=1$，$j=1$。

Step2 对每条规则的元素逐个遍历。

开始遍历：

如果 Y=规则的首个操作数，则把 Y 作为 TF 的孩子，令 $PF=Y$；

基金项目：陕西自然科学基金资助项目(2007E$_2$17)

作者简介：黄光球(1964－)，男，教授、博士，主研方向：网络安全，计算机仿真；赵阿妮，硕士

收稿日期：2009-07-23 **E-mail：**huangnan93@sohu.com

如果 $Y=$“ $\cup$ ”，则继续遍历，把 Y 作为 TF 的孩子，令 $PF=Y$；

如果 $Y=$“ $\cap$ ”，则继续遍历，把 Y 作为 PF 的孩子，令 $PF=Y$；

如果 $Y=$“ (”，则找到其对应的“) ”，将括号内看成一个以整体令其为 V_{ij}，并将 V_{ij} 作为 TF 的孩子，$PF=V_{ij}$, $L_1=Y_0$。虚节点下标标记时，同一级别的 i 对应 j 依次加 1；更高一级别的 i 自动加 1，j 从 1 开始依次加 1；

$TF_1=TF$；

继续遍历，如果 $Y=$“ $\cup$ ”，则继续遍历并把 Y 作为 TF 的孩子，按照上述规则从 L_1 开始遍历括号内，$TF=V_{ij}$，直到遇到其对应的右括号；

继续遍历，如果 $Y=$“ $\cap$ ”，则此时标记 $L_1=Y$, $PF_1=PF$，将上述括号看为整体，按照上述规则遍历括号内，$TF=V_{ij}$，直到遇到其对应的右括号。

然后对 $L_1=$“ $\cap$ ”后继续遍历，把 L_1 后的作为 PF_1 每个叶子节点的孩子，直到括号内遍历结束或遇到“ $\cup$ ”；

$TF=TF_1$；

继续遍历；

若前提遍历完遇到“ $\rightarrow$ ”，本文则按照上述规则遍历结论并将其作为节点 RS 的孩子；

Step3 按照 Step2 遍历完规则，再找 C 每个叶子节点到 C 每个孩子的路径，结合根节点和右孩子即结论节点得到 n 条简化规则。此时得到的各库所都是按照与规则连接或一条简化规则，只有一个输入库所，每条规则只含有一个变迁。

Step4 按照上述步骤遍历完每条规则。同时得到库所的数量 m，库所颜色值集合 Θ，变迁置信度集合 F，权值集合 W；定义 $\boldsymbol{I}$, $\boldsymbol{O}$ 2 个 $m\times n$ 的矩阵，其分别为输入、输出矩阵：$\boldsymbol{I}=(\delta_{ij})_{m\times n}$, $\boldsymbol{O}=(\gamma_{ij})_{m\times n}$；根据 $\boldsymbol{I}$, $\boldsymbol{O}$ 2 个 $m\times n$ 的矩阵及本文模型的定义可以得到 q 条输入规则的本文模型。

1.3 权值学习[2-3]

仿照反向传播(Back Propagation, BP)算法，将本文模型分层，在每一层上对每一个模型节点用误差反传的方法来学习权值。但由于梯度下降法收敛速度慢，因此采用 L-M 算法调整权值。

1.3.1 模型分层

使用文献[2]分层算法处理图 1，处于同一层的变迁 t_1 t_2, t_3 在触发时必将发生冲突。针对该缺点，本文把同一个库所的输出变迁尽可能地置于同一层次，同时在必要时增加相应的虚库所和虚变迁。虚库所和虚变迁起到中间过渡作用，不对规则库产生影响。在分层模型中，虚库所的 θ^0 设为 0，虚变迁的置信度设为 R-1。

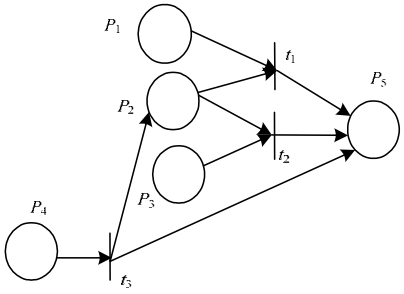

图 1 本文模型实例

本文模型的分层算法如下：

Step1 建立起始库所集 $Pset$：若 $\theta_i^0\neq 0$，则 $p_i\in Pset$。设循环变量 L，初值为 1，标记模型的层数。

Step2 建立变迁集合：$T_L=\{t\in T \mid \forall\ p\in t, p\in Pset\}$；//$T_L$ 表示处于//同一层的变迁集合。

Step3 若 $\exists\, t\in T_L$, $\exists\, p\in t$, $\exists\, t_l\in T\text{-}T_L$, $\exists\, p\in t_l$，则为模型添加虚库所 p_s 和虚变迁 t_r，其中，$s=m+1$；$r=n+1$。

$t\in p_s$ //将变迁 t 置于新增虚库所 p_s 的输入变迁集中

$p_s\in t_r$ //将新增虚库所 p_s 作为虚变迁 t_r 的输入库所

$t_r\in p$ //新增虚变迁 t_r 的输出库所为库所 p

同时，令

$P=P+\{p_s\}$, $m=m+1$； //将新增虚库所 p_s 加入模型库所集 P 中

$T=T+\{t_r\}$, $n=n+1$；//将新增虚变迁 t_r 加入模型变迁集 T 中

$\theta_s^0=0$；//新增虚库所 p_s 所对应 θ_s^0 的设为 0

$\mu_r=R$-1；//新增虚变迁 t_r 的置信度设为 R-1

$w_{sr}=1$；//新增加的虚库所和虚变迁的输入弧权值为 1

Step4 $Pset=Pset\cup\{p\in t \mid \forall\ t\in T_L\}$；//将 T_L 中所有变迁的输出库所//都添加到起始库所集 $Pset$ 中

Step5 $T=T\text{-}T_L$；//从模型变迁集 T 中删除 T_L 中的变迁

Step6 若 $T=\varnothing$，则算法结束，模型层数为 L。否则，$L=L+1$，转 Step2。

经分层算法处理的本文模型如图 2 所示。

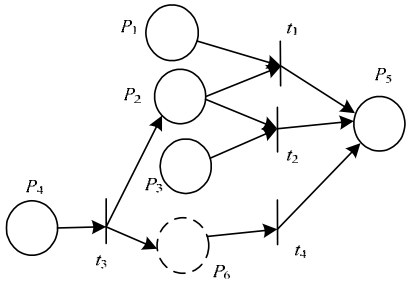

图 2 经分层算法处理的本文模型

1.3.2 反向传播学习算法

因为梯度下降法本身的缺陷以及本文需要，所以对文献[2]算法做了 2 个方面改进：

(1)本文模型中变迁置信度的取值和库所颜色值是 R 值逻辑的，并加入逻辑量 δ_{ij} 和 γ_{kj}，可得连续函数为

$$\theta_k=(1+\exp(-b(x-c)))^{-1}\times\frac{\mu_j}{R-1}\times\sum_{i=1}^{m}\theta_i\times\delta_{ij}\times\gamma_{kj}\times w_{ij} \tag{3}$$

其中，b 是一常量，且足够大；$c=\lambda_j$，c 的取值是跟 R 有关的整数。

(2)因为文献[2]梯度下降法相邻 2 次迭代的搜索方向总是正交的，所以收敛速度慢。针对该问题，本文应用非线性优化方法 L-M 算法[3]，它既有高斯-牛顿法的局部特性，又具有梯度法的全局特性。式(4)为调整权值的 L-M 规则，其中，d 为学习次数。

$$w_{xj}^{(k)}(d+1)=w_{xj}^{(k)}(d)+\Delta w=w_{xj}^{(k)}(d)+(J^T(w)J(w)+\sigma\cdot I)^{-1}J^T(w)E \tag{4}$$

$$w_{mj}^{(k)}(d+1)=1-\sum_{x=1}^{m-1}w_{xj}^{(k)}(d+1) \tag{5}$$

在式(4)中，$\boldsymbol{J}(w)$为最终结论置信度误差对权值导数的 DeJacobian 矩阵，$\boldsymbol{J}(w)=\begin{Bmatrix}\frac{\mathrm{d}E_1}{\mathrm{d}w_{1j}^{(k)}} & \frac{\mathrm{d}E_1}{\mathrm{d}w_{2j}^{(k)}} & \cdots & \frac{\mathrm{d}E_1}{\mathrm{d}w_{mj}^{(k)}}\\ \frac{dE_2}{\mathrm{d}w_{1j}^{(k)}} & \frac{dE_2}{\mathrm{d}w_{2j}^{(k)}} & \cdots & \frac{dE_2}{\mathrm{d}w_{mj}^{(k)}}\\ \vdots & \vdots & & \vdots\\ \frac{dE_a}{\mathrm{d}w_{1j}^{(k)}} & \frac{dE_a}{\mathrm{d}w_{2j}^{(k)}} & \cdots & \frac{dE_a}{\mathrm{d}w_{mj}^{(k)}}\end{Bmatrix}$；$\boldsymbol{E}$ 为误差向量；$\boldsymbol{I}$ 为单位矩阵；σ 为自适应调整值，随 σ 的增大，L-M 规则的 $\boldsymbol{J}^{\mathrm{T}}(w)\boldsymbol{J}(w)$ 可以忽略，$w_{xj}^{(k)}(d+1)$ 接近于梯度法；当 σ 很小时，$\boldsymbol{J}^{\mathrm{T}}(w)\boldsymbol{J}(w)$ 接近于 Hessian 矩阵，$w_{xj}^{(k)}(d+1)$ 接近于高斯-牛顿法。

$\boldsymbol{J}(w)$中每一元素的计算参考文献[2]逐层计算每一批样本数据的一阶梯度，即

$$\begin{cases}\dfrac{\mathrm{d}E}{\mathrm{d}w_{xj}^{(L)}}=\dfrac{\mathrm{d}E}{\mathrm{d}(\theta_j^{(L)})}\dfrac{\mathrm{d}(\theta_j^{(L)})}{\mathrm{d}w_{xj}^{(L)}} & t_j^{(L)}\in T_L, x=1,2,\cdots,m-1\text{或}p_i^{(L-1)}\in O(t_j^{(L-1)}),\ p_i^{(L-1)}\text{是终止库}\\ \dfrac{\mathrm{d}E}{\mathrm{d}w_{xj}^{(L-1)}}=\dfrac{\mathrm{d}E}{\mathrm{d}(\theta_j^{(L)})}\dfrac{\mathrm{d}(\theta_j^{(L)})}{\mathrm{d}(\theta_i^{(L-1)})}\dfrac{\mathrm{d}(\theta_i^{(L-1)})}{\mathrm{d}w_{xj}^{(L-1)}} & t_j^{(L-1)}\in T_{L-1}, p_i^{(L-1)}\in I(t_j^{(L)}),\exists p_j^{(L)}\in O(t_j^{(L)}), x=1,2,\cdots,m-1\end{cases}\tag{6}$$

依照式(6)以此类推，计算 $\dfrac{\mathrm{d}E}{\mathrm{d}w^{(k)}}$，其中，$k$=$L$-2, L-3,⋯,1；x=1,2,⋯,m-1。权值学习算法如下：

Step1 依据算法 2 对本文模型分层，层数为 L。

Step2 当 d=0，训练允许误差 ε，初始权值向量 w_0，σ=σ_0。

Step3 通过本文模型激发规则逐层激发变迁，计算结论置信度，得到最终结论置信度误差向量 $\boldsymbol{E}(w_l)$，其中，$\boldsymbol{E}=\frac{1}{2}\sum_{z=1}^{r}\sum_{i=1}^{a}(\theta_i^z-(\theta_i^z)^1)^2$ (在式 $\boldsymbol{E}$ 中，θ_i^z，$(\theta_i^z)^1$ 分别表示终止库所 p_i 的第 z 批样本数据输出的实际标记值和期望标记值；a 为终止库所个数，i=1,2,⋯,a；共 g 批样本数据)，若$|E(w_d)|<\varepsilon$，则转 Step6，否则转 Step4。

Step4 采用 L-M 算法对权值进行调整。

Step5 若$|E(w_d)|<|E(w_{d+1})|$，则令 $\sigma=\sigma/\beta$，转 Step2，否则，令 $\sigma=\sigma\cdot\beta$，转 Step4，其中，β 为常数。

Step6 训练结束，得权向量 w_d。

2 推理算法

2.1 模型简化

从规则直接生成的攻击模型可能较复杂，很多时候模型对单个攻击，实际上只涉及到知识库的小部分规则。针对该问题，参考文献[4]，对于需要求解的单个攻击命题，从整个攻击知识库中抽取出与该攻击命题有关的规则，同时找到与该攻击命题相关的前提攻击步骤，简化本文模型，使计算量大大减小。应用 Kronecker 乘法模型简化算法如下：

Step1 初始化。向量 $\boldsymbol{P}_s=(a_1,a_2,\cdots,a_m)^{\mathrm{T}}$，$\boldsymbol{P}_s$ 中对应的要分析的目标库所赋值 1，其他均为 0，$\boldsymbol{P}_s$ 中只有一个 1，$B=(0)_{1\times n}^{\mathrm{T}}$，$T_r$=$H$=$\boldsymbol{O}^{\mathrm{T}}\otimes\boldsymbol{P}_s$，$k$=0。

Step2 If B!=H, then k=k+1，执行 Step3，else 转 step4。

Step3 计算：$\boldsymbol{P}_s$=$\boldsymbol{P}_s$+$\boldsymbol{I}\otimes B\times k$, T_r=T_r+$\boldsymbol{O}^{\mathrm{T}}\otimes\boldsymbol{I}\otimes B$，并令 B=H, H=$\boldsymbol{O}^{\mathrm{T}}\otimes\boldsymbol{I}\otimes B$。

Step4 推理结束。

Step5 根据新的输入输出矩阵 $\boldsymbol{I}$', $\boldsymbol{O}$'与模型定义重新表示简化后的模型。

根据去掉 T_r, $\boldsymbol{P}_s$ 对应的行列中为零的项得到新的 $\boldsymbol{I}$', $\boldsymbol{O}$', P', T", F', W', S^0。

2.2 模糊逻辑推理算法

模糊逻辑推理算法步骤如下：

Step1 初始化 I, O, θ^0, F, W, Γ；

Step2 利用算法 4 将本文模型简化；

Step3 利用算法 2 对模型分层，并利用算法 3 学习和训练权值；

Step4 在考虑变迁的激发阈值和输入权值时，推理过程如下：

(1)初始化可激发变迁的组合 T=$\varnothing$，求出起始库所集 $Pset_s$，以及 $Pset_s$ 中每个起始库所的输出变迁 t_j 和权值集合 W。

(2)对 ∀ p_i∈$Pset_s$, ∀ t_j∈T，对 t_j 进行如下操作：

int y=0; for(i=1;i<=m;i++){ $y=y+\theta_i\times\delta_{ij}\times w_{ij}$;}

if(y>=λ_j) T=T+{t_j};

(3)若 T=$\varnothing$，则转Step(5)，否则令 $Pset_s$=$\varnothing$，对 T 中的各 t_j 分别采取：

for(k=1;k<=m;k++){ $\theta_k=\max(\frac{\mu_j}{R-1}\times\sum_{i=1}^{m}\theta_i\times\delta_{ij}\times\gamma_{kj}\times w_{ij},\theta_k^0)$;$Pset_s$=$Pset_s$+{P_k};}

T=T-{t_j};

(4)若 $Pset_s$=$\varnothing$，则转 Step(5)，否则转 Step(2)；

(5)推理结束。

3 实验验证

以后门攻击为例。假设 R=13。其中，p_1 为系统 telnet 服务开放；p_2 为以普通用户 telnet 登录，在较短的时间内连续登录失败较多的次数；p_3 为客户端的 IP 地址与服务器端的 IP 地址不在同一个网段内；p_4 为登录时间为凌晨 0~6:00；p_5 为普通用户成功执行了 cp, chmod, touch；p_6 为普通用户将 cp 操作的目标文件的 setuid 的位置为 1；p_7 为普通用户用 touch 命令创建了一个空文件 x；p_8 为普通用户执行 *mail root*<x；p_9 为用户以根用户的权限将拷贝后的所有目标文件的 setuid 库所置为 1；p_{10} 为用户以普通身份将一个 shell 文件拷贝为另外的多个不同文件名的普通文件；p_{11} 为用户具有普通用户权限；p_{12} 为用户具有根用户权限；p_{13} 为放置后门攻击；p_{14} 为系统共享目录读写权限过大；p_{15} 为操作系统存在内核漏洞；p_{16} 为用户具有普通用户所没有的一些权限；p_{17} 为特权提升攻击；p_{18} 为系统开放了有漏洞的网络服务；p_{19} 为用户具有 guest 权限；t_1~t_6 分别为不同攻击行为；μ_1~μ_6 为各规则的置信度；取变迁的启动阈值 λ=6；设理想权值为 W，求后门攻击发生的可能性。

W={0.3,0.25,0.25,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.25,0.25,0.25,0.25,0.6,0.4,0.7,0.3,0.5,0.5}。

命题规则：

规则 1 if $p_1(12)\cap p_2(10)\cap p_3(11)\cap p_4(11)\rightarrow p_{11}\cap p_{10}(\mu_1$=10)；

规则 2 if $p_5(12)\cap p_6(11)\cap p_7(11)\cap p_8(10)\cap p_{11}\rightarrow p_{12}(\mu_2$=11)；

规则 3 if $p_9(9)\cap p_{10}\cap p_{12}\rightarrow p_{13}(\mu_3$=10)；

规则 4 if $p_{14}(8)\cap p_{11}\rightarrow p_{16}(\mu_4$=10)；

规则 5 if $p_{15}(12)\cap p_{16}\rightarrow p_{17}(\mu_5$=11)；

规则 6 if $p_{18}(9)\cap p_{19}(10)\rightarrow p_{11}(\mu_6$=8)。

经后门攻击的本文模型如图 3 所示。

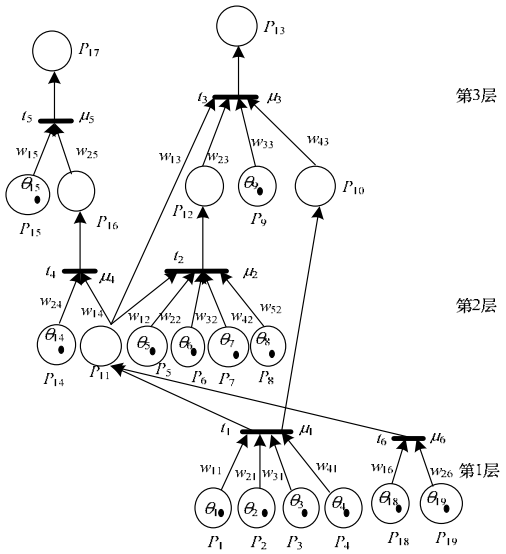

图 3 经后门攻击的本文模型

(1)初始化：

$\boldsymbol{\theta}^0$=(12,10,11,11,12,11,11,10,9,10,0,0,0,8,12,0,0,0,9,10)$^{\mathrm{T}}$；

$\boldsymbol{F}$=diag(10,11,10,10,11,8)；

$\boldsymbol{W}$=(0.3,0.25,0.25,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.25,0.25,0.25,0.25,0.6,0.4,0.7,0.3,0.5,0.5)；$\boldsymbol{\Gamma}$ 中的 λ 全部为 6。

(2)利用算法 4 化简，计算得：

$\boldsymbol{P}_s$=(2,2,2,2,2,2,2,2,1,1,1,1,1,0,0,0,0,1,1)$^{\mathrm{T}}$, $\boldsymbol{T}_r$=(1,1,1,0,0,1)$^{\mathrm{T}}$；

从而得新矩阵 $\boldsymbol{I}$', $\boldsymbol{O}$', $\boldsymbol{P}$'={$p_1,p_2,\cdots,p_{13},p_{18},p_{19}$}, $\boldsymbol{T}$'={t_1,t_2,t_3,t_6}。

$$\boldsymbol{I}=\begin{pmatrix}1&0&0&0&0&0\\1&0&0&0&0&0\\1&0&0&0&0&0\\1&0&0&0&0&0\\0&1&0&0&0&0\\0&1&0&0&0&0\\0&1&0&0&0&0\\0&1&0&0&0&0\\0&0&1&0&0&0\\0&0&1&0&0&0\\0&1&1&1&0&0\\0&0&1&0&0&0\\0&0&0&0&0&0\\0&0&0&1&0&0\\0&0&0&0&1&0\\0&0&0&0&1&0\\0&0&0&0&0&0\\0&0&0&0&0&1\\0&0&0&0&0&1\end{pmatrix}\quad \boldsymbol{O}=\begin{pmatrix}0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\1&0&0&0&0&0\\1&0&0&0&0&1\\0&1&0&0&0&0\\0&0&1&0&0&0\\0&0&0&0&0&0\\0&0&0&1&0&0\\0&0&0&0&0&0\\0&0&0&1&0&0\\0&0&0&0&0&0\\0&0&0&0&1&0\\0&0&0&0&0&0\end{pmatrix}$$

$$\boldsymbol{I}'=\begin{pmatrix}1&0&0&0\\1&0&0&0\\1&0&0&0\\1&0&0&0\\0&1&0&0\\0&1&0&0\\0&1&0&0\\0&1&0&0\\0&0&1&0\\0&0&1&0\\0&1&1&0\\0&0&1&0\\0&0&0&0\\0&0&0&1\\0&0&0&1\end{pmatrix}\quad \boldsymbol{O}'=\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\1&0&0&0\\1&0&0&1\\0&1&0&0\\0&0&1&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}$$

(3)经化简和分层后的本文模型如图 4 所示，增加了虚节点 P_{20}, P_{21} 和虚变迁 t_7, t_8，μ_7=12, μ_8=12。然后利用 1.3.2 节的学习权值，得新权值为

$\boldsymbol{W}$=(0.303 0,0.247 0,0.251 8,0.197 2,0.190 7,0.202 3,0.217 7, 0.199 3,0.190 0,0.260 2,0.249 8,0.251 0,0.249 0,0.495 0,0.505 0)

(4)根据 2.2 节求目标库所的值。为了最终目标库所的准确性，在中间过程不进行取整运算，得：$\boldsymbol{S}^3$=(12,10,11,11,12, 11,11,10,9,9.204,9.204,9.7807,7.8241, 9,10,9.204,9.204)$^{\mathrm{T}}$。通过四舍五入法取整可得到最终状态：$\boldsymbol{S}$=(12,10,11,11,12,11,11,10, 9,9,9,10,8,9,10,9,9)$^{\mathrm{T}}$。

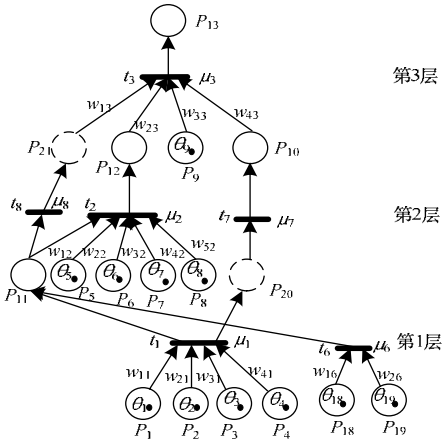

图 4　经化简和分层后的本文模型

由此可知，密码猜测获得普通用户权限攻击发生的颜色值为 10，利用网络漏洞获得根权限攻击发生的颜色值为 10，放置后门攻击发生的颜色值为 8。由最终状态可以发现，在网络攻击者利用密码猜测获得普通用户权限，利用网络漏洞获得根权限成功的情况下，其放置后门攻击的可能性是较大的，与事实相符。

4　结束语

本文从攻击的过程及状态出发，根据命题规则与 Petri 网描述攻击行为的对应关系，提出从命题规则生成 Petri 网的算法。调整权值采用的 L-M 算法是非线性的优化方法，且收敛速度快。权值的学习算法需要进一步改进。

参考文献

[1] 陈　星, 刁永锋, 黄昌海. 一种多值 Petri 网及其应用[J]. 微电子学与计算机, 2005, 22(5): 182-184.

[2] 鲍培明. 基于 BP 网络的模糊 Petri 网的学习能力[J]. 计算机学报, 2004, 27(5): 695-698.

[3] 伏燕军, 杨坤涛. 基于 Levenberg-Marquardt 算法的图像拼接[J]. 激光杂志, 2007, 28(5): 46-47.

[4] 门　鹏, 段振华. 基于代数的模糊 Petri 网逆向推理算法[J]. 系统仿真学报, 2007, 19(1): 161-163.

编辑　陆燕菲

(上接第 120 页)

参考文献

[1] Gates A Q, Mondragon O. FasTLInC: A Constraint-based Tracing Approach[J]. Journal of Systems and Software, 2001, 63(3): 241-258.

[2] Clowes S. Injectso: Modifying and Spying on Running Processes Under Linux and Solaris[Z]. [2008-12-10]. http://www.blackhat.com/presentations/bh-europe-01/shaun-clowes/bh-europe-01-clowes.ppt.

[3] Richter J. Load Your 32-bit dll into Another Process's Address Space Using Injlib[J]. Microsoft Systems Journal, 1996, 9(3): 13-16.

[4] Cesare S. Shared Library Redirection via ELF PLT Infection[J]. Phrack Magazine, 2000, 10(56).

[5] Shacham H. The Geometry of Innocent Flesh on the Bone: Return-into-libc Without Function Calls(on the x86)[C]//Proc. of the 14th ACM Conference on Computer and Communications Security. Alexandria, VA, USA: [s. n.], 2007.

[6] Baratloo A, Singh N, Tsai T. Libsafe: Protecting Critical Elements of Stacks[Z]. (1999-12-13). http://www.research.avayalabs.com/project/libsafe.

[7] Alert7. Solar Designer's Non-executable stack 的实现机理分析[EB/OL]. (2001-12-03). http://www.xfocus.net/articles/200104/160.html.

[8] Linn C M, Rajagopalan M, Baker S, et al. Protecting Against Unexpected System Calls[C]//Proc. of the 14th Usenix Security Symposium. Baltimole, MD, USA: [s. n.], 2005.

[9] ELF Shell Crew. Embedded ELF Debugging: The Middle Head of Cerberus[J]. Phrack Magazine , 2005, 9(63).

[10] Anonymous Author. Runtime Process Infection[J]. Phrack Magazine, 2002, 8(59).

编辑　金胡考