网络病毒的求源问题*)

韩兰胜[1] 韩淑霞[2] Varney Washington[3]
(华中科技大学计算机学院 武汉 430074)[1] (华中科技大学数学系 武汉 430074)[2]
(Cheale Vocational Training Institute, Tubman Boulevard P. O. 816, Monrovia Liberia)[3]

摘 要 现有的网络病毒方面的文章大都强调网络病毒与生物病毒的相似性,而生物病毒的寻源具有很多不确定因素,因此很少有人提出网络病毒寻源的理论。本文强调网络病毒的传播不同于生物病毒的传播,它的传播信息是可以通过网络获取的;病毒的求源虽然是病毒传播的逆过程,但它也不同于严格意义上的反问题。文章提出子网内病毒传播路径的始点即为子网的病毒源。指出病毒的传播引起的计算机结点的状态变迁是获取病毒实际传播路径的主要依据,而计算机感染病毒的诊断和清毒是获取该依据的重要手段。由此,文章建立了子网的状态变迁方程。结合实际的网络因素和实践,文章给出了一套求解的方法和步骤,反复求解子网的状态变迁方程即可求得病毒实际传播的路径,传播路径的始点即为病毒的一个源点。最后,文章对模型及其解法给出了模拟实验,实验证明了理论模型的正确性和求解方法的有效性。文章为建立网络病毒的求源理论打开了一定的思路。

关键词 网络病毒,网络安全,病毒求源

Tracing the Source of Net-Virus

HAN Lan-Sheng[1] HAN Shu-Xia[2] Varney Washington[3]
(School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)[1]
(Department of Mathematics, Huazhong University of Science and Technology, Wuhan 430074)[2]
(Cheale Vocational Training Institute, Tubman Boulevard P. O. 816, Monrova Liberia)[3]

Abstract As most papers about net viruses focus the similarity shared by net viruses and biological virus and the source tracing of the latter relies on many uncertain factors. Few paper establishes theory on the source tracing of net viruses. Different from those papers, this paper focuses on different spreading features of net virus compared with the biological virus, such as the spreading information of net virus can be recorded. Though tracing the source of virus is the inverse of the viruses' spreading, but the paper thinks that tracing the source is not the strict inverse problem so far. The paper presents the definition for the source of viruses in a sub net the start vertex of the spreading path of the virus. The paper points out that state changing of the vertices caused by the spreading of the virus is the important hints to tracing the spreading path of the virus. The scanning and cleaning are main methods to get these hints. Then the paper establishes the source tracing equations for the net virus. Combining with the practice, the paper presents the main steps and methods to get the solutions to the equations. Working out the equations repeatedly, the paper get the spreading path of the virus; thus the start vertex of the path is got which must be the source of the virus in the sub net. Finally, the paper carries out the simulation test on an email group net. The results of the test verify our tracing model and methods of working out the equations. Thus the paper opens a theoretic way to tracing the source of net viruses.

Keywords Network virus, Network security, Tracing source of virus

1 前言

网络病毒已构成网络安全的一个很重要的组成部分[1]。寻找病毒的传播路径并进而求得病毒的源点,对于打击病毒的植入者、防御并制止病毒的传播已成为必要的课题。然而令人遗憾的是,当前探测病源的工作还只集中在肤浅的技术层面,至今还没有形成成熟的理论[2]。

鉴于网络病毒与生物病毒有一定的相似性[3~5],而生物病毒的求源具有很多不确定的因素,这也是网络病毒求源至今未能形成理论的原因之一。事实上,正如我们在文[6]中指出的,网络病毒具有与生物病毒不同的传播特征。网络病毒作为一种应用程序,它在借助网络传播的过程中,传播信息是可以被记录的,病毒对电脑系统的感染也是可被记录的。这为我们寻找到网络病毒的源头提供了有利依据。

我们知道网络病毒作为一个应用程序,依据其传播特性,一般都是在一个逻辑的子网中传播的[7],比如基于邮件传播的病毒是在相互拥有邮件地址的邮件组网中传播,该子网常常由亲朋好友或有业务往来的企业组成,它在一定的时间内是相对稳定的。

故我们在寻找网络病毒源头时采用分划的办法,先在一定的逻辑子网中求源。等找到该子网中的病毒源头时,再来判断它是否是最初的源头;若是就结束;若不是,那么它必然

*)国家自然科学基金(No:60403027)资助项目。**韩兰胜** 讲师,博士研究生,研究方向:网络信息安全;**韩淑霞** 讲师,博士,研究方向:应用数学及数学模型;**Varneg Washington** 访问学者,美国政府资助,研究方向:计算机网络。

由另一子网传来，我们就上溯到另一个子网再求源。严格来讲，子网的数目和子网中的结点的数目都是有限的，所以我们总会在有限步内找到源头。

事实上，网络病毒的求源是病毒传播的逆过程，也可算作是一类反问题。然而它也不像严格意义上的反问题[8,9]。首先，它并没有反问题中那样严格的定解条件。其次，它的传播媒介并非连续，而是离散的、逻辑的通讯网。第三，反问题中的有些"扩散媒介"的性质甚至至今还不确定，而网络病毒的传播路线是可记录的。所以这种逆过程是客观的存在、唯一的，也就是说这类"反问题"的解是存在的。

我们首先给出子网中病毒源头的定义：在一定的逻辑子网中某病毒传播途径上的始点为该病毒在该子网中的源点。根据此定义，子网中的病毒源不一定唯一，这也符合实际情况。

2 求源建模

将子网内的计算机系统视为结点 $v_i(i=1,2,\cdots,n)$，其中 n 是子网内结点的数目。将结点 v_i 与 v_j 的连接视为 v_j 指向 v_i 的一条有向边，记为 (v_j,v_i)，它对应病毒的传播方向。那么得到一个有向图 $D(V,E(t))$，其中 V 为子网内的结点集，$E(t)$ 为 V 上的二元关系，是有向边的集合。因为连接是动态的，故令它是时间的函数。由此可以用一个变化的布尔矩阵 $C_t=(c_{ij}^t)_m$ 来描述子网内在时间 t 时计算机间的连接，其中

$$c_{ij}^t=\begin{cases}1, & (v_i,v_j)\in E(t)\\ 0, & (v_i,v_j)\notin E(t)\end{cases}$$

我们知道病毒传播的必要条件是计算机之间必须是连接的。可是反过来，计算机之间的连接并不能保证该病毒的传播，这还要看连接是否满足具体病毒的传播条件。比如，基于邮件传播的病毒并不能在 FTP 的连接环境中传播。所以，求源还必须弄清结点间的连接类型。在上述连接矩阵的基础上进一步定义（实际上对应病毒的传播机制）病毒的邻接矩阵，记为 $A_t=(a_{ij}^t)_{n\times n}$，其中

$$a_{ij}^t=\begin{cases}1, & (v_i,v_j)\in E_\gamma(t)\\ 0, & (v_i,v_j)\notin E_\gamma(t)\end{cases}$$

也就是说，当 $(v_i,v_j)\in E_\gamma(t)$，即 $a_{ij}^t=1$，则病毒在 t 时就从 v_i 传播到 v_j。之所以在 C_t 的基础上再定义 A_t，主要是从实际应用的角度依据将大量没有产生连接的序偶先排除，从而大大缩小我们的考虑范围。

引入布尔量 x_{it} 来表示结点 v_i 在时间 t 时的状态，称之为状态量。若 $x_{it}=1$，则表示 v_i 在时间 t 时是染毒的；若 $x_{it}=0$，则表示 v_i 在时间 t 时没有染毒。因此向量 $(x_{1t},x_{2t},\cdots,x_{nt})^T$ 就可以表示子网内所有结点在时间 t 时的状态向量。

寻找病毒的源点，结点感染病毒的时间是非常重要的信息，而人们对感染信息的获取是离散的，因此将时间离散化，分成一段一段的时间段。时间段的大小一般依据具体病毒的传播特征和具体网络的连接状况。比如，一个段内至少完成一次连接，或完成病毒的一次传播，或者说保证结点的状态能发生一次改变。当然间隔不宜过大，因为太大会漏掉必要的信息，从而增加求源的难度。

依据现行的网络的协议和能力，我们知道多个结点在发送信息时并不协调一致，而是随机的。故设定在同一个时间间隔内结点 v_i 可以接收来自多个结点的连接，同时 v_i 也可以向多个结点发送信息（如广播、多播）。依据毒源的定义，这里只要求出感染 v_i 的结点即可。故用向量 $(a_{i1}^t,a_{i2}^t,\cdots,a_{in}^t)$ 来表示 v_i 在时间 t 时的接受向量，并假设子网在 t 时的状态向量为 $(x_{1t},x_{2t},\cdots,x_{nt})$ 那么经过这一时间段的连接，子网对 v_i 的状态的影响可用下式表示：

$$a_{i1}^t\cdot x_{1t}+a_{i2}^t\cdot x_{2t}+\cdots a_{in}^t\cdot x_{nt} \tag{1}$$

式(1)的 $\cdot$，$+$ 分别为布尔乘、布尔加。假如 v_i 在时间 t 时的状态为 $x_{it}=0$，而在 $t+1$ 时，$x_{i(t+1)}=1$，那么在式(1)中必然有 $a_{ij}^t=1$，且 $x_{jt}=1$，这样的 j 也不一定唯一。那么这些 $v_j(j=s,t,\cdots)$ 即为 v_i 的可疑感染源，称为 v_i 的嫌疑染点集，记为 $S(v_{i(t+1)})=\{v_{t1},v_{t2},\cdots v_{tn}\}$，其中在该时段第一个感染 v_i 的结点称为 v_i 的前染点，比如是 v_j 的，记为 $P(v_{it+1})=v_{jt}$。也可以用状态量来标记：$P(x_{it+1})=x_{jt}$，反过来称 v_i 为 v_j 的后染点，显然 $P(x_{it+i})\in S(x_{it+1})$。考虑到许多病毒并不重复感染，况且只利用前染点就可以找到子网的毒源，所以这里只要求出 v_{it+1} 的前染点 $P(x_{it+i})$ 即可。

另外，影响 v_i 状态改变的还有一个因素，即结点的清毒，比如杀毒或重新装机等。它们之间的状态转换关系如图 1 所示。

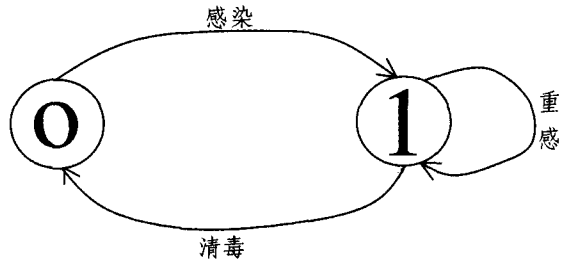

图 1 结点的状态变迁图

实际上，查杀病毒也是获取结点感染状态的重要手段，本文就是建立在实际的网络的运行状态之上的：如有些结点杀毒，有些结点没有杀毒；有时杀毒，有时不杀毒。子网在时间 t 时的杀毒向量记为 $(\delta_{1t},\delta_{2t},\cdots,\delta_{nt})$，因此假设每一个结点在每一个时间间隔中可以杀毒，也可以不杀毒，所以用 $\delta_{it}=1$ 表示有毒并且杀了毒，用 $\delta_{it}=0$ 表示查毒，没有发现病毒；令 $\delta_{it}=\Theta$ 表示 t 时没有查毒。但我们知道某结点在传播前杀毒和传播后杀毒对所连接的结点的影响是不一样的，比如 v_j 在连接 v_i 之前杀毒，则 $v_{jt}\notin S(v_{it+1})$；而 v_j 在连接 v_i 之后杀毒，则 $v_{jt}\in S(v_{it+1})$ 就有可能。令 δ_{it}' 表示 v_i 在 t 到 $t+1$ 时段内传播后的查杀病毒变量。我们还是考虑 v_i 在 t 到 $t+1$ 时的状态改变量为：

$$(a_{i1}^t,a_{i2}^t,\cdots,a_{in}^t)\cdot((x_{1t},x_{2t},\cdots,x_{nt})^T+(\delta_{1t},\delta_{2t},\cdots,\delta_{nt})^T)+\delta_{it}'=x_{i(t+1)} \tag{2}$$

考虑到整个子网中所有结点的状态在时间 t 到 $t+1$ 时的变化，即为：

$$\begin{pmatrix}a_{11}^t & a_{12}^t & \cdots & a_{1n}^t\\ a_{21}^t & a_{22}^t & \cdots & a_{2n}^t\\ \cdots & \cdots & \cdots & \cdots\\ a_{n1}^t & a_{n2}^t & \cdots & a_{nn}^t\end{pmatrix}\cdot\left[\begin{pmatrix}x_{1t}\\ x_{2t}\\ \vdots\\ x_{nt}\end{pmatrix}+\begin{pmatrix}\delta_{1t}\\ \delta_{2t}\\ \vdots\\ \delta_{nt}\end{pmatrix}\right]+\begin{pmatrix}\delta'_{1t}\\ \delta'_{2t}\\ \vdots\\ \delta'_{nt}\end{pmatrix}=\begin{pmatrix}x_{1(t+1)}\\ x_{2(t+1)}\\ \vdots\\ x_{n(t+1)}\end{pmatrix}$$

上式可简化为：

$$A_t\cdot(X_t+\delta_t)+\delta'_t=X_{t+1} \tag{3}$$

其中 $A_t,\delta_t,\delta'_t,x_{t+1}$ 是已知，那么病毒寻源的问题即为求解向

量 X_t 的过程，即求解所有染病结点的前染结点 $P(x_{i(t+1)})$ 和每个结点在 t 时状态 x_t 的过程。若求出上述两点，那么再由 t 时的状态求得 $t-1$ 时的状态。上述过程重复下去，即可得到该子网的初始状态 X_0 和由感染结点与其前染点形成的传播链，传播链的始点即为病毒源。

3 求源方程的求解

经过上面的分析，我们知道子网中对病毒求源的过程，即是对状态方程(3)反复求解的过程。从某一时刻如 $t+1$ 求解时刻 t，再到 $t-1$，一直到初始时刻 t_0（即该子网中最初感染病毒的时刻），这正好是病毒传播的逆过程。为此，在解方程(3)时要解决两个问题：①尽量将所有结点尤其是染毒结点，依据其 $t+1$ 时的状态 x_{t+1} 解出 t 时刻的状态 x_t；②找出每一个感染结点的前染点 $P(x_{i(t+1)})$。由于方程(3)是一个布尔量和布尔运算构建的方程，基于清毒向量中有些结点 δ_{it} 或 δ'_{it} 是未知的，故不宜采用消元法来求解。而是依据具体的结点的连接去解，经过分析和反复实验，我们得到如下求解方法和经验。

第一步，由于在实际的网络中在每一时段参与连接的结点仅仅是完全图中的一小部分，换句话说，也就是 A_t 常常是一个稀疏的布尔矩阵，因此先将所有的接受向量为 0 的结点挑出来，影响它们的状态变化的只可能是清毒，而清毒时又可以得知它们的感染信息，其状态变化为：

$$x_{it+1}=x_{it}+\delta_{it}+\delta'_{it}$$

具体解法如下：若 $x_{it+1}=1$，则 $x_{it}=1$，故 $P(x_{it+1})=x_i$。即状态没有变化，前染点为自己；若 $x_{it+1}=0$，且若 δ_{it} 或 $\delta'_{it}=1$，则 $x_{it}=1$，否则 $x_{it}=0$，$P(x_{it+1})=0$ 此时，即无前染点。

第二步，挑出所有 $x_{it+1}=0$ 的结点来，因为此时 $P(x_{it+1})=\phi$，即无前染点，它们的状态变化如下：

$$\sum_{j=1}^{n}(a_{ij}^t(x_{jt}+\delta_{jt})+\delta_{jt}')+x_{it}+\delta_{it}+\delta_{it}'=0$$

对上式中 $a_{ij}^t=0$ 的结点由第一步已被删除，那么依据 $\delta'_{jt}=1$ 或 $\delta_{jt}=1$ 知 $x_{jt}=1$；若 $\delta'_{jt}=0$ 或 $\delta_{jt}=0$，则 $x_{jt}=0$。那么剩下的结点是 δ'_{jt} 或 $\delta_{jt}=\Theta$ 是结点，也就是状态不被改变的结点即为：

$$\sum_{j\in\{1,\cdots,n\}}x_{jt}=0$$

即只能是 $x_{jt}=0$。

第三步，经一、二两步处理后，剩余的为 $x_{it+1}=1$ 的结点，其状态变迁如下：

$$\sum_{j=1}^{n}(a_{ij}^t(x_{jt}+\delta_{jt})+\delta_{jt}')+x_{it}+\delta_{it}+\delta_{it}'=1$$

首先由第一步已将上式中 $a_{ij}^t=0$ 的那些项删除，只剩下 $a_{ij}^t=1$ 的项，即上式变化为：

$$\sum_{j=1}^{n}(x_{jt}+\delta_{jt}+\delta_{jt}')+x_{it}+\delta_{it}+\delta_{it}'=1$$

由 δ_{jt}'，$\delta_{jt}=1$ 得 $x_{jt}=1$；若 δ_{jt}'，$\delta_{jt}=0$，得 $x_{jt}=0$，此时 $x_{jt}\notin S(x_{it+1})$；最后只剩下 $\delta_{jt}'=1$ 和 δ_{jt}'，$\delta_{jt}=\Theta$ 即没有查毒的结点，它们的状态变迁为：

$$\sum_{j=1}^{n}x_{jt}+x_{it}=1$$

对于该式要么 $x_{jt}=1$，要么 x_{jt} 的状态不能确定。显然，项数越多，每项的状态就越难确定。此时可以由时间间隔的大小来调整项的多少，最好在 $\sum_{j=1}^{n}x_{jt}$ 中只保留一项，那么大部分 x_{jt} 和 $P(x_{i(t+1)})$ 就可以确定。当然，不能确定的情况也可能存在，形式如下：

$$x_{it+1}=x_{it}+x_{jt}=1$$

$$x_{jt+1}=x_{jt}+x_{it}=1$$

它对应实际网络中两个结点在同一时间段相互发送数据并且在该时段结束后都已染毒，我们称其为环态。从理论上讲，满足环态的解有 3 组：

$$\begin{cases}x_{it}=0,x_{jt}=1,P(x_{it+1})=x_{jt},P(x_{jt+1})=x_{jt};\\ x_{it}=1,x_{jt}=0,P(x_{it+1})=x_{it},P(x_{jt+1})=x_{it};\\ x_{it}=1,x_{jt}=1,P(x_{it+1})=x_{it},P(x_{jt+1})=x_{jt}。\end{cases}$$

但实际的病毒传播只能是其中之一。好在实际工作中，此种情况非常少，在我们的模拟实验中就没有出现过。但我们并不排除此种情况的发生。若在实际工作中出现了这种情况，它意味着在 t 时段起，直到开始追踪该病毒的那一刻，这些结点的状态没有因为清毒而改变。换句话说，也就是这些结点在此时段内没有清毒，那么我们就可以对这些结点查杀病毒。由于清毒工具可以记录计算机系统对于病毒的感染信息，比如感染的文件和感染时间等。于是就能够得到这些结点在 t 时刻的状态，从而解决此问题。

经过上述几步的处理，我们就得到了子网内所有结点 t 在时刻的状态 $X_t=(x_{1t},x_{2t},\cdots,x_{nt})$，同时也得到了所有被感染结点在 $t+1$ 时的前染点 $P(v_{it+1})$。那么我们就可以转入求解前一时段的状态方程组：

$$A_{t-1}(X_{it-1}+\delta_{it-1})+\delta_{i(t-1)'}=X_{it}$$

重复此过程，直到子网中所有被感染的结点的前染点不在本子网，即 $P(x_{it})=\phi$。那么 x_{it} 即为本子网内病毒传播路径的一个始点，x_{it} 就是子网内该病毒的一个源毒。一般说来，子网中的毒源不一定唯一，比如还可能存在另一个 $P(x_{jt})=\phi$，那么 x_{jt} 也是一个毒源。

4 求源的几点结论

如果将一感染结点 v_{it+1} 与其前染结点 $P(v_{it+1})=v_{jt}$ 组成一个二元组 $(v_{it+1},P(v_{it+1})=v_{jt})$，那么求解状态变迁方程的过程便是确定子网内所有被感染结点与其前染点二元关系的过程。将这些“首”“尾”相同的二元组排列起来，便得到一个求源路径，如：

$$(v_{i(t+1)},P(v_{i(t+1)})=v_{jt}),(v_{jt},P(v_{jt})=v_{k(t-1)}),\cdots,$$

$$(v_{l(t-1)},P(v_{l(t-1)},\cdots)=v_0)$$

简记为 $v_{i(t+1)}v_{jt}v_{k(t-1)}\cdots v_{l(t-1)}v_0$。$v_0$ 即为一个病毒源。子网内求源路径长度最长的长度即为求解方程重复的次数。该求源路径的逆序即为病毒实际的传播途径。这也是网络病毒不同于生物病毒传播求源的一大优点，原因是以现在的技术，许多生物病毒传播信息是不可记录的，而网络病毒传播信息是数字化的、是可以记录的。

如果子网内所有被感染的结点的求源路径收敛于一点 v_{i0}，那么 v_{i0} 即为该子网内某病毒的一源点。如果子网内所有被感染的结点的求源路径不收敛于一点，那么该子网的源点是不唯一的。

依据这些路径，可以得到一些更为有用的信息。比如可以利用一定时间内传播途径上某结点的出度的多少来衡量结点毒性的大小。如图 2 为模拟实验 1 中由求源方程解出的病毒的传播过程，其中最毒的结点要算 v_3，v_9；一定时间内重复次数最多的路径为最毒的路径等。有了这些信息，就会在病毒再次袭击时采取相应的措施，减少病毒对子网的危害。比如，切断最毒的路径，隔离或保护最毒的结点等，这也为网络毒的免疫提供了切实可行的依据[10]。

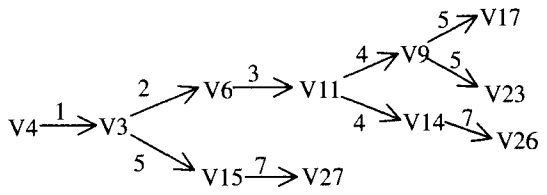

图 2　实验 1 中由方程解出的病毒的传播过程
（有向边上的权值为实验中的第几个时间段）

5　实验模拟验证

由于实验室内的机器均处在一个局域网内，它们的连接也不是应用层的连接。为了模拟一般的 Internet 环境下的子网，我们选择了 30 台电脑，并分别为它们开辟了新的实验邮箱。为了方便控制和信息采集，我们要求该邮箱不允许对外开放，并且每台电脑的用户可以自由地在剩余的 29 台电脑中选择通讯对象，那么这些用户的邮件通讯地址表便界定了一个逻辑子网。

为了不对机器造成危害，我们选择了 VBS 脚本病毒，并且限制该病毒的传播只能依靠 E-mail 附件的形式[附录 1]。

考虑到实际的收发邮件的频率不是太高，我们的时间间隔以天(24h)为单位，每次实验持续 8～10 天。在实验中，每个用户可以按自己的习惯查杀病毒。连接信息的采集通过查询用户的通讯日志和具体邮件的收发日志获得，日志的采集每天进行一次。为了节省空间，我们只列出被感染结点的状态变化。

表 1　只植入一个病毒源的求源表(表中数据为计算机编号)

Vi8	P(vi8)	P(vi7)	P(vi6)	P(vi5)	P(vi4)	P(vi3)	P(vi2)	P(vi1)
3	3	3	3	3	3	3	3	4
4	4	4	4	4	4	4	4	4
6	6	6	6	6	6	6	3	4
			δ=1	9	11	6	3	4
				δ=1	11	6	3	4
15	15	15	15	3	3	3	3	4
		δ=1	17	9	11	6	3	4
		δ=1	23	9	11	6	3	4
26	26	14	14	14	11	6	3	4
27	27	15	15	3	3	3	3	4
第 8 天	第 7 天	第 6 天	第 5 天	第 4 天	第 3 天	第 2 天	第 1 天	植毒日

从表 1 中可以看出，第四个结点 v_4 为该邮件子网的唯一毒源；v_9，v_{11}，v_{17}，v_{23} 分别在第 5、4、6、6 天采取了杀毒措施。

表 2　植入 2 个病毒源的求源表(表中数据为计算机编号)

Vi8	P(vi8)	P(vi7)	P(vi6)	P(vi5)	P(vi4)	P(vi3)	P(vi2)	P(vi1)
		δ=1	5	5	5	5	5	6
			δ=1	6	6	6	6	6
7	7	7	27	27	27	15	15	15
9	9	9	9	11	11	11	6	6
11		11	11	11	11	11	6	6
14	14	14	14	14	11	11	6	6
		δ=1	15	15	15	15	15	15
16	16	16	27	27	27	15	15	15
17	17	17	9	11	11	11	6	6
23	23	9	9	11	11	11	6	6
25	25	14	14	14	11	11	6	6
	δ=1	27	27	27	27	15	15	15
第 8 天	第 7 天	第 6 天	第 5 天	第 4 天	第 3 天	第 2 天	第 1 天	植毒日

从表 2 中可以看出，子网有两个毒源 v_6 和 v_{15}，并且 v_5，v_6，v_{11}，v_{15}，v_{27} 分别在第 6、5、7、6、7 天采取了杀毒措施，杀毒后也没有再传染。通过反复实验和求解，我们也得到了一些经验和结论：

①选择的时间间隔相对越小，方程求解越容易，但耗费的迭代过程也越多。

②当子网内病毒的感染率较高时，方程的求解过程较麻烦，耗时也较长，而感染率较低时，方程的求解相对容易些，而且耗时也较少。

③子网中查杀病毒的次数越多，方程的求解越容易。

④子网中结点间的连接率越低，方程的求解越容易。

因此，为准确求得毒源，子网中结点尽量及时查杀病毒，以便为方程的求解提供方便和足够的信息。

结论和致谢　本文指出网络病毒的传播信息是可以通过网络获取的；它不同于生物病毒的传播，病毒的求源也不同于严格意义上的反问题。本文将病毒源定义为：子网内病毒传播路径的始点。文章建立了子网的状态变迁方程，结合实际的网络因素和实践反复求解子网的状态变迁方程，即可求得病毒实际传播的路径，传播路径的始点即为病毒的一个源点。文章对模型及其解法给出了模拟实验，实验证明了理论模型的正确性和求解方法的有效性。文章为建立网络病毒的求源理论打开了一定的思路。

在该课题的研究过程中，我们得到了武汉大学计算机学院的孟庆树博士、甘肃省公安厅信息技术科的李南和甘肃省疾病防治中心的李慧等有关同志提供的病毒查询技术现状和建议；华中科技大学计算机学院 05 届的刘洋等同学实施了模拟实验。在此表示由衷的感谢。

参 考 文 献

1　Wang C, Knight J C, Elder M C. On computer viral infection and the effect of immunization. In: Proceedings of the 16th ACM Annual Computer Security Applications Conference, Dec. 2000

2　White S R. Open Problems in Computer Virus Research. Virus Bulletin Conference, Munich, Germany, Oct. 1998

3　Kephart J O, White S R. Directed-graph epidemiological models of computer viruses. In: Proceedings of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy, May 1991. 343～359

4　Kozlov I, Shilov A A, Kurmanova A G, et al. Recombination Origin of the Epidemic A/ussr/90177 Strians of Influenza. Dokl Akad Nauk SSSR, 1981, 257(3): 721～724

5　Karpukin G I, Golubev D B. Epidemiologic Aspects of the anthroponotic Concept of the Origin of Pandemic and Epidemic Strians of Influenza a Virus. Zh Mikrobiol Epidemic Immunobiol, 1983(5): 13～18

6　Han Lansheng, Liu Hui, Kojo A B. Analytic Model for Network Viruses. In: Proceedings of the ICNC2005, LNCS3612, Springer-Verlag, Berlin Heidelberg, 2005. 903～910

7　Zou C C, Towsley D, Gong Weibo. Email Virus Propagation Modeling and Analysis. [Technical Report]. TR-CSE-03-04, 2003

8　Tarantola A. Inverse Problem Theory and Model Parameter Estimation. Society of Industrial and Applied Mathematics(SIAM), 2004. 5～67

9　Aster R C, Borchers B, Thurber C H. Parameter Estimation and Inverse Problems. Elsevier Academic Press, 2005

10　Wang C, Knight J C, Elder M C. On computer viral infection and the effect of immunization. In: Proceedings of the 16th ACM Annual Computer Security Applications Conference, New York: IEEE Press, 2000. 246～262

（下转第 82 页）

关联。关联规则是寻找在同一个条件中的不同项的相关性，比如在一次购买活动中所购买不同商品的相关性。

关联规则挖掘的方法很多，其中最著名的是 Apriori 算法，其基本思想可分为两步：第一步是找出所有的频繁项目集：要求所有频繁项目集的支持度不低于设定的最小支持度；第二步是从所有的频繁项目集中挖掘出强关联规则：要求这些规则必须同时满足不低于最小支持度和最小置信度。其中，求所有频繁项目集的计算量最大。

由于入侵检测系统中的数据属于海量数据，用传统的求频繁项目集的方法效率很低，致使实时性不强，因此，本文用粗糙集理论对数据中的属性进行知识约简，删去冗余的属性，从中找出决策规则，再找出强关联规则。代替了传统的 Apriori 方法，使其效率、准确性都能得到提高。下面将会比较详细地介绍粗糙集的关联规则挖掘的思路以及其中所用到的一些粗糙集的方法。

3.1 构造决策表

把收集到的网络数据作为信息系统，比如一个连接记录包含以下属性：

(time, duration, service, src_host, dest_host, scr_bytes, dst_bytes, flag)

其中，time：表示连接开始的时间；duration：表示连接从开始到结束所经历的时间；service：即连接的应用协议如 WWW, FTP, DNS, Telnet 等；src-host：源主机；dst-host：目的主机；flag：连接状态标记，包括正常结束状态和连接请求被拒绝的状态。

本文把源主机端口，连接协议，连接开始的时间，经历的时间，目的端口，连接状态标记作为条件属性，连接的应用协议作为决策属性。每行表示一个连接记录，每列表示一个属性，这样就构成了一个关于网络数据流的决策表。

3.2 确定各属性的重要性

决策表中并不是每个属性都是同等重要的，有些对决策结果起主要作用，而有些是不重要的，判断属性重要性的基本思想是根据该属性对分类结果的影响，若去掉该属性对分类影响大，说明该属性是重要的，反之，是不重要的。设决策表中的条件属性和决策属性集分别为 C，D，对于任意属性的重要性[3]定义为：$sig^{Y}_{X-\{x\}}(x)=(|S_X(Y)|-|S_{X-\{x\}}(Y)|)/|U|$，其中 $S_X(Y)=Y^{(U/X)^-}=Y_{V\in U/X, V\subseteq Y}V$。由此，可以计算网络数据中的每一个属性的重要性，以便进行知识约简。

3.3 知识约简[3]找出决策规则

知识约简是粗糙集理论的核心内容之一。网络数据流决策表中并不是每个属性都是同等重要的，甚至其中有些是冗余的，这些属性会产生一些无用的规则。因此必须在保持知识库的分类能力不变的条件下，删除其中不相关或不重要的知识。在网络数据决策表中，如果重要性小于给定的最小重要性域值，说明该属性为不重要属性，从决策中删除。并且属性值完全相同的行也应合并为一行。经过约简，便得到约简后的网络数据流决策表，然后根据简化后的决策表，得出一系列的决策规则。

3.4 关联规则的产生

对得到的每个决策规则再计算其支持度和置信度，如果该规则不小于指定的最小支持度阈值和最小置信度阈值，就是强关联规则，保留下来，否则就丢弃。假设通过求属性重要性，删去不重要的属性 time, duration, flag 后，得到如下一条关联规则：src_host=202.96.7.5,dst_host=202.108.35.210→service=WWW,10,80，这条规则表示在所有的网络连接中有 10%符合源主机端口的 IP 地址为 202.96.7.5，目的主机端口 IP 地址为 202.108.35.210 且所提供的服务是 WWW。在源主机端口的 IP 地址为 202.96.7.5，目的主机端口 IP 地址为 202.108.35.210 的网络连接中，有 80%提供的是 WWW 服务。

结束语 粗糙集理论作为一种新型的数据挖掘工具，已经很好的体现出了它的优势，本文提出了基于粗糙集理论的关联规则挖掘算法，能比较迅速地挖掘出潜在规律，提高了入侵检测的性能。

参 考 文 献

1 周明全，吕林涛，李军怀. 网络信息安全技术. 西安电子科技大学出版社，2003

2 张云涛，龚玲. 数据挖掘原理与技术. 电子工业出版社，2004

3 张文修，吴伟志. 粗糙集理论与方法. 科学出版社，2003

4 周庆敏，李永生，等. 基于粗集理论的数据挖掘应用. 南京工业大学学报，2003(2)：44～47

5 宁玉杰，郭晓淳. 基于数据挖掘技术的网络入侵检测系统. 计算机测量与控制，2002，10(3)：189～191

(上接第 11 页)

附录 1 通过 Email 附件传播的 VBS 脚本病毒主要代码(仅用于实验没有危害)

```
Function mail_virus_test()
  On error resume next
  wscript. echo
  Set outlookApp = CreateObject("Outlook. Application") //创建
      //一个 outlook 应用对象
  If outlookApp = "Outlook"Then
     Set mapiObj = outlookApp. GetnameSpace("MAPI") //获取
        //MAPI 的名字空间
     Set addiList = mapiObj. AddressLists //获取地址表的个数
     For Each addr In addrList
        If addr. AddressEntries. Coun()0 Then
          AddrEntCount = addr. AddressEntries. Count //获取每
             //个地址表的
                                  // Email 记录
        For addrEntIndex = 1 To addrEntCount
        Set item = outlookApp. CreatItem(0) //遍历地址表的
                //Email 地址
        Set addrEnt = addr. AddressEnties ( addrEntIndex ) //获取
                //具体 Email 地址
        item. To = addr. Address //填入收件人地址
        item. Subject = "邮件病毒传播实验" //写入邮件标题
        item. Body = "收到此信不要担心，这仅是实验，对电脑没有
                                  危害"
                                  //真正病毒的破坏部分，
                                  这里仅是感染标记
        Set attachments = item. Attachment //定义附件
        Attachments. Add fileSysObj. GetSpecialFolder ( 0 ) &.
        "test. jpg. vbs"
        Item. DeleteAfterSubmit = Ture //信件提交后自动删除
        If item. To ( )"" Then
           Item. Send //发送邮件
           ShellObj. regwrite "HKCU\softerware\Mailtest\mailed"
           , " 1 "
                //病毒感染标记，以免重复感染
        Endif
        Next
        End if
        Next
        End if
End Function
```