基于频繁模式挖掘的 Internet 骨干网攻击发现方法研究*)

顾荣杰[1] 晏蒲柳[1] 邹 涛[2] 杨剑峰[1]

(武汉大学电子信息学院通信工程系 武汉 430072)[1] (北京系统工程研究院 北京 100101)[2]

摘 要 DDOS、蠕虫和病毒邮件已经成为影响骨干网络安全的3大主要因素,近几年来不断爆发的安全事件已经造成了巨额的经济损失。这些攻击具有贪婪性以及模式频繁重复的特点。本文对这3种方式分别进行了建模,提出了一种新的基于攻击行为模式分析的TIR模型,并提供了相应的快速挖掘算法。基于本文的方法在应用中能以较低的资源代价检测到未知的攻击并定位受害源。为提高算法的效率,本文提出了基于双页表结构的TIR攻击树构建方法,实验证明它能有效地提升信息采集速度。

关键词 大规模网络威胁,频繁模式检测,双页表 HASH 结构算法

An Adaptive Internet Backbone Traffic Anomalies Detection Algorithm Based on Frequent Pattern Mining

GU Rong-Jie[1] YAN Pu-Liu[1] ZOU Tao[2] YANG Jian-Feng[1]

(School of Electronic Information, Wuhan University, Wuhan 430072)[1] (Beijing Istitute of System Engineering, Beijing 100101)[2]

Abstract DDOS, worm and mass mailing have a significant growth recent years, which has endangered Internet security. Most attack like worm has the nature of aggressive, greedy and behavior pattern of self-similar. This paper proposed an attack behavior analysis based model, TIR model and devised a fast algorithm based on frequent pattern mining. It can effectively detect known or unknown threats with a low cost and has the ability to report the suspicious address. This paper also puts forward an effective algorithm to improve the real-time detection performance, namely 2-page hash table algorithm. As a result we developed a distributed system namely M-Detector based our theories.

Keywords Massive malicious activity, Frequent pattern mining, 2-page hash table algorithm

在过去30年里,从最初一个简单的实验网络发展到目前世界范围互联的全球性复杂异构网络,Internet 经历了一个高速增长的黄金时期。从个人生活、商业应用到军事行动无一不与网络相关,并且越来越依赖于网络的稳定性、安全性。近几年来,拒绝服务攻击、蠕虫造成了巨大的经济损失。2001年7月,红色代码蠕虫[1]在爆发后的1h里感染了250,000台计算机,直接经济损失达到了26亿美元。2003年1月,SQL Slammer 蠕虫[2]在最初的15min时间内直接导致了9.5～12亿美元的经济损失。与红色代码每37min翻一番的速度相比,SQL Slammer 蠕虫仅需要8.5s便完成同样规模的种群。对有线和无线运营商而言,保证服务的平稳、可靠和连续性是首要目标。随着越来越频繁的蠕虫爆发、大量拒绝服务攻击事件,Internet 正在面临日益严重的安全挑战,极大地威胁到了 Internet 上数据的私密性、完整性和服务的可用性。已经有学者做了相关的研究,以评估这些威胁所带来的严重后果[3]。根据美国 DARPA 著名的 PDR 安全模型(Protection-Detection-Reaction),安全保护人员的策略是尽可能缩短检测和响应时间,从而延长保护状态的时间。本文将寻找一种可行的办法,以尽可能地实时检测发生的攻击。

1 相关工作

许多理论已经应用于在骨干网上进行蠕虫和 DDOS 造成的流量异常检测,但这些理论往往过于局限于其统计模型,因此误报率也往往很高。S. Muthukrishnan 等人[5]研究了异常流量序列,试图通过寻找流量曲线上的偏离点来发现蠕虫爆发时间点。A. Lakhina 等人[6]尝试利用子空间(Sub-Space)理论来特征化 DDOS 和蠕虫攻击并应用于实时监测,但这种方法的缺点是特征是模糊、诲涩的,并且最重要的是无法验证该特征是否准确。与此类似的还有基于信号解析理论的分析[7]。K. Labib 等人[8]使用主成分分析方法(PCA)试图对蠕虫爆发进行特征门限的提取和建模,这种方法在用于检测时的简单、迅速是一个优点,但它存在同样的结果难以理解的问题。另一个问题是基于流量归并的异常流量检测无法还原出细节的信息,因此无法进行异常的定位。

基于检测效率、对知识库的依赖等考虑,常规的基于特征检测的 IDS 无法满足骨干网蠕虫和 DDOS 事件的检测要求。本文的目标是检测异常并进行定位。本文提出的系统将满足以下3个要求:

- 实时性。根据 PDR 模型,检测和应急响应时间必须是当蠕虫开始大规模传播的最短时间里完成,因此系统要求是实时的;
- 适应性。检测算法要求既能支持大规模的骨干网,也能应用于局域网上。能对多种网络拓扑提供弹性支持,具有较强的适应性;
- 低资源消耗。尽可能降低系统资源开销,通过改进算法,减少算法对系统 CPU 的占用时间,这也是实时性的要求;
- 能检测未知事件并能提供威胁定位。能检测威胁并报告被 DOS 攻击主机或者被蠕虫感染主机地址,以帮助 ISP 安全管理人员作出及时有针对性的应急响应。

本文将提出一种骨干网主要威胁的建模方法:TIR 模型(基于流量威胁兴趣度的关系模型),进而基于该方法建立一个适用于骨干网的分布式威胁检测系统。基于提高系统效率

*)国家自然科学基金项目(90204008)。**顾荣杰** 博士研究生,主要研究方向包括计算机网络通信、智能网络管理、骨干网络安全等;**晏蒲柳** 博士、教授、博士生导师,长期从事计算机网络通信、网络管理等方面的研究。

的考虑，本文研究了一种基于二级 HASH 页表的新的 IP 地址链表构建方法，加速系统的链表创建和搜索过程。

2 理论与结果分析

目前，Internet 骨干网面临两大主要威胁：DOS(DDoS)和蠕虫。从技术上划分，蠕虫又可细分为两大类：快速主动随机扫描型蠕虫与被动邮件型蠕虫病毒。它们拥有一些共同的特征，但在表现上各有所异。在本章的开始首先给出它们在本文中的定义，然后进一步讨论如何去检测它们。

2.1 威胁的定义和主要特征分析

定义 1 DDOS 攻击 DDOS 攻击即拒绝服务攻击，攻击者通过事先控制的多台网络主机同时对某一受害主机发起的某种特定模式的分布式洪水攻击。

在这种攻击中，DDOS 攻击的源 IP 地址信息往往经过伪造，这些信息对于追踪的意义不大。尽管 DDOS 有几种攻击方式[9]，但是对于目标端主机的监测而言，特征是相似的，即短时间内来自许多个源地址的数据包对同一个目标主机发起连接请求(见图 1)。

定义 2 蠕虫 蠕虫是一种利用远程主机的某种服务漏洞自动寻找缺陷主机并积极自我复制传播的一段攻击代码。它可以分为快速主动随机扫描型蠕虫与被动邮件型蠕虫病毒[10]。

其中，快速主动随机扫描型蠕虫根据自身内置的伪随机地址发生器迅速产生大量随机的目标 IP 地址，并主动进行扫描和攻击。理想条件下，该伪随机数能产生均匀覆盖 32 位 IPv4 地址空间的随机地址，因为这样才能使该蠕虫以最大限度地进行传播。被这种蠕虫感染的主机对外呈现以下特点：在一个较短的时间窗内产生大量重复模式(如某个或某几个端口)的对外请求(见图 1)。而被动邮件型蠕虫往往通过蠕虫的附件进行传播，当某一主机被感染以后，邮件病毒会驻留内存，并根据受害主机上的联系人和地址薄大量对外发送 SMTP 请求，而病毒体往往包含在邮件的附件里。邮件接收者不小心打开附件，就会激活病毒代码，成为新的受害者和扩散源(见图 1)。

2.2 威胁兴趣关系定义和攻击模式建模

一般而言，主动发起连接的一方往往包含着某种企图。源(发起连接者)和目标(响应连接者)之间的连接隐含着某种关系，本文称之为兴趣。在大规模的骨干网攻击中处处体现着这种兴趣(企图)关系。攻击也是这种兴趣关系之一。这 3 种攻击兼具的本质是贪婪性和攻击性，最重要的特征是当攻击大规模发生时会导致网络流量在瞬间产生显著的峰值。某个攻击过程可以被分解成许多个独立的原子行为。每一个原子行为代表着一种上面提到的兴趣关系。这样根据 3.1 节可以归结出 3 种原子行为模式，本文将其统称为威胁兴趣关系模型，简称 TIR 模型(Threats Interestedness Relation Model)。

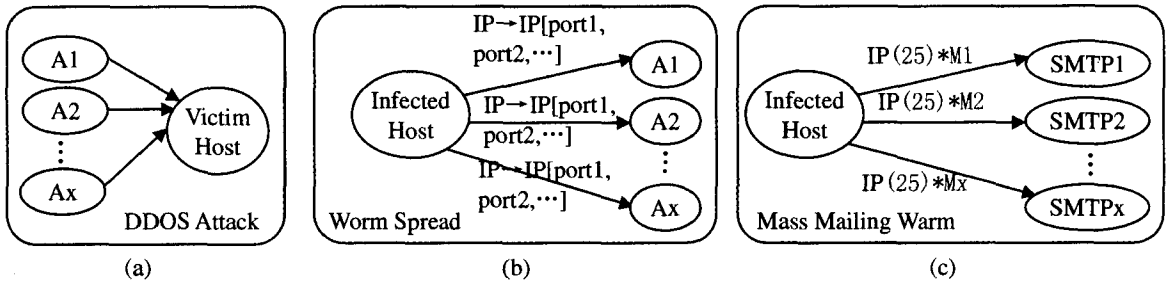

图 1 TIR 模型的攻击模式定义

表 1 TIR 模型的模式定义

名称	行为元模式	频繁模式
DOS	$SIP_{Attackx} \rightarrow DIP_{Victim}$ (UDP, TCP)	$\{SIP_x \rightarrow DIP_{Victim}$(UDP, TCP), $x=1,2,\cdots,N\}$
扫描蠕虫	$SIP_{Infected} \rightarrow IP_{random}$ (Port1, Port2, …)	$\{IP_{infected} \rightarrow IP_{dest\_x}$(Port1, Port2, …), $x=1,2,\cdots,N\}$
邮件蠕虫	$SIP_{Infected} \rightarrow DIP_{SMTP}(25)$	$\{SIP_{Infected} \rightarrow DIP_{SMTP_x}(25)*M\}$

TIR 模型描述了连接发起方和目标方之间的兴趣关系，揭示了攻击连接请求背后双方的本质关系。本文将其作为研究的切入点，是因为：a)它揭示的关系十分简单，当骨干网上发生大规模攻击时针对内容进行检测十分耗费系统资源，以兴趣关系(从数据包头直接提取 TCP/IP 四元组、协议类型，负载长度等)取代，对内容的分析更为省时、有效。事实上，频繁地自我重复是这几种攻击的典型特征。b)产生的随机地址可能是不可访问的，此时会话并不会建立，但 TCP SYN 请求仍可以监视到，传统的基于特征的检测方法将失效。图 1(a)显示了 DDOS 攻击中的兴趣关系。图 1(b)说明了一个受主动随机扫描型蠕虫感染的主机对外行为模式。基于本文以上的讨论，同一种蠕虫的对外行为模式具有自相似性。本文定义 $SIP_{Infected} \rightarrow DIP_{random}$(Port1, Port2, …)为随机扫描型蠕虫的原子行为模式，这样蠕虫的行为可以看作是这种模式的不断重复。图 1(c)表示受病毒邮件蠕虫感染的主机不断对外发送 SMTP 请求的模式。它的原子行为模式可以表示为 $SIP_{Infected} \rightarrow DIP_{SMTP}(25)$。下面对 3 种攻击模式进行总结，见表 1。

3 检测算法和结果分析

在本节里将要介绍上述攻击模式的检测算法。基于攻击模式检测方法，本文开发了一个攻击检测系统，称为 Malicious-Detector。

3.1 M-Detector 系统的架构设计

M-Detector 是一个双层结构的分布式系统，其架构设计如图 2 所示。系统由安全控制台和检测引擎构成。安全控制台是该系统的中心，负责对接收从检测引擎来的告警，同时协调系统中的其它安全组件做出响应，以击退到来攻击，恢复被攻击威胁的网络。检测算法在检测引擎上实现。一个或多个检测引擎根据网络的拓扑结构被部署于不同的子网出入口。每个引擎负责监视每个子网。引擎每隔为 5min，对过去的 5min 的数据进行分析检测，将检测到的攻击及相关 IP 地址生成告警，上报到控制台。这种分布式的结构处理使得系统有更好的适应性和伸缩性。

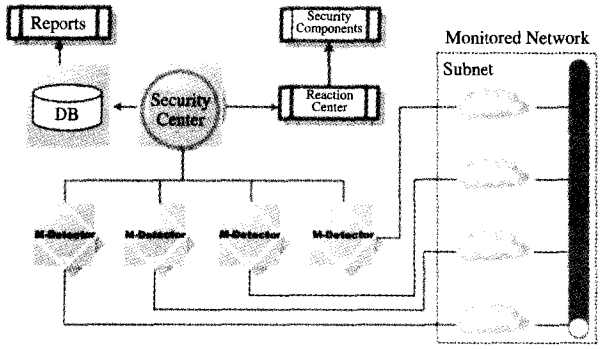

图 2　基于行为模式识别的检测系统架构图

3.2　DDOS 行为模式检测

根据资料分析，现在有多种不同形式的分布式拒绝服务攻击方式(根据发起方式和使用手段)，大多数拒绝服务攻击拥有在短时间内耗尽带宽的巨大威力。在以前的一次可控的 DDOS 攻击实验中，由其中一台实验主机发起的 FakePing 攻击产生了 26Mbytes/s 的流量，而这并不是危害能量最大的攻击。在分布式拒绝服务攻击中的许多数据包信息是经过伪造的，因此很难对分析追踪攻击源取证提供多大帮助。同时对巨量数据进行基于细节信息的分析将对分析主机和系统产生性能负担。根据上文的 TIR 模型，本文将重点放在对兴趣度关系的挖掘上，即对攻击模式的重复性进行分析。对于拒绝服务攻击，本文以目标 IP 地址为关注目标，统计被监测子网内每个被访问目标 IP 地址被访问的协议次数，将它与一个预先的设定阈值进行比较。如果发现有超过该阈值的 IP，则可以向控制台报告该 IP 以及相关的协议类型。

3.3　快速随机扫描型蠕虫的检测算法

表 2　近年来 CERT/CC 官方发布的蠕虫资料[11]

Worm	Affected Port	Vulnerabilities used	Target OS
Nimda	80,139,600	IIS, Code Red II and the backdoor left by Sadmind	Windows
Code Red I	80	IIS 4.0/5.0 Index Service	Windows
Code Red II	80	IIS 4.0/5.0 Index Service	Windows
Adore	23,53,111,515	Bind, LPRng, RPC. statd, wu-ftpd	Unix
Sadmind/IIS	80,111	IIS, Solstice, Sadmind	Win/Unix
Lion	53,10008	BINDservers	Unix
Ramen	27374	LPRng, rpc. statd, wu-ftpd	Unix(Redhat)
Cheese	10008	Backdoor left by Lion	Unix
Slapper	80,1433	OpenSSL, Apache	Unix
SQL Slammer	1433	Microsoft SQL Server	Windows
Witty	4000	ISS products(Black Ice, Realsecure)	Windows
MS Blaster	139,445, 69,4444	RPC	Windows

快速随机扫描型蠕虫根据协议可以划分成两大类：TCP 和 UDP 类型。到目前为止，已知的扫描型蠕虫大部分属于 TCP 类型，只有一小部分属于 UDP 类型，如 SQL Slammer 等。蠕虫是针对目标系统存在溢出漏洞的服务而设计存在的，同时由于每个版本的蠕虫代码是完全相同的，因此同一版本蠕虫的对外模式具有自相似性，攻击端口和基本的扫描行为都是一致的。根据近年来 CERT/CC 发布的蠕虫资料[11]总结成表 2。

为使处理更具针对性，并提高检测效率，本文对两类蠕虫采用不同的处理方法。

1)TCP 连接

一次 TCP 会话通常由一系列数据包构成。在整个包序列中，建立会话的每一个 TCP SYN 请求是最重要的，因为它提供最关键的兴趣关系信息。因此在实际中只捕获 TCP SYN 信息即可，这样将极大限度地降低采集的数据量，简化计算。根据表 2，一个蠕虫可能攻击一个或多个端口，以 Nimda 蠕虫为例，它会攻击 80 和 139 端口，它的模式可以被定义为 $SIP_{\text{Infected}} \rightarrow DIP_{\text{random}}(139,80)$。但对于未知蠕虫，无法知道什么端口以及哪几个端口会被组合攻击。本文的算法要求具备这样的能力，自动发现组合的攻击模式、检测到未知蠕虫。本文采用频繁模式挖掘算法以发现受蠕虫感染的主机。一般情况下，组合模式的项集数目不会超过 3 个，实际上 2 个已经可以精确定位蠕虫，考虑到系统的实时性效率要求，本文仅挖掘 3 项以下的组合攻击模式。由此本文的工作分为两个阶段：a)发现频繁的单端口扫描蠕虫；b)进而发现组合式端口的 2 端口扫描蠕虫；

• Step1　创建 TIR 攻击信息树。根据在过去 5min 里获取的会话信息建立 TIR 模型分析需要的攻击信息树。树结构如图 3 所示。在对快速随机扫描型蠕虫的攻击建模中，本文关注源 IP 地址。因此，TIR 攻击树中的第一层是对源 IP 地址进行链表构建。第二层将所有被访问的目标 IP 地址及相关的访问端口挂接在相应的源 IP 地址节点下。一般地，攻击树将被建模成图 3 所示结构。

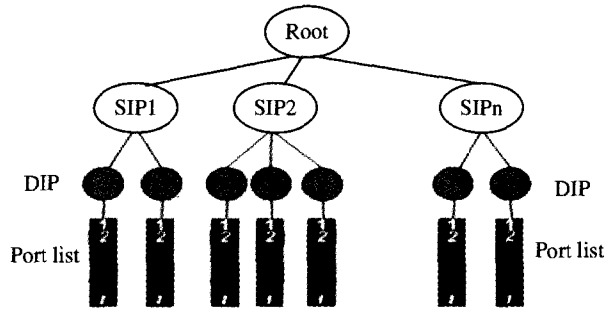

图 3　TIR 攻击信息树结构

Step2　TIR 树精简。令 T 代表 TIR 树中的源 IP 地址数目，从 TIR 树的一个 SIP 节点开始，对当前节点计算其访问过的独立目标 IP 个数。令该数值为 V，如果 $V < Thr_{\text{worm}}$ (Thr_{worm} 表示预设的一个阈值，以区别正常主机与蠕虫感染主机，根据实验 1000 是一个建议值)，保留该节点，否则将该节点从内存中释放，完成之后统计下一个 SIP 节点的独立目标 IP 数目，重复第二步，直到 SIP 链表结束；

• Step3　对精简后的 TIR 树的每一个节点，计算在本节点中出现的目标端口被访问的次数 $P_i = \text{Sum}(port_i), i=1,2,\cdots,n$。令该节点所有端口被访问总次数累计为 S，计算每个端口的访问比重：$Sup(P_i) = P_i/S, i=1,2,\cdots,n$。

• Step4　给定一个最小的访问比重 $Sup_{\min}$，将 $Sup(P_i)$ 与 $Sup_{\min}$ 比较并保留 $Sup(P_i) \geqslant Sup_{\min}$ 的目标 IP，从树结构中删除释放其它的目标端口节点。此处令最小访问比重 $Sup_{\min}$ 为 0.3。令 Q 为过滤后仍保留下来的节点集。如果 $Q=\Phi$ 为空集，则转至第三步，处理下一个源 P 节点。若 $Q \neq \Phi$ 为非空集，设函数 $R[Q]$ 代表 Q 中的集合数目，当 $R[Q]=1$ 时只存在

一次频繁端口，上报给安全控制台，转第三步；当 $R[Q]=2$ 时，存在唯一的二次频繁端口组合，上报给安全控制台，转第三步；当 $R[Q]=3$ 时，设集合 $Q=\{P'_1,P'_2,P'_3\}$，由此生成新的组合端口集合：$Q'=\{S_1,S_2,S_3\}=\{(P'_1,P'_2),(P'_1,P'_3),(P'_2,P'_3)\}$，转第 5 步进行处理。

• Step5 Count the visited times combination pattern 统计第 4 步中产生的组合模式被访问次数 $S_i\in Q'=\{S_1,S_2,S_3\}$。类似地，本文对每个访问组合计算访问比重 $Sup\{S_i\mid S_i\in Q'=\{S_1,S_2,S_3\}\}$，同时将其与预设的最小访问比重阈值进行比较，淘汰低于最小访问比重的组合。令 $Q''=\{S_i\}$ 为过滤后的组合端口集合。如果 $Q''\neq\Phi$ 为非空集，则说明发生了类似 Nimda 的二次以上端口组合的蠕虫攻击模式，然后将 IP 地址以及 Q'，Q'' 报告至安全控制台，之后转至第二步。如果 Q'' 为空，报告 IP 地址以及 Q' 报告至安全控制台，之后移动至下一个源 IP 节点，转第二步处理。在真实的网络环境中，某个 IP 的频繁端口集合数目应不超过 3 个，因此集合 Q' 的成员数目应该≤3，这样处理过程对算法的实时性效率没有任何影响。

2)UDP 连接

由于 UDP 是面向无连接的协议类型，它没有类似于 TCP 通信那样三次握手过程。数据包的头部没有任何信息显示某个包是序列的第一个，通信过程是完全在应用层进行协调的。因此本文将追踪所有 UDP 包，并关心那些源 IP 分布于被监测网段的数据包，以此为依据进行 TIR 树建模并进行与 TCP 相似的分析。

3.4 病毒邮件蠕虫的检测

病毒邮件蠕虫的检测是 3.3 节的一个特殊例子，差别是其仅需要监控目标 25 号端口。算法过程将大大简化，并且不需要检测 2 级以上频繁端口组合，此处不再展开描述。

3.5 数据分析

基于本文的实验模型已经部署在一个拥有 3500 台以上日常服务主机的政府电子政务骨干网上。本文通过分布在该网出入口上的核心交换机 Cisco Catalyst 6509，采用 Spanning 技术取得分析用的数据。图 4 是本文分析使用的数据的一个快照。

ID	SRCIP	DIP	PORT	TIMESTAMP
45	31.0.48.120	31.0.186.147	445	2005-1-18 17:42:12
46	31.0.48.120	31.0.86.106	445	2005-1-18 17:42:12
47	31.0.80.139	222.19.20.130	19997	2005-1-18 17:42:12
48	31.0.43.37	190.122.97.33	445	2005-1-18 17:42:12
49	31.0.43.37	58.25.226.95	445	2005-1-18 17:42:12
50	31.0.48.53	8.197.62.55	445	2005-1-18 17:42:12
51	31.0.48.53	14.130.100.188	445	2005-1-18 17:42:12
52	31.0.48.53	135.70.15.206	445	2005-1-18 17:42:12
53	31.0.48.53	147.139.140.221	445	2005-1-18 17:42:12
54	31.0.48.53	21.89.214.180	445	2005-1-18 17:42:12
55	31.0.48.53	151.222.247.114	445	2005-1-18 17:42:12
56	31.0.48.53	74.22.5.236	445	2005-1-18 17:42:12
57	31.0.50.130	213.185.52.7	445	2005-1-18 17:42:12
58	31.0.43.37	31.0.156.170	445	2005-1-18 17:42:12
59	31.0.48.53	19.122.203.5	445	2005-1-18 17:42:12
60	31.0.89.40	31.189.24.82	445	2005-1-18 17:42:12
61	31.0.50.130	31.86.135.213	445	2005-1-18 17:42:12
62	31.0.50.130	31.94.75.164	445	2005-1-18 17:42:12
63	31.0.48.120	185.180.200.106	445	2005-1-18 17:42:12
64	31.0.48.120	149.76.212.223	445	2005-1-18 17:42:12
65	31.0.48.53	31.158.75.11	445	2005-1-18 17:42:12
66	31.0.48.53	116.101.229.126	445	2005-1-18 17:42:12
67	31.0.80.139	219.221.98.60	17157	2005-1-18 17:42:12

图 4 从政务网上取得的原始数据快照(攻击)

经过基于本文算法的分析，本文得到了反映当时网络的一个攻击状态的结果，见表 3(对 4 个不同的采样间隔内的数据分析结果)。

表 3 政府骨干网上的检测结果(Cisco Catalyst 6509)，4 个不同采样间隔的数据分析

Threats Type	1st	2nd	3rd	4th
DDOS	0	0	0	0
Mass Mailing	31.0.54.10 221.6.6.4 192.168.100.254 31.0.36.243 31.0.70.58	31.0.54.10 221.6.6.4 192.168.100.254 31.0.36.243 31.0.70.58	31.0.54.10 192.168.100.254 31.0.36.243 31.0.70.58	31.0.36.243 31.0.54.10 192.168.100.254
Worm Infected	31.0.43.31(445, 603) 31.0.80.1(135,6500,) 31.0.48.99 (445,1659) 31.0.69.40 (445,1653) 31.0.46.1 (445,1652) 31.0.68.80 (445,1044) 31.0.50.130 (445,1527) 31.0.68.1 (445,1652) 31.0.43.37 (445,1653)	31.0.43.31 (445) 31.0.80.1 (135) 31.0.48.99 (445) 31.0.69.40 (445) 31.0.46.1 (445) 31.0.50.130 (445) 31.0.43.37 (445)	31.0.43.31 (445) 31.0.80.1 (135) 31.0.48.99 (445) 31.0.69.40 (445) 31.0.50.130 (445) 31.0.43.37 (445)	31.0.43.31 (445) 31.0.80.1 (445) 31.0.69.40 (445) 31.0.48.120 (445) 31.0.50.130 (445) 31.0.43.37 (445)

表 3 显示了本文检测到的网络中的主机被蠕虫感染的情况。由于当时的真实网络环境中没有拒绝服务攻击发生，因此尽管网络使用频繁但没有产生误报。在本文的结果中，地址 31.0.80.1 被 MS Blaster[11] 蠕虫感染，其它被检测到的主机如 31.0.43.31，31.0.50.130 等均被 Sasser 蠕虫[13] 感染。在检测过程中发现，4～5 台主机被感染了病毒邮件蠕虫，大量地向外发送邮件。

4 快速信息树构建方法：基于双页表 HASH 结构的构建算法

在实际运用发现，TIR 树的链表构建会不可避免地遇到效率的问题，尤其当系统运行于骨干网上时，由于独立的 IP 地址众多(理论上为 2^{32})，此时算法处理效率成为一个值得考虑的问题，传统的链表将无法满足这一要求。

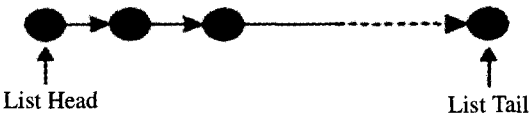

图 5 传统的链表处理方法

本文的问题具有两个特点：1)节点为 IP 地址；2)节点数目可能是巨大的。本文研究了一种快速的 TIR 树构建方法——基于双页表结构的高速 TIR 树构建算法。假设当前节点为 N 个，按传统链表处理方法，当一个新的 IP 地址到来

的时候，将最多搜索 N 次插入到链表中。这样，算法的处理复杂度为 $O(1+2+\cdots n)=O(\frac{n(n+1)}{2})$。当链表长度持续变长时，算法的时间复杂度以 N^2 级增长。根据本文的实验，当独立 IP 地址数超过 80,000 时，传统链表的处理时间将超过本文的处理时间窗口 5min，显然是不符合实时检测要求的。因此本文提出以下快速 TIR 树构建方法。

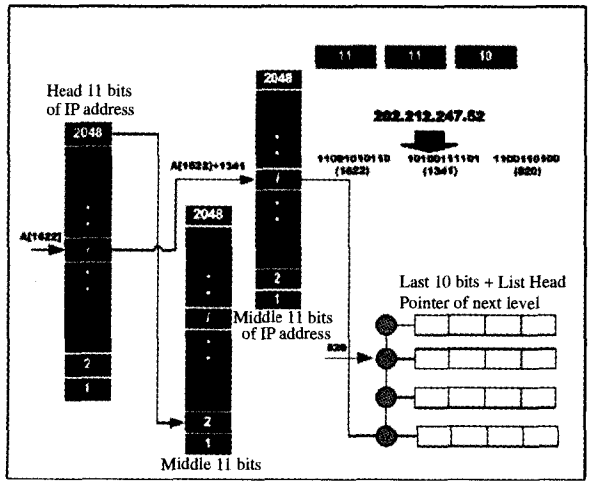

图 6　基于双页表结构的 TIR 树构造方法示意图

基于双页表的处理方法将一个 32 位的 IP 地址按位数拆成 3 段：头(11 位)、中间(11 位)、尾(10 位)。举例来说，如图 6，当新的 IP 地址 202.121.247.52 (0xCAD4F734H)到来的时候，本文将其分成 3 段，表示成二进制数为：(11001010110)-(10100111101)-(1100110100)。第一级页表本文用一个长为 2048 的静态数组来存放它，通过数组索引来访问静态数组的内容——即下一级页表的起始地址。这种静态定位方式省去了链表搜索的过程，直接定位到应该去的地方。IP 地址 202.121.247.52 拆成 3 段的十进制数表示为 1622,1341,820。首先通过 A[1622]定位第一级内容。如果有内容，则根据内容指向的地址直接访问该地址空间，定位第二级首地址，再根据索引号 1341 定位到相应的地址。如果发现第一级定位处的内容为空(NULL)，则申请一片 2048 的数组，将首地址存入 A[1622]，同理对第二级进行操作。最后根据低 10 位的 820 去建立或者访问已存在的链表，并将相应的信息写入链表。伪代码表示如图 7。

```
LET A=The first level array (Initialize to zero)
WHEN a new IP address W captured:
{H:=the first 11 bits of W,M:=the middle 11 bits of W,
 L:=the first 11 bits of W,LET B=content in A[H]+M;
   IF A[H] IS NOT null
   THEN{IF B IS NOT null
          THEN{IF B IS NOT null
          THEN{B is Linked List head,search the List B
                 IF node L is exist THEN fill it
                 ELSE insert a new node at tail;}
          ELSE new a linked list,Let B←List Head pointer;}
 ELSE new a array with size 2048,initialize to zero.
      new a linked list,Let B←List Head pointer;
}
```

图 7　双页表结构伪代码描述

这种方法避免了对长链表进行遍历的时间复杂度，将理论上的每次 2^{32} 搜索范围限制为 1024 次(实际更少)，大大提高了算法的处理效率。显然，骨干网上的 IP 地址越多，本文的算法效率体现越明显。图 8 是本文的实验结果，横轴为独立 IP 个数，纵轴为处理时间(单位为 ms)。可以见到，本文的算法效果是非常明显而有效的，完全能满足实时处理要求。

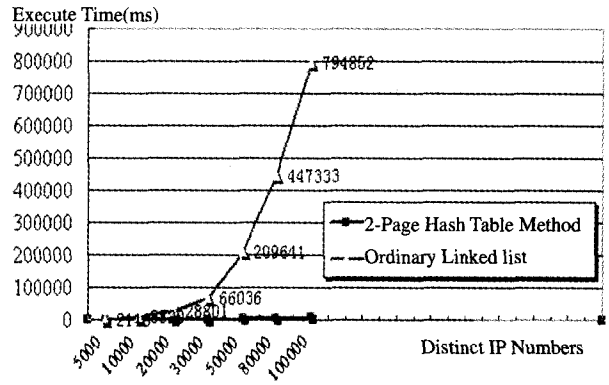

图 8　双页表结构算法与传统链表算法效率比较

总结与展望　本文分析了当前影响 Internet 安全的 3 大威胁，并从兴趣度关系的角度对其进行了针对性建模，提出了基于频繁模式的挖掘算法。根据本文的算法，设计实现了一个原型系统，并且以并联的方式部署于某政府电子政务骨干网上。本文的另一个贡献是针对大型骨干网上基于 IP 地址的海量信息处理，提出基于双页表结构的高速信息树构建算法，进而给出了在不同压力下算法的表现性能评价实验结果。结果表明，本算法应用于骨干网的实时检测时是十分有效的。

在本文的研究中，只使用了 TCP/IP 四元组、协议类型、TCP 标志等有限的变量信息。作为对将来研究的建议，可以将数据包负载部分的长度也纳入考虑指标，这是因为大量的 DDOS 和蠕虫攻击包在数据包的负载内容是重复一致的，因而长度也是一致的。作为反击蠕虫的一种方式，本文可以建立类似于 CAIDA 的 Network Telescope 项目组使用的 IP 地址黑名单。只要发现有主机对该名单进行了访问，即可判断是随机扫描蠕虫的行为。作为检测方法，这也是一种简单而有效的方法。

参 考 文 献

1　Shannon M D, et al. Code-Red: a case study on the spread and victims of an Internet worm. IMW, 2002
2　Moore D, Paxson V, et al. The Spread of the Sapphire/Slammer Worm. CAIDA, ICSI, Silicon Defense, UC Berkeley EECS and UC San Diego CSE, 2003
3　Lan Kun-chan, Hussain A, Dutta D. Effect of Malicious Traffic on the Network. In: Proceedings of PAM, 2003
4　Muthukrishnan S, Shah R, Vitter J S. Mining Deviants in Time Series Data Streams. 16th International Conference on Scientific and Statistical Database Management
5　Lakhina A, Crovella M, Diot C. Characterization of Network-Wide Anomalies in Traffic Flows: [Technical Report]. BUCS-2004-020. Boston University, 2004
6　Moore D, Voelker G M, Savages S. Inferring Internet Denial-of-Service Activity. Usenix Security Symposium, 2001
7　Barford P, Kline J, Plonka D, et al. A signal analysis of network traffic anomalies. Internet Measurement Workshop, 2002
8　Labib K, Vemuri V R. Detecting and Visualizing Denial-of-Service and Network Probe Attacks Using Principal Component Analysis. Third Conference on Security and Network Architectures. SAR'04, La Londe, France, June 2004
9　Hussain A, Heidemann J, Papadopoulosc C. A Framework for Classifying Denial of Service Attacks. In: Proceedings of ACM SIGCOMM, 2003
10　Weaver N, Paxson V, Staniford S, et al. A Taxonomy of Computer Worms. In: Proc ACM CCS Workshop on Rapid Malcode, October 2003
11　http://www.microsoft.com/security/incident/blast.asp
12　http://www.microsoft.com/security/incident/sasser.asp