ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.12, December 2007, pp.3060−3070
DOI: 10.1360/jos183060
© 2007 by *Journal of Software*. All rights reserved.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

高速网络监控中大流量对象的提取*

王风宇[1,2,3+], 云晓春[1], 王晓峰[4], 王 勇[1,3]

[1](中国科学院 计算技术研究所 信息智能与信息安全研究中心,北京 100080)

[2](山东大学 计算机科学与技术学院,济南 250101)

[3](中国科学院 研究生院,北京 100049)

[4](哈尔滨工业大学 计算机网络与信息安全技术研究中心,黑龙江 哈尔滨 150001)

Identifying Heavy Hitters in High-Speed Network Monitoring

WANG Feng-Yu[1,2,3+], YUN Xiao-Chun[1], WANG Xiao-Feng[4], WANG Yong[1,3]

[1](Institute of Computing Technology, The Chinese Academy of Sciences, Beijing 100080, China)

[2](School of Computer Science and Technology, Shandong University, Ji'nan 250101, China)

[3](Graduate University, The Chinese Academy of Sciences, Beijing 100049, China)

[4](Research Center of Computer Network and Information Security Technology, Harbin Institute of Technology, Harbin 150001, China)

+ Corresponding author: Phn: +86-531-88391099, E-mail: wangfengyu@sdu.edu.cn

Wang FY, Yun XC, Wang XF, Wang Y. Identifying heavy hitters in high-speed network monitoring. *Journal of Software*, 2007,18(12):3060−3070. http://www.jos.org.cn/1000-9825/18/3060.htm

Abstract: Due to the deficiency of traffic measurement capability in high-speed network, it's valuable for detecting large-scale network security incident to identify heavy hitters precisely in time. An algorithm of identifying heavy hitters based on two-level replacement mechanism is proposed in this paper. In this algorithm, LRU replacement and LEAST replacement are combined together to improve its accuracy. The heavy hitters can be identified accurately in small constant memory space, so the data can be treated more rapidly in limited space of SRAM. It's unnecessary to provide more memory space for more network data, so the algorithm is scalable.

Key words: network measurement; heavy hitter; replacement mechanism; anomaly detection

摘 要: 在高速网络环境下,由于受计算及存储资源的限制,及时、准确地提取大流量对象对于检测大规模网络安全事件具有重要意义.结合LRU淘汰机制和LEAST淘汰机制,建立了基于二级淘汰机制的网络大流量对象提取算法(LRU&LEAST replacement,简称 LLR),两种淘汰机制相互弥补不足,较大地提高了算法的准确性.由于算法占用存储空间较少,从而可以在有限的 SRAM 空间中更快地处理流量信息.该算法在网络数据量增加的情况下不必增加存储空间,具有很好的可扩展性.

关键词: 网络测量;大流量对象;淘汰机制;异常检测

中图法分类号: TP393 **文献标识码**: A

* Supported by the National Natural Science Foundation of China under Grant No.60573134 (国家自然科学基金); the Program for New Century Excellent Talents in University of China (新世纪优秀人才支持计划)

Received 2006-05-16; Accepted 2006-11-13

© 中国科学院软件研究所 http://www.jos.org.cn

网络流量监控是网络安全管理的重要方式,大部分大规模网络安全事件的检测都是通过网络流量的采集分析来完成的.目前,主要的流量异常检测系统都是以中小规模网络作为监控对象,如企业网络、校园网等.然而,对于当前一些大规模网络攻击,仅监控低端网络是不够的.以网络蠕虫爆发为例,它能够在 10 分钟甚至更短的时间内感染互联网内大部分的脆弱主机,并且由于产生大量的数据报文而导致网络被淹没甚至崩溃.因此,有必要在主干网络进行监控,以及时发现并有效控制这些大规模网络攻击.

很多研究通过总体流量的变化来发现异常,并不能定位异常的来源.要定位异常来源,不仅需要总体流量变化信息,更重要的是要知道导致流量异常变化的对象.网络监控中通常监控的对象类型主要有:目的 IP 地址、源 IP 地址、目的端口和协议、源端口和协议等[1].对相同关键值的对象进行流量累加计数产生流量信息可以有效发现异常,比如,统计不同目的 IP 地址的流量就可以定位 DDoS 攻击的目标.

然而,按照当前主干网络链路速度和流量,监控系统要同时统计几十万甚至上百万的对象流量,要完整统计这些信息,主机的存储和处理速度已经力不从心了,从算法改进的角度有两种提高数据处理效率的途径[2]:一是降低存储空间,使流量信息能够储存在访问速度更快的 SRAM 中;二是减少存储器访问次数.对于一些大规模网络安全事件,我们只需要关注大流量对象就可以了,例如 DDoS 攻击固定的目标地址,蠕虫扫描少数几个端口.有的网络研究扩充了传统五元组流的定义,把这些对象称作大流(large flow)[3],而在流数据(streaming data)研究中,大流量对象被称为 heavy hitter 或频繁项(frequent entry),本文中统称为大流量对象.

仅统计大流量对象可以节省大量存储空间,使流量信息能够储存在 SRAM 中,从而可以更加快速地处理网络数据;同时,由于剥离了无关网络流量,使安全事件的“罪魁祸首”更加容易识别.关键问题在于预先并不知道哪些对象是大流量对象,需要一种能够准确而高效地提取大流量对象的算法.本文结合 LRU(least recent used,最近最久未用)和 LEAST 淘汰机制建立了基于二级淘汰机制的大流量对象统计算法(LRU&LEAST replacement,简称 LLR),该算法能够准确地提取高速网络中的大流量对象.本文第 1 节介绍相关研究状况.第 2 节描述并分析基于二级淘汰机制的大流量对象统计算法.第 3 节通过实验对算法进行评估.最后对全文进行总结.

1 相关研究

当前最常用的 flow 级网络流量测量工具是 Cisco 的 NetFlow[4],它是在 RFC-2722[5]标准下工作的.NetFlow 把到达的包按照七元组归并为 flow,并将 flow 信息传递给上层监控系统.一方面存储全部 flow 信息要占用大量的存储空间;另一方面,向应用层传送日志信息对系统也造成了很大的压力.NetFlow 很容易造成网络数据转发的延迟和系统性能的降低.Sampled NetFlow 采用了周期采样的策略来减少内存占用和处理开销.周期抽样是一种最简单的抽样方式,每隔固定时间产生一次抽样.周期抽样简单且应用广泛,但它存在以下一些缺点:测量因具有周期性与很强的可预测性以及会使被测网络陷入一种同步状态.

抽样策略能够有效降低监控系统的负载,使系统具有很好的可扩展性,因此成为高速网络流量测量的研究热点[6−12].Estan 等人在文献[6]中提出的自适应采样方法通过动态调整采样率增强了 NetFlow 的可扩展性.文献[7]提出了一种非均匀抽样方法,在流测量设备输出流记录时通过对流记录的抽样来适应资源的限制.文献[8]根据 IP 报文中标识字段的随机性使用确定的掩码与标识字段的部分比特相匹配以实现分布式报文抽样,具有较好的随机性.文献[9]利用测量缓冲区对定长时间内到达的分组进行固定数量的抽样,既可以使抽样率自适应于流量变化,又可以控制资源的消耗.Kodialam 等人提出的 RATE 算法[10]通过对 two-runs(与流量数据中两个连续的样本匹配的 flow)的计数来估计流量对象的大小,该方法能够在较小的空间内准确估计大流量对象的大小,但对小流量对象的估计不够准确.文献[11,12]则分别通过对 RATE 算法中 flow 计数方法的改进,提高了算法准确性,降低了流量估计复杂度.抽样策略可以有效降低资源消耗,提高系统的测量能力,但难以满足需要较高准确度的应用.

采用 Hash 方法可以压缩流量对象信息空间,但难点在于如何从 Hash 结果推导出原始对象信息.文献[13]采用 SCBF(space-code Bloom filter)和最大似然估计能够较准确地估计高速网络线路流量对象的大小,算法计算效率高(可测量 OC-768 线路),但难以获得流量对象的完整信息.文献[14]使用带 Hash 增强算法的 Bloom Filter

© 中国科学院软件研究所 http://www.jos.org.cn

Reproduction 方法对 TCP 连接大规模异常的信息快速再现,使得在检测过程中无须维护 TCP 五元组信息,根据 Hash 结果可以较准确地还原 TCP 连接信息,能够以较少的资源占用和较高的准确性检测网络异常.文献[2]则提出了一种可逆 Hash 算法,能够通过少量的 Hash 比对找到原始对象,具有较高的空间压缩率.Hash 方法极大地减少了占用的存储空间,加快了处理速度,但压缩率和准确性之间需要找到一个平衡点,而且其占用的空间是随着网络数据量线性增加的.

Estan 等人第 1 次明确提出了在网络流量测量中采取“抓大放小(focusing on the elephants,ignoring the mice)”的策略[3],以放弃小流量对象信息为代价换取了存储空间和准确性上的优势,并提出了 Sample and Hold 和 Multi-Stage Filters 两种算法.Sample and Hold 算法是在周期采样基础上的改进,对于已被采样的 Flow,其后属于该 flow 的包均被统计,该算法实现简单,准确性有了较大提高.Multi-Stage Filters 算法采用多级 Hash 表对 flow 计数,只有每一级 Hash 表项均达到测量阈值的 flow 时才被正式记录,该算法占用空间小,能够准确识别出 Large flow.但由于 Hash 算法会把不同的 flow 映射到相同的键值,会高估 Large Flow 的实际流量,所以在 Flow 数量增长很大的情况下要适当增加 Hash 表的数量以降低碰撞率.文献[15]用 trie 树保存流量对象信息,并在节点对包的数量计数,当包数量超过一个阈值时则继续向下扩展 trie 树,该算法能够从不同层次估计一维和二维大流量对象,但 trie 树的存储代价比较大,而且流量估计的准确性难以保证.文献[16]提出了用类似于页面置换的 LRU 策略来识别流量大、持续时间长的 flow,然后对其采取惩罚或重路由等相应策略以控制拥塞,这种方法速度快,识别率较高,但该算法在测量时间段的后期会丢弃最近没有包到达的大流量对象,影响准确性.“抓大放小”策略不但可以节省大量存储空间,而且可以准确估计大流量对象,满足特定应用需求.

2 基于二级淘汰机制的统计算法

解决高速网络流量监控问题的途径可以归纳为 3 种:一是通过采样减少处理数据;二是利用 Hash 方法压缩空间,加快处理速度;三是通过淘汰小流量对象减少数据量.基于二级淘汰机制的统计算法采取的是第 3 种途径,该算法通过 LRU(最近最久未用)淘汰机制和 LEAST(最少)淘汰机制的结合,优势互补,实现了在固定大小空间内准确提取大流量对象.

2.1 算法描述

二级淘汰机制的基本结构是两个固定长度的对象列表.第 1 个列表采用 LRU 淘汰机制,最新有包到达的对象排在最前面,当列表满时,最久没有包到达的对象被淘汰,如果被淘汰对象的包的数量超过一个预设阈值 LEAST_LH(本文实验中取值为 2),则进入第 2 个列表,否则直接丢弃;第 2 个列表采用 LEAST 淘汰机制,即当列表满时丢弃包数量最少的对象,当该链表中的对象有包到达时,该对象重新进入第 1 级列表的最前面.后面我们把第 1 级列表称作 LRU_L,第 2 级列表称作 LEAST_L.

LLR 算法基本描述.

```
1.  初始化列表 LRU_L 和 LEAST_L;
2.  LRU_L 长度=L1,LEAST_L 长度=L2;
3.  包 P 到达,抽取对象识别字段,如{目的 IP 地址};
4.  if P 属于 LRU_L 中的对象 F1
5.          F1 的包计数加 1,并把 F1 放在 LRU_L 最前面;
6.  elseif P 属于 LEAST_L 中的流 F2
7.          F2 的包计数加 1,从 LEAST_L 取下 F2 加到 LRU_L 最前;
8.          F3=取下 LRU_L 最后一项,
9.  else
10.         为 P 建立一个新的对象,并放在 LRU_L 最前面;
11.         if LRU_L 项数>L1
12.                 F3=取下 LRU_L 最后一项;
```

© 中国科学院软件研究所 http://www.jos.org.cn

13. **endif**;
14. **endif**;
15. **if** *F*3 的包数量>预设阈值 *LEAST_LH*
16. 把 *F*3 插入 LEAST_L;
17. **if** LEAST_L 项数>*L*2
18. 丢弃 LEAST_L 中的最小项;
19. **endif**;
20. **endif**;

虽然减少存储空间可以在 SRAM 中更快地处理流量信息,但算法的计算复杂度仍然是影响处理效率的主要因素,需要提高网络数据包的处理效率.当一个包到达,首先要根据其协议字段在二级列表中查找相应的对象,我们对二级列表可以采取两种存储方案:一种是硬件上配合使用内容寻址存储器 CAM(content addressable memory),只需 1 次访存就可定位流量对象;另一种是软件上采用 Hash 表存储,处理 Hash 冲突需要少量额外的操作.因为 LRU_L 中的对象要不断调整位置,为了方便流量对象移动将其组织为双向链表结构.对 LEAST_L 的操作包括插入 LRU_L 淘汰的对象和删除对象等,可以把 LEAST_L 组织为二叉排序树以方便操作.图 1 为 LLR 算法的实现结构图.

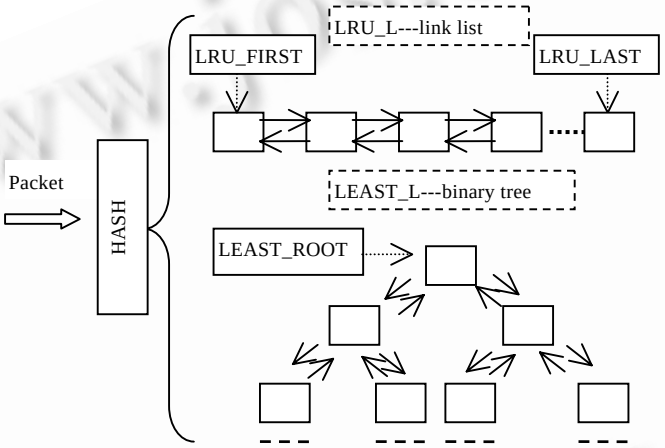


Fig.1 Structure of LLR algorithm

图 1 LLR 算法实现结构

2.2 算法准确性分析

以一个时间段(比如 1 秒、1 分钟或 1 小时)为统计范围,假设占总流量百分比 *PT*(例如 0.1%)以上的对象为大流量对象,LRU 链表长度为 *L*1,包总数为 *N*,对象 *A* 的包的总数为 N_A,*A* 占总流量的比例恰好为 *PT*.设大流量对象被从 LRU_L 淘汰出去的概率为 *P*(LRU),从 LEAST_L 淘汰出去的概率为 *P*(LEAST).

大流量对象被错误地从 LRU_L 淘汰出去的概率 *P*(LRU)小于等于 *A* 被淘汰的概率,也就是 *A* 在连续 *M*(*M*>*L*1)个报文中没有出现的概率,为超几何分布概率:

$$\binom{N_A}{0}\binom{N-N_A}{M}\Big/\binom{N}{M}<\binom{N_A}{0}\binom{N-N_A}{L1}\Big/\binom{N}{L1} \tag{1}$$

当 *L*1 相对于分组总数 *N* 很小时,上面的超几何分布概率可以用二项分布来近似计算,则有:

$$P(\mathrm{LRU})<(1-PT)^{L1} \tag{2}$$

图 2 是在 *PT*=0.1%的条件下根据式(2)计算的 LRU_L 长度和错误淘汰率的关系,随着 *L*1 在 1/*PT* 这一数量级上增加,错误淘汰率很快趋近于 0.

© 中国科学院软件研究所 http://www.jos.org.cn

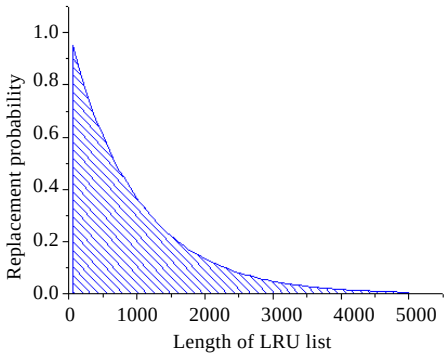


Fig.2 Relation between length of LRU_L and replacement probability of large flow (PT=0.1%)

图 2 LRU_L 长度和错误淘汰率的关系

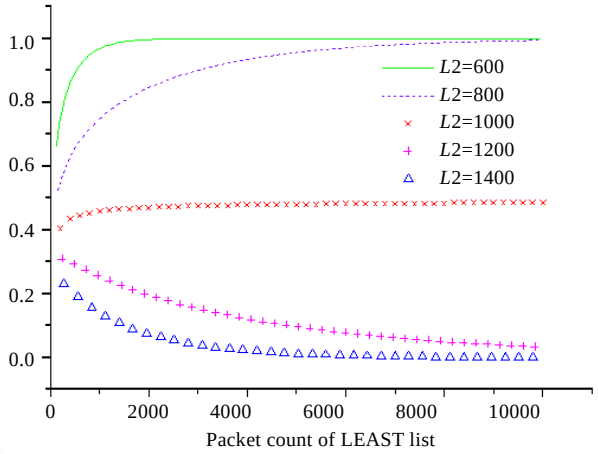


Fig.3 Relation between packet count of LEAST_L and replacement probability of large flow (PT=0.1%)

图 3 LEAST_L 的包数和错误淘汰率的关系

假设 LEAST_L 长度为 $L2$,A 进入 LEAST_L 时已经到达该列表的包的总数为 n,如果 A 被从 LEAST_L 中淘汰,则其中留下的每一个对象都应该比 A 已经到达的包数 n_A 要多,即 $n_1 > n_2 > ... > n_{L2} > n_A$.由于 $n_1 + n_2 + ... + n_{L2} + n_A < n$,所以 $n_A \times (L2+1) \le n$.也就是说,如果 A 被淘汰,必定有: $n_A \le \lfloor n/(L2+1) \rfloor$,其概率为超几何累积分布概率:

$$P\left(n_A \le \lfloor n/(L2+1) \rfloor\right) = \sum_{i=0}^{\lfloor n/(L2+1) \rfloor} \binom{N_A}{i} \binom{N-N_A}{n-i} \Big/ \binom{N}{n} \tag{3}$$

所以 A 被 LEAST 链表淘汰的概率 P(LEAST)上限为

$$P(\text{LEAST}) < \sum_{i=0}^{\lfloor n/(L2+1) \rfloor} \binom{N_A}{i} \binom{N-N_A}{n-i} \Big/ \binom{N}{n} \tag{4}$$

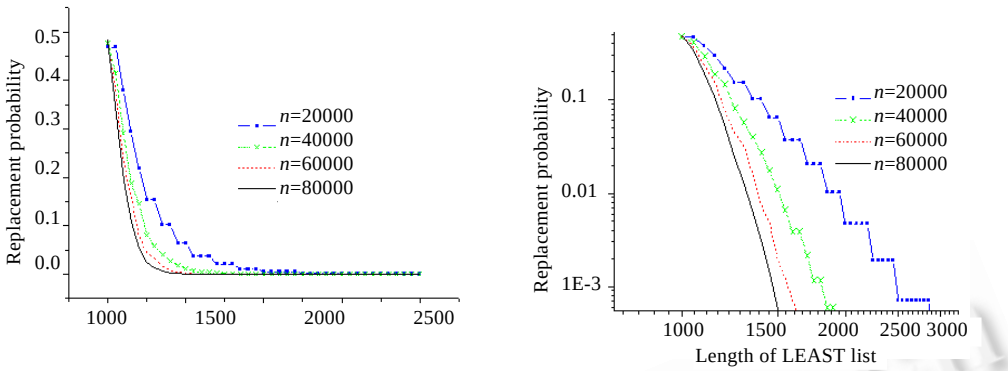
式(4)中有两个影响概率的因素:已经到达的包的总数 n 和 LEAST_L 的长度 $L2$,由于不能直观地从式(4)中得出它们与 A 被淘汰的概率之间的关系,我们通过样本值来观察.

图 3 为固定 $L2$ 的情况下,根据式(4)计算的随着 n 的变化 A 被淘汰的概率上限.可以看出,在 $L2=1/PT$(图中为 1000)时,错误率上限基本保持在 50%左右,小于这个长度,错误率上限会随着到达的包数快速上升,而 $L2>1/PT$ 则可以使错误率上限随着包数的增加快速下降,所以要求 $L2>1/PT$.图 4(a)为固定 n,随着 $L2$(>=$1/PT$)的变化 A 被淘汰的概率,图 4(b)是其 log-log 图,可以看出,图 4(b)中的曲线接近直线,所以 A 被淘汰的概率和($L2-1/PT$)之间的关系可以近似为幂函数衰减关系.然而这里对 LEAST 淘汰策略的分析还没有考虑 LRU_L 的作用.通过前面对 LRU 淘汰机制的分析我们知道,只要设定适当的 LRU_L 长度,大流量对象被从其中淘汰的概率可以降到很低,也就是 LRU_L 能够容纳大部分的大流量对象,所以 LEAST_L 的长度 $L2$ 可以缩减很多.假设 LRU_L 容纳了($plru$)%的大流量对象,那么 $L2$ 只要大于(1−($plru$)%)×($1/PT$)即可.

更为重要的是,两级淘汰机制形成一种互补.假设单独使用 LRU_L,如果大流量对象集中于时间段的前半部分,则该对象可能在时间段的后半部分由于很少有包到达而被淘汰.而在二级淘汰机制下,由于大流量对象的数据总量比较大,会被保存在 LEAST_L 而避免被错误淘汰;假设单独使用 LEAST_L,如果大流量对象前期到达包很少,后期才开始集中到达,则该对象可能在前期由于自身包的数量较小而被淘汰,到了后期,由于 LEAST_L 已满,不论后面到达多少包都会被淘汰.而在二级淘汰机制下,LRU_L 就可以容纳这种对象,避免其被错误淘汰.要使对象 A 最终从二级列表中淘汰,就要首先使其从 LRU_L 淘汰,然后从 LRU_L 淘汰,所以最终被淘汰的概率为

© 中国科学院软件研究所 http://www.jos.org.cn

$$P=P(\text{LRU})\times P(\text{LEAST}).$$



(a) Relation between length of LEAST_L and replacement probability of large flow
(a) LEAST_L 错误淘汰率与长度之间的关系

(b) Relation between length of LEAST_L and replacement probability of large flow(log-log)
(b) LEAST_L 错误淘汰率与长度之间的关系(log-log)

Fig.4
图 4

2.3 算法复杂度分析

2.3.1 时间复杂度

当一个数据包到达,首先要做的是查找相应的流量对象,无论是采用 CAM 方案还是 Hash 方案,其计算复杂度均为 $O(1)$.随后的操作包括流量对象在双向链表和排序二叉树中的移动、LRU_L 淘汰的流量对象插入 LEAST_L 以及在 LEAST_L 中查找最小流量对象等.对象的移动是有限常数次指针操作,其计算复杂度为 $O(1)$,而二叉树的查找定位操作最复杂,时间复杂度为 $O(\log(L2))$,$L2$ 为 LEAST_L 的长度.因此 LLR 算法处理一个数据包的时间复杂度为 $O(\log(L2))$.

2.3.2 空间复杂度

通过第 2.2 节的分析我们知道,只要满足 LRU_L 长度 $L1>1/PT$(PT 为大流量对象阈值)的要求,随着空间增加,准确性会快速提高,在$(1/PT)$这个数量级内,错误淘汰率就已经接近于 0,所以 LRU_L 的空间复杂度是 $O(1/PT)$.同样,通过第 2.2 节的分析知道,如果 LRU_L 容纳了($plru$)%的大流量对象,那么,LEAST_L 的长度 $L2$ 只要大于$(1-plru\%)\times(1/PT)$即可,因此 LEAST_L 所需空间比 LRU_L 低了一个数量级.用于定位流量对象的 Hash 表长度和表中的记录数(即二级列表长度)是同一个数量级,空间复杂度同样是 $O(1/PT)$.所以 LLR 算法空间复杂度为 $O(1/PT)$,PT 为大流量对象阈值.

2.3.3 访问存储器次数

除了存储空间,算法是否能够在线处理高速网络数据的另一个重要因素是最坏情况下处理一个数据包需要访问存储器的次数.要准确计算算法访问存储器次数比较困难,为了使问题简化,我们以算法对二级列表中的表项的访问次数作为访问存储器次数.表 1 是算法在不同情况下处理 1 个数据包的访存次数,从左到右是完整的处理流程,每个操作后用括号标出了访存次数,最右列为不同流程总的访存次数.如果被 LRU_L 淘汰的对象需要插入 LEAST_L,则还要淘汰最小的对象,需要在二叉树上执行两次定位操作,这种情况下操作最为复杂,最多需要 $8+2\times\log(L2)$次存储器访问,$L2$ 是 LEAST_L 的长度.在随机情况下,LEAST_L 二叉树平均查找长度约为 $\log(L2)$;极端情况下,二叉树蜕变为单支树,平均查找长度为$(L2)/2$,最大查找长度为 $L2$.由于插入 LEAST_L 的流量对象大小是随机的,所以蜕变为单支树的情况发生的可能微乎其微,查找长度可以近似为 $\log(L2)$.以大流量对象阈值 $PT=0.001$ 为例,假设 LRU_L 的长度设定足够保留 90%的大流量对象,则根据前面的分析,$L2>100$ 即可,所以算法的最复杂操作大约要访问 22 次存储器,一次 SRAM 访问时间最少约为 5ns[12].在 2.5Gbps(OC-48)网络环境下,假设平均包长度是 1 000bit,则满负荷状态下大约每 400ns 到达一个数据包,因此该算法的复杂度理论上

© 中国科学院软件研究所 http://www.jos.org.cn

可以满足 2.5Gbps 网络线路的在线测量需求.在第 3 节中我们将通过模拟实验检测 LLR 算法的处理速度.

当有包到达时,PSH(packet sample and hold)算法[1,3]只需要一次查找定位,访存次数最少;Multi-Stage Filter 算法[3]则要访问 $\log_{10}(n)$次 Hash 表和 1 次大流量对象表,n 为测量时间段内流量对象的总数,随着网络数据量的大量增加,要保证算法准确性就要增加存储空间和访存次数;在涉及到对 LEAST_L 二叉树操作时 LLR 算法访存次数较多,但无须随总数据量变化而增加,其唯一影响因素是 LEAST_L 的长度.表 2 是 LLR 算法访问存储器次数和 PSH 以及 Multi-Stage Filter 算法的比较.

Table 1 Times of accessing memory of LLR

表 1 LLR 算法访问存储器次数

Search the flow entry which the packet belongs to in two lists. (1)	If the flow entry $F1$ is found	If $F1$ is located in LRU_L	1. Modify flow entry $F1$.(3) 2. Modify the pointer of the first one in LRU_L.(1)		5
		If $F1$ is located in LEAST_L	1. Delete F1 from LEAST_L.(log($L2$)) 2. Modify flow entry $F1$.(2) 3. Discard the last one of LRU_L $F2$.(1) 4. Insert F2 into the LEAST_L.(log($L2$)+2)		2×log($L2$)+6
	If no flow entry is found	1. Create a new flow entry.(1) 2. Modify the firt one of LRU_L.(1) 3. Discard the last one of LRU_L $F2$.(1)	If $F2$ > LEAST_LH	1. Insert $F2$ into LEAST_L.(log($L2$)+2) 2. Delete the minimum flow entry of LEAST_L.(log($L2$)+2)	2×log($L2$)+8
			If $F2$ < LEAST_LH	1. Discard $F2$.(1)	5

Table 2 Times of accessing memory of different algorithms

表 2 不同算法访问存储器次数

Algorithm	LLR	Multi-Stage filters	PSH
Min memory accesses	5	$\log_{10}(n)$	1
Max memory accesses	8+2×log($L2$)	1+$\log_{10}(n)$	1

2.4 流量特征对算法的影响

前面第 2.2 节用概率理论分析了 LLR 算法列表长度和准确性之间的关系,给出了随着列表长度的变化,错误淘汰大流量对象概率的上限.而网络流量的分布并非完全随机的,这可能会对算法产生较大影响.研究表明,网络流量数据的分布具有少量流量对象占有大部分流量的类幂律特征[17,18].如果流量比较平均地分布在不同对象,则难于识别大流量对象;反之,如果大部分流量集中于少数的对象中,LLR 算法识别大流量对象相对容易,在固定存储空间下,能够达到更高的准确性,或者说达到相同的准确性需要更少的存储空间.

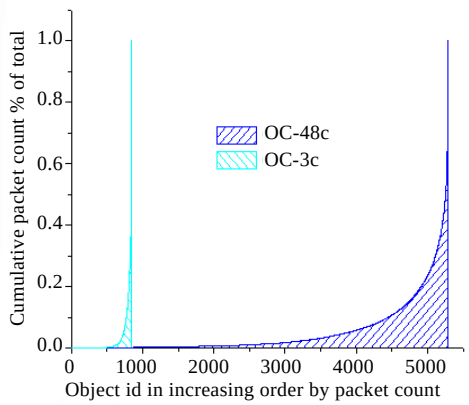


Fig.5 Cumulative distribution probability of traffic volume

图 5 流量的累积分布概率

我们用 NLANR 测量的两组数据[19]对网络流量的分布特征进行实例分析,一组是 Auckland 大学校园网出口 OC-3c 线路的单向流量 Auckland-Ⅳ,使用的是 2001 年 2 月 21 日全天数据集,另一组是 OC-48c 线路 IPLS-CLEV 的单向流量 Abilene-Ⅰ,使用的是 2002 年 8 月 14 日 10:00 测量的 10 分钟的数据集.本实验对两个数据集分别取前 20 万网络数据包,采用{目的 IP 地址}定义对象,其他定义方法的结果比较近似,本文不另作分析.把全部对象按从小到大排序,计算对象占总流量比例的累积分布,结果如图 5 所示.图 6 是不同长度的流量对象占对象总数比例的累积分布图.可以看出,少数的大流量对象占据了总流量的绝大部分.

少数的大流量对象占据了总流量的绝大部分,意味着对大部分包都能够在二级列表中直接找到相应的对

© 中国科学院软件研究所 http://www.jos.org.cn

象;大部分流量对象都很小,在图 7 中可以看到,仅 1~2 个包的对象占了对象总数的 40%还多.在实际应用中,我们可以设定一个阈值 *LEAST_LH*(例如 2),包的个数不高于 *LEAST_LH* 的对象从 LRU_L 淘汰的时候不必插入 LEAST_L,可以直接丢弃.因此只有在少数情况下需要执行复杂度为 $O(\log(L2))$的向 LEAST_L 插入对象的操作.

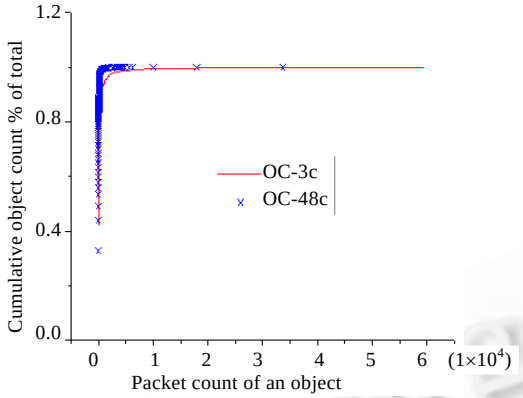


Fig.6 Cumulative distribution probability of object size

图 6 对象长度的累积分布

3 实验与评价

3.1 算法处理速度

在第 2.3 节中我们分析了 LLR 算法的访问存储器次数,并由此估计算法能够满足 2.5Gbps 网络线路的在线测量,下面通过模拟实验来测试 LLR 算法的实际处理能力.测试环境的配置为:CPU-PⅣ2.0G;4G 内存;RedHat Linux 7.2-2.4.18 操作系统.实验数据采用 NLANR 测量的 OC-48c 线路 IPLS-CLEV 的单向流量 Abilene-Ⅰ[19],采用的数据集是在 2002 年 8 月 14 日 09:00 测量的 10 分钟数据.用{目的 IP 地址}定义流量对象.LLR 算法中用长度为 64K 的 Hash 表来定位流量对象,并用链地址法处理 Hash 冲突,LRU_L 的长度为 2000,LEAST_L 的长度为 200,大流量对象阈值为 0.1%,*LEAST_LH*=2.每次实验预先把不同数量(1Gbit~ 10Gbit,1G=10^9)的网络数据的摘要信息全部读入内存,然后在此基础上用 LLR 算法测量大流量对象.图 8 为处理速度实验结果,由于不同网络线路平均报文长度相差较大,图中各点旁标出了该点对应的网络流量中包含的报文数量(如 2Gbit 流量包含 354 032 个报文).实验结果近似为一条直线,这说明 LLR 算法处理单报文的平均速度不会随着数据量的变化而变化.实验中处理单个报文的平均时间开销约为 65ns,这个处理速度完全可以满足 2.5Gbps 线路的在线测量需求,甚至可用于 10Gbps 网络线路的测量.

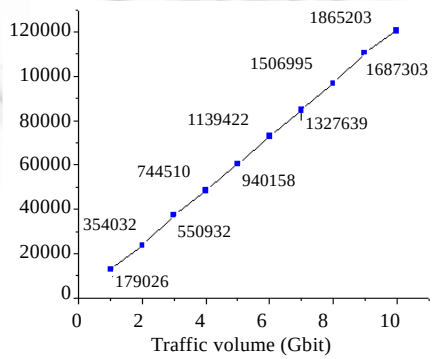


Fig.7 The process speed of LLR

图 7 LLR 算法处理速度

© 中国科学院软件研究所 http://www.jos.org.cn

3.2 算法准确性

基于二级淘汰机制的大流量对象统计算法在特定情况下会产生误差,如果某个大流量对象开始时到达的包比较少且分散,则会被淘汰掉.随后有两种可能:一种可能是随后又有大量的后续包到达而被保留下来,但对象的测量流量低于实际流量;还有一种可能是经过多次被淘汰,该对象因最后被统计的流量很小而被彻底淘汰.假设在一个时间段内有 J 个对象的流量超过了阈值,其中包的数量分别为 $N_j(j=1,2,\ldots,J)$,在用基于二级淘汰机制的大流量对象统计算法测得的结果中,相应的对象大小为 $M_j(j=1,2,\ldots,J)$,如果测得的结果中没有相应的对象,则 $M_j=0$.我们用两个尺度来衡量算法的准确性:

平均相对误差率(ARER):
$$\mathrm{ARER}=\sum_{j=1}^{J}\frac{|N_j-M_j|}{N_j\times J}.$$

错误淘汰率(FN):被二级淘汰算法错误丢弃的大流量对象的比例.

实验数据采用 NLANR 在 2002 年 8 月 14 日 09:00 测量的主干网 OC-48c 线路 IPLS-CLEV 的单向流量 Abilene-Ⅰ[19],数据总长度为 10 分钟,实验分别以网络异常检测中常用的{目的 IP 地址}和{目的端口,协议}为例定义对象,大流量对象的阈值 *PT* 设为 0.1%,*LEAST_LH* 设为 2.

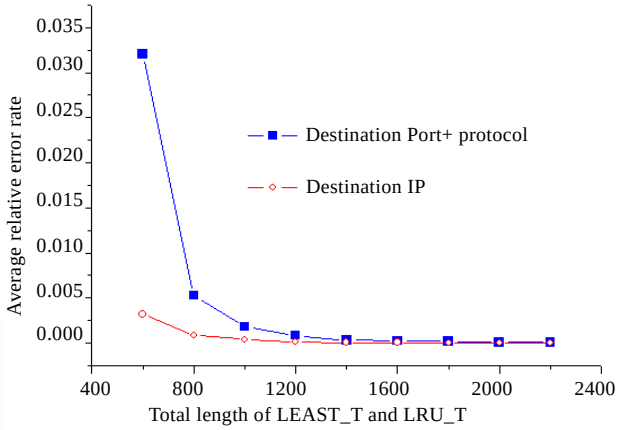


Fig.8 Relation between total length of lists and average relative error

图 8 列表的长度和平均相对误差之间的关系

LRU_L 和 LEAST_L 的长度是影响准确性的重要因素,前面我们通过分析给出了对二级列表长度的大致要求,下面通过对实际数据的统计观察随着列表总长度的变化,平均相对误差率的变化.这里把列表总长度简单地平均分配给 LRU_L 和 LEAST_L,统计数据包的量为 100 万.图 9 是实验统计的结果,平均相对误差率随着列表长度的增加而快速降低,在总长度达到 1 200 时基本达到了最佳效果.该实验中没有出现淘汰大流量对象的现象.

随着网络带宽的不断提高,同一时间间隔内到达的数据量会不断增加.在满足算法基本要求的情况下,算法的准确性不会随着数据的大量增加而降低,才能保证算法的可扩展性.我们把 LRU_L 和 LEAST_L 的长度均固定为 1 000,不断增加网络数据量,结果如图 10 所示,随着测量的网络数据包数量的增加,平均相对误差率并没有随之增加的趋势.该实验中也没有出现错误淘汰大流量对象的现象.

前面的准确性实验中,我们把存储空间平均分配给了 LRU_L 和 LEAST_L,而 LEAST_L 的长度对算法访问存储器次数会产生直接影响,如果要降低访问存储器次数,需要减少 LEAST_L 长度,下面通过实验检测 LEAST_L 长度对算法准确性的影响.二级列表的总长度设为 2 000,LEAST_L 的长度取值从 100~1000,其余的存储空间分配给 LRU_L,测量时间间隔为 20s.实验结果如图 11 所示,在 100~1000 的变化范围内,测量准确性没有受到明显影响,因此,我们可以通过适当调整 LEAST_L 的长度来降低算法计算量.

实际应用中 SRAM 的容量有限,下面我们在固定的存储空间内比较二级淘汰算法和常用的 Packet Sample and Hold(PSH)算法以及 Multi-Stage Filters 算法[1,3]的准确性.同时,为了证实 LRU 淘汰机制和 LEAST 淘汰机制的互补性,在比较中加入了单独的 LRU 淘汰算法和单独的 LEAST 淘汰算法.实验数据是 3 组 OC-48c 线路 60s 的流量,用{目的 IP 地址}定义对象.我们把存储空间限制为 1Mbit,这个大小可以适应大部分系统的 SRAM 存储器.二级淘汰算法中表项数为 4000,采用了两种长度设定:一个实验中 LRU_L 和 LEAST_L 长度均为 2 000;另一个实验中 LEAST_L 长度为 200,LRU_L 长度为 3 800.单独 LRU 和单独 LEAST 的表项数均为 4 000.PSH 算法的采样率为 1/1000,最终结果流量对象的数量约为 5000.Multi-Stage Filters 算法散列表数量为 4,每个散列表有

© 中国科学院软件研究所 http://www.jos.org.cn

2 647 个计数器,2 773 个表项用来存储大流量对象[3].除了比较在阈值 0.1%以上的准确性以外,还分别在0.1%~0.01%和0.01%~0.001%做了准确性比较,以对算法性能有比较全面的了解.统计结果见表3,与其他几种算法相比,在固定存储空间内,基于二级淘汰机制的大流量对象统计算法的准确性有明显的优势,而增大LEAST_L 的长度对较大流量对象(0.1%~0.01%和 0.01%~0.001%)的测量准确性的影响较大.相同长度的单独的LRU 淘汰算法和单独的 LEAST 淘汰算法准确性低很多,证实二级淘汰机制实现了两种淘汰算法的互补.

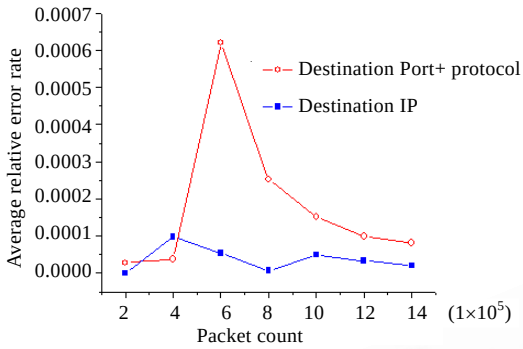


Fig.9 Relation between packet count and average relative error

图 9 平均相对误差率和测量数据量之间的关系

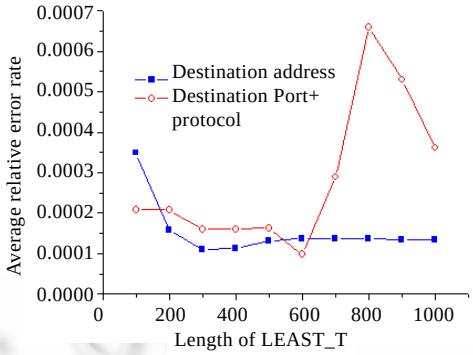


Fig.10 Relation between length of LEAST_L and average relative error

图 10 平均相对误差率和 LEAST_L 长度的关系

Table 3 The accuracy of different algorithms

表 3 算法准确性的比较

Category	>0.1%		0.1%~0.01%		0.01%~0.001%	
	ARER (%)	FN (%)	ARER (%)	FN (%)	ARER (%)	FN (%)
LLR (2000,2000)	0.0016	0	0.0192	0	5.877	3.551
LLR (3800,200)	0.012	0	0.56	0	62.45	40.56
LRU	11.78	4.41	23.53	10.47	70.81	44.31
LEAST	3.7	3.68	6.47	6.39	31.8	31.75
PSH	4.75	0	21.2	0.2	58.31	22.92
Multi-Stage filters	0.089	0	2.36	0	79.38	46.58

4 结束语

在高速网络流量监控中,要及时而准确地测量所有流量对象变得越来越困难,而大部分大规模网络安全事件的检测只需要提取大流量对象,这为问题的解决提供了新的途径.本文结合 LRU 淘汰机制和 LEAST 淘汰机制,建立了基于二级淘汰机制的大流量对象统计算法 LLR,能够准确测量大流量对象.该算法不但占用存储空间较少,而且存储空间不会随着网络数据的增加而扩大.该算法使得在 SRAM 中处理大流量对象信息成为可能,从而使系统能够更快处理网络数据、分析和实验表明,该算法可以满足 2.5Gbps(OC-48)线路的在线测量需求.

虽然 LLR 算法需要在 LEAST 列表执行对象插入和删除,使 LLR 算法的计算复杂度相对要高,但网络流量的类幂率分布特征[16,17]极大地削减了这种操作发生的次数,而且通过设定合理的 LEAST 列表长度,可以使这种操作的计算量降低.相对于 PSH 算法和 Multi-Stage Filters 算法[3],LLR 算法的准确性有较大的提高且可扩展性好,但计算复杂度也相对较高,应用中要根据实际需求在存储空间、准确性、计算复杂度之间作出权衡.

高速网络大流量对象信息的提取,对于大规模网络安全事件的检测具有重要意义,一方面加快了网络流量信息的处理速度,避免了离线处理,保证了网络安全事件报警和处理的及时性,另一方面能够把网络安全事件可能的“罪魁祸首”过滤出来,使其特征暴露更加明显,从而降低识别的难度.例如,Slammer 蠕虫爆发初期,高速网络总流量变化并不明显,在线测量所有的流量对象又难以实现,而提取以目的端口定义的大流量对象,就会使以1 434 为目的端口的 Slammer 蠕虫流量变化凸现出来,从而能够及早对该端口实施封堵.基于大流量对象测量的

© 中国科学院软件研究所 http://www.jos.org.cn

大规模网络安全事件检测是我们下一步研究的重点.

References:

[1] Keys K, Moore D, Estan C. A robust system for accurate real-time summaries of Internet traffic. SIGMETRICS Performance Evaluation Review, 2005,33(1):85−96.

[2] Schweller R, Gupta A, Parsons E, Chen Y. Reversible sketches for efficient and accurate change detection over network data streams. In: Proc. of the ACM SIGCOMM Conference on Internet Measurement. Taormina: ACM Press, 2004. 207−212.

[3] Estan C, Varghese G. New directions in traffic measurement and accounting: Focusing on elephants, ignoring the mice. ACM Trans. on Computer Systems, 2003,21(3):270−313.

[4] Cisco NetFlow. 2006. http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html

[5] Brownlee N, Mills C, Ruth G. Traffic flow measurement: Architecture. RFC 2722, 1999.

[6] Estan C, Keys K, Moore D, Varghese G. Building a better NetFlow. In: ACM SIGCOMM Computer Communication Review, 2004,34(4):245−256.

[7] Duffield N, Lund C, Thorup M. Flow sampling under hard resource constraints. In: ACM. SIGMETRICS 2004, New York: ACM Press, 2004. 85−96.

[8] Cheng G, Gong J, Ding W. Network traffic sampling measurement model on packet identification. Acta Electronica Sinica, 2002, 30(12A):1986−1990 (in Chinese with English abstract).

[9] Wang HB, Wei AM, Lin Y, Cheng SD. Time stratified packet sampling based on measurement buffer for flow measurement. Journal of Software, 2006,17(8):1775−1784 (in Chinese with English abstract).

[10] Kodialam M, Lakshman TV, Mohanty S. Runs based traffic estimator (RATE): A simple, memory efficient scheme for per-flow rate estimation. In: IEEE INFOCOM 2004. Hongkong: IEEE Press, 2004.1808−1818.

[11] Hao F, Kodialam M, Lakshman TV. ACCEL-RATE: A faster mechanism for memory efficient per-flow traffic estimation. In: ACM SIGMETRICS. New York: ACM Press, 2004. 155−166.

[12] Hao F, Kodialam M, Lakshman TV, Zhang H. Fast, memory-efficient traffic estimation by coincidence counting. In: INFOCOM 2005. Miami: IEEE Press, 2005. 2080−2090.

[13] Kumar A, Xu J, Wang J, Spatschek O, Li L. Space-Code Bloom filter for efficient per-flow traffic measurement. In: IEEE INFOCOM. Hong Kong: IEEE Press, 2004. 1762−1773.

[14] Gong J, Peng YB,Yang W, Liu WJ. Reconstructing the parameter for massive abnormal TCP connections with Bloom filter. Journal of Software, 2006,17(3):434−444 (in Chinese with English abstract).

[15] Yin Z, Sumeet S, Subhabrata S, *et al*. Online identification of hierarchical heavy hitters: Algorithms, evaluation, and applications. In: Internet Measurement Conf. (IMC 2004). Taormina: ACM Press, 2004. 101−104.

[16] Kim I. Analyzing network traces to identify long-term high rate flows [MS. Thesis]. Texas A&M Univ., 2001.

[17] Xu K, Zhang ZL, Bhattacharyya S. Profiling Internet backbone traffic: Behavior models and applications. In: ACM SIGCOMM. Philadelphia, 2005. 169−180.

[18] Duffield N, Lund C, Thorup M. Charging from sampled network usage. In: ACM SIGCOMM Internet Measurement Workshop. San Fransisco: ACM Press, 2001. 245−256.

[19] NLANR. 2006. http://pma.nlanr.net/Special/

附中文参考文献:

[8] 程光,龚俭,丁伟.基于分组标识的网络流量抽样测量模型.电子学报,2002,30(12A):1986−1990.

[9] 王洪波,韦安明,林宇,程时端.流测量中基于测量缓冲区的时间分层分组抽样.软件学报,2006,17(8):1775−1784.

[14] 龚俭,彭艳兵,杨望,刘卫江.基于 Bloom Filter 的大规模异常 TCP 连接参数再现方法.软件学报,2006,17(3):434−444.

王风宇(1973－),男,山东龙口人,博士,主要研究领域为网络行为学,网络安全.

王晓峰(1978－),男,博士生,主要研究领域为网络安全,网络模拟.

云晓春(1971－),男,博士,教授,博士生导师,主要研究领域为计算机网络,信息安全.

王勇(1976－),男,博士生,主要研究领域为对等网络测量,网络安全.

© 中国科学院软件研究所 http://www.jos.org.cn