软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2014,25(11):2616−2626 [doi: 10.13328/j.cnki.jos.004512]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种采用混合切分法的报文分类算法*

韩伟涛, 伊 鹏, 张 霞

(国家数字交换系统工程技术研究中心,河南 郑州 450002)

通讯作者: 韩伟涛, E-mail: hanweitao.cn@gmail.com

摘 要: 传统的基于几何区域分割的报文分类算法在空间切分时,通常只采用一种切分方法,并不会根据每个域的特点选取不同的对策.提出了一种采用混合切分法的报文分类算法 HIC(hybrid intelligent cuttings).首先,按照 IP 前缀长度将规则集分组;然后,在每个分组中根据当前切分域的特点,分别对 IP 域和端口域采用比特位切分法和精确投影点切分法实现空间分解;最后,构建混合切分结构的决策树.仿真结果表明,HIC 算法具有较好的规则集适应性,其时间性能与空间性能分别比代表算法 EffiCuts 提高了 46%和 74%.

关键词: 网络安全;服务质量;报文分类;决策树;空间分割

中图法分类号: TP393

中文引用格式: 韩伟涛,伊鹏,张霞.一种采用混合切分法的报文分类算法.软件学报,2014,25(11):2616−2626. http://www.jos.org.cn/1000-9825/4512.htm

英文引用格式: Han WT, Yi P, Zhang X. Hybrid cutting algorithm for packet classification. Ruan Jian Xue Bao/Journal of Software, 2014,25(11):2616−2626 (in Chinese). http://www.jos.org.cn/1000-9825/4512.htm

Hybrid Cutting Algorithm for Packet Classification

HAN Wei-Tao, YI Peng, ZHANG Xia

(National Digital Switching System Engineering and Technological R&D Center, Zhengzhou 450002, China)

Corresponding author: HAN Wei-Tao, E-mail: hanweitao.cn@gmail.com

Abstract: Traditional packet classification algorithms based on space-decomposition usually use only one heuristic to split the rule space, and they don't adopt different heuristics according to the characteristics of each dimension. This paper proposes a hybrid intelligent cutting (HIC) scheme for packet classification. HIC firstly partitions the ruleset according to the IP prefix length. Then, taking into account the characteristics of current cutting dimension in each subruleset, HIC uses bit cuttings and precise projection point cuttings to cut the IP dimension and port dimension, respectively. At last, HIC builds the decision tree of hybrid cutting structures. Simulation results show that HIC has better scalability with different rulesets. Compared with EffiCuts, its time and space performance have increased by 46% and 74% respectively.

Key words: network security; quality of service; packet classification; decision tree; space-decomposition

报文分类作为当今许多互联网关键技术的基础,在诸多领域发挥着重要的作用,例如防火墙(fire wall)、入侵检测系统(intrusion detection system)、服务质量(quality of service)等.随着网络流量的逐渐增大,工业界对报文分类处理的实时性要求越来越高,报文分类算法处理速度的快慢和对规则集适应性的强弱将直接影响到网络的性能[1].

报文分类是指根据数据包头中的特征信息,把报文划分为不同数据流的过程.报文分类从本质上讲是计算几何中的多维空间点定位问题[2].早期的基于几何区域分割的报文分类算法[3,4]对规则集的适应性较差,难以处

* 基金项目: 国家重点基础研究发展计划(973)(2012CB315901); 国家高技术研究发展计划(863)(2011AA01A103); 国家科技支撑计划(2011BAH19B01)

收稿时间: 2013-05-14; 定稿时间: 2013-10-21

理复杂的规则集合.Qi 等人提出的 HyperSplit 算法[5,6]采用了一种非等分的二叉树结构,在一定程度上减少了规则复制的出现,但其决策树深度过大,影响到算法的时间性能.Balajee 等人提出的 EffiCuts 算法[7]通过采用多种启发式方法大幅度降低了算法的内存使用,但其内存访问次数却增加了数倍之多.近年来,仍有许多学者对报文分类问题做了详细的研究[8−11],但这些算法都只采用 1 种方法实现空间的分解,并未根据每个域的特点选取不同的对策.

本文根据报文五元组信息的特点,提出了一种采用混合切分法的报文分类算法 HIC(hybrid intelligent cuttings).该算法在进行空间切分时,根据切分域的特征,在 IP 域按照比特位的 0,1 分布完成规则集的划分,在端口域,通过选取特定的规则投影点实现规则集的分解.此外,为了优化算法的空间性能,本文采用了一种基于 IP 前缀长度分布的规则分组法,以使算法能够适应较大规模的规则集.仿真结果表明,HIC 算法具有较好的规则集适应性,与代表算法 EffiCuts 相比,时间和空间性能分别提高了 46%和 74%.

1 算法思想来源

1.1 比特位切分

报文分类所要处理的报文五元组信息可以用 104 比特来表示,其中,源、目的 IP 域各占 32 比特,源、目的端口域各占 16 比特,协议域占用 8 比特.

表 1 为一个真实规则集的示例.

Table 1 Example of ruleset

表 1 规则集示例

源 IP	目的 IP	源端口	目的端口	协议
190.187.172.234/32	229.163.191.200/32	53:53	0:65535	0x06/0xFF
248.180.24.174/32	190.187.172.98/32	0:65535	69:69	0x11/0xFF
31.117.6.39/32	233.55.47.5/32	0:65535	9003:9003	0x06/0xFF
7.71.190.75/32	31.58.90.41/32	0:65535	143:143	0x11/0xFF
6.106.237.80/32	229.163.191.200/32	0:65535	25:25	0x11/0xFF

不同的报文规则具有不同的比特位信息,因此,可根据规则的这种特点实现规则集切分.图 1 为比特位切分示例,图中选取的切分位为 X 域的第 1 位和第 3 位,规则集被划分为 4 个子集,分别包含切分比特位为“00”,“01”,“10”和“11”的规则.

比特位切分具有两个优点:

第一,存储的简洁性.算法需要记录的切分信息极为简洁,便于硬件实现.以图 1 为例,通过选取 X 域的第 1 位、第 3 位作为切分位,可以将规则集划分为 4 个子集,而所需要存储的切分信息只需要 2 bits.

第二,切分的高效性.通过选取少量的比特位,就可以将规则集划分为较多个规则子集.例如,8 个比特位就可以将规则集划分为 2^8=256 个子规则集.

虽然比特位切分具有很大的优点,但如何选取最优切分位,是这种方法所面对的最大的难点.考虑到五元组信息共有 104 位,从中选取 5 位作为切分位,可能的组合就有 $C_{104}^{5}=91962520$ 种,在如此众多的组合中寻找最优解,其运算量将是惊人的.针对这个问题,文献[12]提出了一些启发式方法来降低比特位选取的复杂度,这些方法都是以牺牲选取结果的最优性为代价,换取预处理性能的提高.本文为了降低切分位选取的复杂度,将比特位选取的范围缩小到 IP 域,即,在一次比特位选取的过程中,只考虑源 IP 或者目的 IP 的 32 bits.这样,通过减少比特位的候选值,降低了算法的预处理时间.

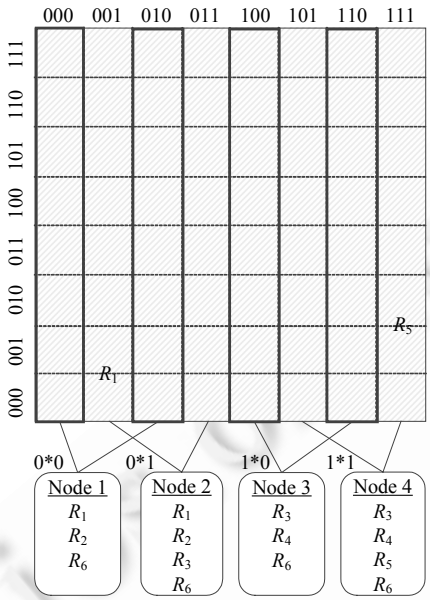


Fig.1 Example of bit cuttings

图 1 比特位切分示例

1.2 精确投影点切分

在报文规则的 104 位五元组信息中,源、目的 IP 和协议域均由前缀表示,这种形式便于切分比特位的选取.然而,规则端口域是由范围表示的,若采用比特位切分法切分,会涉及到范围到前缀的转换问题[13,14],从而增加算法处理的复杂度.因此,比特位切分并不适用于端口域.

虽然比特位切分法无法很好地完成端口域的分解,但考虑到端口域仍然具有很强的区分性[15],因此本文采用精确投影点切分法划分端口域.端口域范围表示特性使其投影端点可以分布在规则空间中的任何一个位置,而传统的等比例切分法只能在规则空间的等分点处完成规则集的划分,这种方法在端口域中往往效果较差,若将切分点选在规则投影的端点处,则切分效果会得到很大的优化,如图 2 所示.

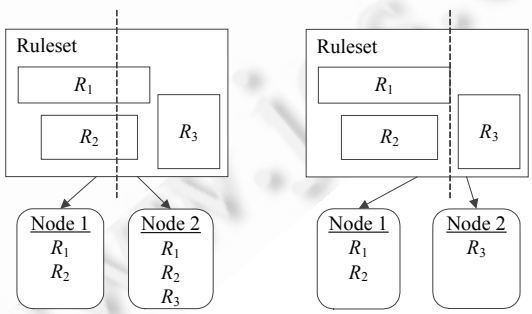


Fig.2 Aligned cuttings and precise projection points cuttings

图 2 等比例切分与精确投影点切分

虽然在 IP 域也可以采用精确投影点切分法实现空间分解,但是由表 1 中规则集的特点可知,规则 IP 域中每一个投影端点需要 32 bits 来表示,若切割次数为 N 次,那么所需要记录的切分信息为 N×32 bits,其所占用的存储空间为同样情况下端口域切分的 2 倍,这会使决策树中间节点的数据结构过于复杂,从而降低算法的时间性能.因此,本文并未采用精确投影点切分法来切分规则集的 IP 域.

2 HIC 算法

HIC 算法主要采用两种启发式方法来完成多域空间分解:1) 比特位划分规则集(IP 域);2) 精确规则投影点切分(端口域).

在决策树构建时,算法会根据当前切分域选取不同的切分方法,以适应各个域的特点.算法还采用了一种基于 IP 域前缀长度分布的规则分组法来提高算法的空间性能.HIC 切分域的选取沿用了文献[5]中的启发式方法.此外,通过对规则集特征的分析可知,五元组前 4 个域的信息已经足以完成规则集的划分[16,17].因此,在算法执行时不考虑协议域.下面对算法进行详细的介绍.

2.1 比特位的选取

为了方便问题的描述,结合 HIC 算法的具体实施方案,进行以下定义:

定义 1(有效位 C-Bit). 规则 IP 域中,能够有效地划分规则集的比特位称为有效位.

定义 2(切分向量 *V*). 承载切分信息的总长度为 32 bits 的二进制比特串称为切分向量.

定义 3(查找表 *T*). 设 C-Bit 个数为 n,那么由 n 个比特购成的比特串所有可能的取值称为查找表,其表项最多有 2^n 个.

有效位 C-Bit 主要描述切分位所在的位置;切分向量则是根据 C-Bit 的信息,将向量中所对应的位置“1”,其余位置“0”.以图 1 为例,C-Bit 为 X 域的第 1 位、第 3 位,切分向量为“101”,其查找表有 4 个表项,分别为“00”,“01”,“10”,“11”.

定义 4(比特位选取度量函数).

$$C(R,v)=\max_{i=0}^{n-1}|R_i|,\ n=2^{|v|}-1 \tag{1}$$

其中,R 为规则集,v 为切分向量,$|R_i|$为比特位切分后每个分组所包含的规则数量.

定义 5(比特向量方差函数).

$$var(v)=\sum_{i=0}^{n-1}\frac{\left(|R_i|-\overline{|R_\Sigma|}\right)^2}{n},\ n=2^{|v|}-1 \tag{2}$$

其中,$\overline{|R_\Sigma|}$ 为每个分组所包含的规则数量的平均值.

HIC 算法切分向量选取的基本思想是:在保证比特位分割后的分组容量的最大值尽可能小的情况下,使每个分组内规则数量近似相同.

根据 HIC 算法的基本思想,切分向量确定方法如下:选取具有最小度量函数的比特向量为候选值,并在所有的候选值中,选取具有最小方差函数的比特向量作为切分向量.

度量函数的最小化是为了确保决策树子节点中包含的规则尽可能地少,以降低算法处理子节点的复杂度.此外,子节点规则数量的减少也起到了一定的减少冗余规则的效果.选取最小方差函数的目的是尽量均衡每个子节点中所包含的规则数量,避免决策树结构过于“偏斜”,降低决策树的平均深度,提高算法的时间性能.与文献[12]中的比特选取方法不同,HIC 算法由于只在 IP 域中选取 C-Bit,其选取的基数较小.因此,HIC 采用遍历所有可能比特值的方法,一次性确定最优的切分向量.此外,通过引入比特向量方差函数,确保规则能够更加均衡地分布到每一个子集中.

设规则集为 R,划分好的子规则集为 P,则 HIC 算法切分向量选取算法伪码如算法 1 所示.

算法 1. 切分向量选取.

Input: R.

Output: V,P.

1. //Init
2. V_0={*Any Possible Vector*}, c'=+∞, var=+∞
3. //Bit Vector Selection

```
4.  for v∈V0 do
5.      c=C(R,v)
6.      if c′≥c then
7.          if c′=c and var>var(v′) then
8.              var=var(v′)
9.              v′←v
10.             continue
11.         end if
12.         var=var(v′)
13.         v′←v
14.         c′←c
15.     end if
16. end for
17. //Partition Ruleset
18. for r∈R do
19.     Pstr=BitJoin(r.dim & v′)
20.     for i∈Pstr do
21.         P[i].add(r)
22.     end for
23. end for
24. V=v′
25. return V,P
```

下面以一个例子来说明切分向量的选取.在图 1 中,假定在 X 域拟选取 2 个 C-Bit,可能的结果共有 3 种,分别计算其度量函数,可得 $C(R,110)=4$,$C(R,101)=4$,$C(R,011)=5$.从 3 个结果中选取具有最小度量函数的比特向量,分别计算其方差函数,可得 $var(110)=0.5$,$var(101)=0.25$,最后选取具有最小方差函数的比特向量为切分向量,即 $V=101$(对应于算法 1 的第 1 行~第 16 行).然后,根据切分向量 V 划分规则集(对应于算法的第 17 行~第 24 行).

2.2 精确投影点的选取

精确投影点切分的基本思想是:通过选取适当的 n 个规则投影点划分规则集,将规则集划分为 $n+1$ 个包含相近规则数的规则子集.下面具体介绍精确投影点切分法.

定义规则投影点数组为 $RP[i]$,其中,$0\leqslant i<N$,N 为投影点总个数,数组中的元素是按照从小到大的顺序排列的规则投影点.定义每两个投影点构成的区间所包含的规则个数为 $nSg[i]$,其中,$0\leqslant i<N-1$.根据以上两个定义,给出区间规则加权函数的定义.

定义 6(区间规则加权函数).

$$L(s,e)=\sum_{i=s}^{e}nSg[i],s\leqslant e \tag{3}$$

其中,s 为起始区间,e 为结束区间.

根据精确投影点切分的基本思想,HIC 算法采用了一种渐进式的方法来完成切分点的选取,即,按照从小到大的顺序遍历每一个规则投影点,求出其区间规则加权函数,并与预先设定好的子规则集所包含的规则数目相比较,当某一点的加权函数大于或者等于预先设定好的规则数目时,此点就被选为切分点.若所有的切分点选取完毕,则算法按照选好的投影点划分规则集,并建立相应的决策树.

由于保存端口域切分点信息需要 16 bits,过多的切分点会增加算法的空间复杂度.此外,考虑到投影点切分的非等分性,查找算法在最坏情况下需要对每一个投影点进行比较才能确定报文所对应的子节点,这会降低算

法的时间性能,因此,切分投影点的数量不宜过多.

设拟选取的切分点个数的最大值 M,切分点选取算法伪码如算法 2 所示.

算法 2. 精确投影点选取.

Input: RP, $nSeg$, M.

Output: $p_1,p_2,\ldots,p_M$, $NumCut$.

```
1.  //Init
2.  i=0, j=0, s=0, sum=L(0,N–2)
3.  while j<N–1 do
4.      if L(s,j)≥sum/(M+1) then
5.          p_i=RP[j+1]
6.          s=j+1
7.          i=i+1
8.          if i>M then
9.              break
10.         end if
11.     end if
12.     j=j+1
13. end while
14. NumCut=i
15. return p_1,p_2,…,p_M, NumCut
```

2.3 规则分组

为了降低算法的内存占用,HIC 采用了一种基于 IP 前缀长度分布的规则分组法.与传统的基于通配符(wildcards)分组法不同,HIC 的分组方法是根据 IP 域的前缀来界定大规则与小规则.规则分组法介绍如下:

设分组前缀界定长度为 l,那么根据 l 值的大小,把规则集划分为 3 部分:

1) 源 IP 域前缀长度小于 l 并且目的 IP 域前缀长度大于 l 的规则;
2) 目的 IP 域前缀长度小于 l 并且源 IP 域前缀长度大于 l 的规则;
3) 源、目的 IP 域前缀长度均小于 l 规则.

规则分组的目的是把那些占用较大规则空间范围的规则与小规则分离开,避免算法切分时,由于小规则与大规则之间交叉而产生不必要的冗余[7].由于每个规则分组之间并不具有完备性,因此,查找时需要遍历每一棵子树,并通过汇总每一棵树的查找结果得出最终的匹配决策.这会增大算法的内存访问次数,导致时间性能的下降.

2.4 数据结构

HIC 的中间节点的数据存储结构见表 2.其中,2 bits 用来记录中间节点的切分域信息.29 bits 存储了指向子节点的偏移地址信息.切分信息总共占用 32 bits,其内容与当前选取的切分域相关,若切分域为 IP 域,则存储切分向量;若切分域为端口域,则存储相应的切分端点.规则上提指示位占用 1 bit,上提规则指针占用 N×16 bits.

Table 2 Data structure of middle nodes

表 2 中间节点数据结构

存储内容	所占空间(bit)
多分域信息	2
偏移地址信息	29
切分信息	32
规则上提指示位	1
上提规则指针	N×16

HIC 的子节点存储结构见表 3,根据子节点所包含的规则个数,共占用 N×16 bits.

Table 3 Data structure of child nodes

表 3 子节点数据结构

存储内容	所占空间(bit)
规则指针	N×16

此外,在决策树遍历查找结束后,需要在子节点内进行规则的线性查找,因此,需要保存全部规则的五元组信息,这部分内容在内存中按照优先级从高到低的顺序连续存储,以方便查找时对内存空间的寻址.已知标准的五元组信息总共 104 bits,设规则数为 n,那么这部分存储空间的大小为 n×104 bits.

2.5 查找算法

HIC 的决策树通过两种启发式方法构建,因此,算法在查找时,对应不同的域,采用不同的查找策略.

若节点的切分域为 IP 域,那么当报文到达时,提取其 IP 域信息,采用文献[12]的查找方法完成子节点的匹配.HIC 的切分向量只有 32 bits,其所需的位宽较小.低位宽的数据会降低硬件布局布线的复杂度,提高运行的时钟频率.因此,HIC 算法更易于在硬件中实现.根据图 1 中的规则集,比特查找过程如图 3 所示.

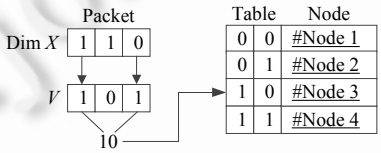


Fig.3 Bit lookup

图 3 比特查找

若节点的切分域为端口域,那么算法直接将报文的包头信息与切分点逐一比较,确定报文所属的子节点.

决策树的查找复杂度为 $O(D)$,其中,D 为决策树的最坏深度.在子节点中,采用线性查找确定匹配的规则,复杂度为 $O(M)$,其中,M 为子节点规则数.因此,HIC 的查找复杂度为 $O(D+M)$.

3 仿真与性能分析

本文所采用的规则集均由 ClassBench[15]产生.ClassBench 是由华盛顿大学路易斯分校的 Taylor 所开发,其产生的数据集在报文分类研究中广泛使用.它是目前较为权威的规则库的产生和测试平台,主要包含 ACL (access control list),FW(fire wall)和 IPC(linux IP chains)这 3 类规则.

仿真采用 C 语言编程,使用 GCC-4.4.3 编译,操作系统为 Ubuntu 10.04 LTS.此外,为了进一步提高 HIC 的算法性能,HIC 引入了规则覆盖删减[3]和规则上提[4].

参与比较的算法参数设置如下:

1) 3 种算法子节点包含规则的最大值均为 8,内存访问位宽设定为 8 bytes.
2) EffiCuts 的规则上提阈值设为 1,其他参数采用算法默认值[7].
3) 在 IP 域中,HIC 选取的有效位为 2 位,在端口域中,最大的切分点个数为 2;规则分组中前缀界定长度为 8;规则上提的阈值为 1.

考虑到过多的有效位会增加算法的预处理复杂度,因此选取有效位为 2 位.此外,为了使端口域切分信息与切分向量的长度相匹配,设定端口域切分点的最大值为 2.

3.1 规则集适应性对比

本节通过对比算法的空间与时间性能,从侧面来反映算法对规则集的适应性.对比算法均为单决策树算法.单决策树是在没有采用规则分组的情况下,算法所建立的决策树.由于算法在对规则集处理前并未对规则集进

行分组优化,因此,单决策树能够更好地体现算法本身对规则集的适应性.单决策树 HIC 算法(single tree HIC,简称 HIC-ST)没有采用第 2.3 节的规则分组法.对比的对象是单决策树代表算法 HyperSplit[5].

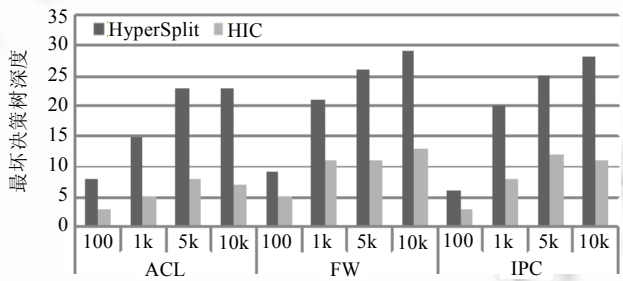
表 4 比较了 HyperSplit 算法与 HIC-ST 算法的空间性能.从表 4 中可以看出,与 HyperSplit 相比,HIC-ST 内存的使用平均减少了 58%.

Table 4 Memory comparison of HyperSplit and HIC-ST

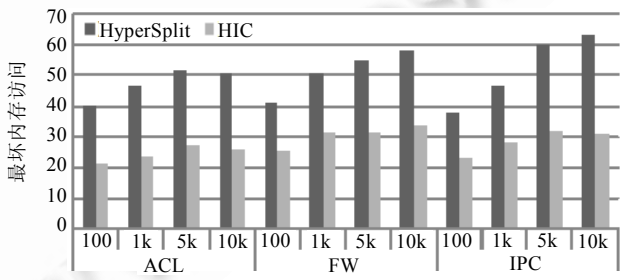
表 4 HyperSplit 与 HIC-ST 内存使用对比

规则集		HyperSplit (KB)	HIC-ST (KB)	减少(%)
ACL	100	2	1	50
	1k	18	14	22
	5k	179	68	62
	10k	219	150	32
FW	100	2	2	0
	1k	312	43	86
	5k	15 256	4 268	72
	10k	76 581	31 657	59
IPC	100	2	1	50
	1k	67	21	69
	5k	5 782	112	98
	10k	19 052	240	99

基于决策树的报文分类算法的时间性能与其决策树深度和内存访问次数直接相关,因此,在图 4 中分别比较了 HyperSplit 和 HIC-ST 在最坏情况下决策树深度和内存访问次数.从图 4 中可以看出,与 HyperSplit 算法相比,HIC-ST 算法的最坏决策树深度平均减少了 58%,内存访问次数平均减少了 45%.



(a) 最坏决策树深度



(b) 内存访问次数

Fig.4 Time performance comparison of HyperSplit and HIC-ST

图 4 HyperSplit 与 HIC-ST 时间性能对比

由表 4 和图 4 的对比结果可以看出,HIC 算法对 3 类规则集的适应性与 HyperSplit 算法相比有所提高.对于 ACL 规则集、IPC 规则集,HIC 算法具有较好的适应性;对于 FW 规则集,虽然 HIC 取得了一定的优化,但其内存使用仍然较多.HIC 算法对 FW 规则集适应性较低的主要原因是:FW 规则存在着大量的通配符,导致规则交叉情

况较为严重,使得算法在进行域切分时引入大量的规则复制,降低了算法的性能.

3.2 算法性能对比

本节比较了 HIC 算法与代表算法 EffiCuts[7]的性能.其中,HIC 算法采用了第 2.3 节中的规则分组法.

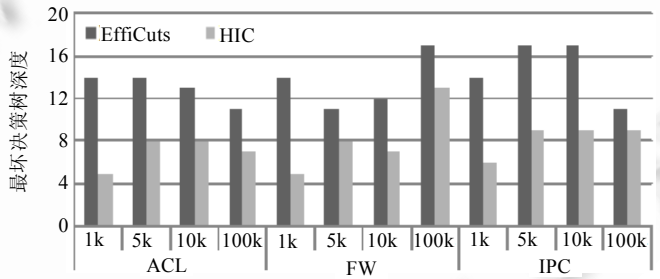
表 5 比较了 EffiCuts 算法与 HIC 算法的空间性能,其中,HIC 在处理 100 k 规则集时,规则指针位宽增加到 17 bits.可以看出,与 EffiCuts 算法相比,HIC 算法的内存使用平均减少了 74%.

Table 5 Memory comparison of EffiCuts and HIC

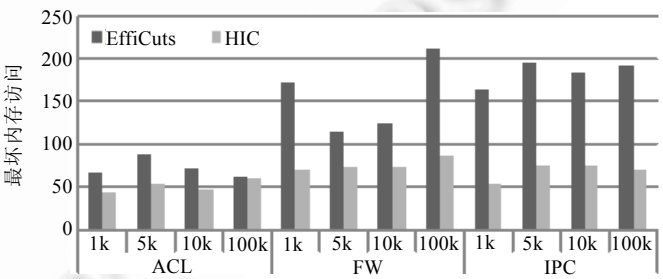
表 5 EffiCuts 与 HIC 内存对比

规则集		EffiCuts (KB)	HIC (KB)	减少(%)
ACL	1k	70	13	81
	5k	410	65	84
	10k	750	146	80
	100k	4 990	1 509	70
FW	1k	60	13	78
	5k	280	154	45
	10k	620	170	73
	100k	7 900	2 475	69
IPC	1k	50	13	74
	5k	370	68	82
	10k	640	139	78
	100k	6 900	2 129	69

图 5 比较了 EffiCuts 与 HIC 的时间性能.可以看出,与 EffiCuts 相比,HIC 的最坏决策树深度和内存访问次数平均分别减少了 42%和 46%.



(a) 最坏决策树深度



(b) 内存访问次数

Fig.5 Time performance comparison of EffiCuts and HIC

图 5 EffiCuts 与 HIC 时间性能对比

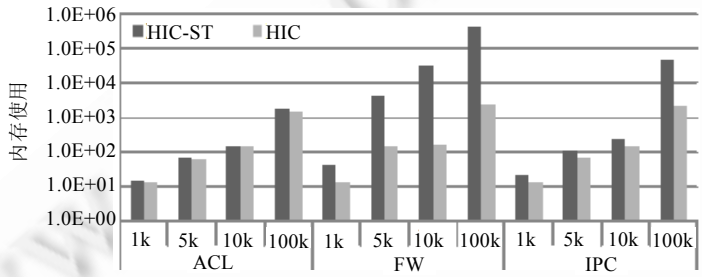
3.3 性能分析

(1) 从第 3.1 节中的比较可以看出:HIC-ST 算法对规则集的适应性优于 HyperSplit 算法,其主要原因是 HIC

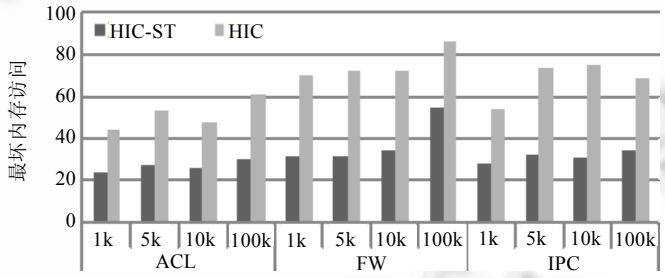
算法采用了比特位切分法.与 HyperSplit 算法在每个域只切分 1 次不同,HIC 使用 n 个比特位将规则集一次性地划分为 2^n 个子集,提高了切分效率,从而减少了决策树深度和内存使用,增加了算法对规则集的适应性.

(2) EffiCuts 算法虽然采用了多种启发式方法,使内存占用得到了大幅度优化,但其使用的启发式方法复杂度较高,导致 EffiCuts 的数据结构较为复杂,不利于硬件实现.HIC 算法所采用的启发式方法数据结构较为精简,复杂度低于 EffiCuts,内存使用较少,且易于在硬件上实现.从第 3.2 节的比较中也可以看出:具有较为精简数据结构的 HIC 空间性能优于 EffiCuts 算法,在采用同样内存访问位宽的情况下,HIC 算法的内存访问次数低于 EffiCuts.

(3) 图 6 比较了 HIC-ST 算法与 HIC 算法的性能.从图 6 中可以看出:采用规则分组法后,算法的内存使用平均降低了 51%,但内存访问次数却增加了 2 倍多.虽然规则分组可以使算法的内存使用大幅度降低,但由于查找需要遍历每一棵决策树,其内存访问次数会有所增加.因此在实际的应用中,可以根据网络设备对算法空间和时间性能的需求,选择是否采用规则分组.



(a) 内存使用对比(KB)



(b) 内存访问次数对比

Fig.6 Performance comparison of HIC-ST and HIC

图 6 HIC-ST 与 HIC 性能对比

4 结束语

本文在分析报文规则特点的基础上提出了一种采用混合切分法的报文分类算法.该算法根据切分域的特征,分别在 IP 域和端口域采用比特位切分和精确投影点切分来划分规则集.此外,为了使算法能够适应大规模规则集(100k 以上),提出了一种基于 IP 域前缀长度分布的规则分组法,以降低规则集的预处理复杂度.仿真结果表明,HIC 算法具有较好的规则集适应性,其时间性能与空间性能分别比代表算法 EffiCuts 提高了 46%和 74%.

References:

[1] Taylor DE. Survey and taxonomy of packet classification techniques. ACM Computing Surveys, 2005,37(3):238−275. [doi: 10.1145/1108956.1108958]

[2] de Berg M, van Krefeld M, Overmars M, Cheong O. Computational Geometry: Algorithms and Applications. 3rd ed., Springer-Verlag, 2008. [doi: 10.1007/978-3-540-77974-2]

[3] Gupta P, McKeown N. Classifying packets with hierarchical intelligent cuttings. IEEE Micro, 2000,20(1):34–41. [doi: 10.1109/40.820051]

[4] Singh S, Baboescu F, Varghese G, Wang J. Packet classification using multidimensional cutting. In: Proc. of the ACM SIGCOMM. ACM Press, 2003. 213–224. [doi: 10.1145/863955.863980]

[5] Qi YX, Xu LH, Yang BH, Xue YB, Li J. Packet classification algorithms: From theory to practice. In: Proc. of the IEEE INFOCOM. IEEE Press, 2009. 648–656. [doi: 10.1109/INFCOM.2009.5061972]

[6] Qi YX, Fong J, Jiang WR, Xu B, Li J, Prasanna V. Multi-Dimensional packet classification on FPGA: 100 Gbps and beyond. In: Proc. of the 20th Int'l Conf. on Field-Programmable Technology. IEEE Press, 2010. 241–248. [doi: 10.1109/FPT.2010.56 81492]

[7] Vamanan B, Voskuilen G, Vijaykumar TN. Efficuts: Optimizing packet classification for memory and throughput. In: Proc. of the ACM SIGCOMM. ACM Press, 2010. 207–218. [doi: 10.1145/1851182.1851208]

[8] Ma YD, Banerjee S. A smart pre-classifier to reduce power consumption of TCAMs for multi-dimensional packet classification. In: Proc. of the ACM SIGCOMM. ACM Press, 2012. 335–346. [doi: 10.1145/2342356.2342428]

[9] Chang YK, Wang YH. CubeCuts: A novel cutting scheme for packet classification. In: Proc. of the 26th Int'l Conf. on Advanced Information Networking and Applications Workshops. IEEE Press, 2012. 274–279. [doi: 10.1109/WAINA. 2012.110]

[10] Song HY, Turner JS. ABC: Adaptive binary cuttings for multidimensional packet classification. IEEE/ACM Trans. on Networking, 2013,21(1):98–109. [doi: 10.1109/TNET.2012.2190519]

[11] Chang YK, Chien CY. Layer partitioned search tree for packet classification. In: Proc. of the 26th Int'l Conf. on Advanced Information Networking and Applications. IEEE Press, 2012. 276–282. [doi: 10.1109/AINA.2012.122]

[12] Yang BH. Key technologies in network traffic management and optimization [Ph.D. Thesis]. Beijing: Tsinghua University, 2012 (in Chinese).

[13] Meiners CR, Liu AX, Torng E. Bit weaving: A non-prefix approach to compressing packet classifiers in TCAMs. IEEE/ACM Trans. on Networking, 2012,20(2):488–500. [doi: 10.1109/TNET.2011.2165323]

[14] Vamanan B, Vijaykumar TN. TreeCAM: Decoupling updates and lookups in packet classification. In: Proc. of the 7th Conf. on Emerging Networking Experiments and Technologies. ACM Press, 2011. Article No. 27. [doi: 10.1145/2079296.2079323]

[15] Taylor DE, Turner JS. ClassBench: A packet classification benchmark. IEEE/ACM Trans. on Networking, 2007,15(3):499–511. [doi: 10.1109/TNET.2007.893156]

[16] Jiang WR, Prasanna VK. Scalable packet classification on FPGA. IEEE Trans. on Very Large Scale Integration Systems, 2012, 20(9):1668–1680. [doi: 10.1109/TVLSI.2011.2162112]

[17] Jiang WR, Prasanna VK. Large-Scale wire-speed packet classification on FPGAs. In: Proc. of the 17th ACM/SIGDA Int'l Symp. on Field Programmable Gate Arrays. ACM Press, 2009. 219–228. [doi: 10.1145/1508128.1508162]

附中文参考文献:

[12] 杨宝华.网络流量管理与优化关键技术[博士学位论文].北京:清华大学,2012.

韩伟涛(1989－),男,河北无极人,硕士,主要研究领域为宽带信息网络,报文分类技术.
E-mail: hanweitao.cn@gmail.com

伊鹏(1977－),男,博士,副教授,主要研究领域为高速路由器关键技术,网络流量均衡技术.
E-mail: 15238363586@139.com

张霞(1980－),女,工程师,主要研究领域为宽带信息网络.
E-mail: zhangxia_2005@163.com