An Accurate and Efficient 3-Phase Measurement Method for IP Traffic Flow on High Speed Link

Gaogang Xie, Jianhua Yang, Valérie Issarny, Alberto Conte

An Accurate and Efficient 3-Phase Measurement Method for IP Traffic Flow on High Speed Link†

Gaogang XIE[1,2,3], Jianhua YANG[1], Valerie ISSARNY[2], Alberto CONTE[3]
1Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100080, China
2 INRIA-Rocquencourt, Domaine de Voluceau, 78153 Le Chesnay, France
3 Alcatel Research & Innovation Center, Marcoussis, France
gaogang.xie, Valerie.issarny@inria.fr

Abstract

Flow metrics are critical for protocol research, anomaly detection, network operation and application deployment. There are great challenges to match every packet into millions of flows in high speed network link. A 3-phase measurement method for IP traffic flow is proposed in this paper. The packets are captured and classified into the flows through hash table, search tree and linear table with this approach. The traffic measurement systems in general purpose CPU platform and network processor platform with this method have been implemented and deployed in backbone links. The experiment on OC-48 backbone shows the average length search path for each packet is 7.7. Average process times of mapping successfully and unsuccessfully are about 1.3 μs and 1.8 μs respectively at the length of search path 350. Traffic matrix, host behavior and other metrics can been easily deduced with our approach.

1. Introduction

New applications, for example P2P-based VoIP, have not only changed characteristic of traffic and behavior of network, but also proposed more requirements on QoS provisioning ability of network infrastructure. A deep understanding of Internet traffic is significant to protocol research [1, 2], anomaly detection [3], network operation and application deployment [4, 5]. An alarm for a traffic burst can be arose from the traffic rate measurement, but we cannot distinguish whether it is normal or abnormal and infer the reason of the traffic burst, for example which application and which hosts arising the burst. There is an extensive body of prior work on traffic characterization on IP backbones- especially in terms of statistical properties (e.g. heavy-tail, self-similarity) for the purpose of network performance engineering [2]. More sophisticated traffic metrics at flow level can provide much valuable information to deal with these issues, such as flow size distribution, per-flow traffic volume, packets size, the number of concurrent/new added/closed flows, elephant and mouse flows.

A flow is a packets sequence aggregated based on the well-known five-tuple: the source IP address (srcIP), destination IP address (dstIP), source port (srcPrt), destination port (dstPrt), and protocol(ToP) [1]. Flows measurement in a link also identify the malicious flows causing congestion from normal flows, and provide some instructions to the security system to block these malicious flows. The flow metric must be measured in real time instead of only from packet trace to achieve these purposes. There exists a great body of literature about flow measurement and analysis including of architecture of traffic flow measurement [7], format of flow export [8], algorithm of packet mapping [9,10,11], algorithm sampling [12,13,14], deployment policy [16] and characteristic of flows [1,17]. Although there has been significant progress in traffic flow measurement method and model, there is great challenge to measurement traffic flow on line for high speed link at the core of the Internet. For example, on a link of OC-768, it is difficult to map each packet which arrives every 25ns into a flow from millions flows even with the most powerful hardware of the status of art. Sampling algorithm is an acceptable option for traffic flow measurement on high speed link in some cases, especially for elephant flows measurement [12, 13, 14]. The great mass of mouse

† This research is funded by National Natural Science Foundation of China with grant no. 60403031 and 90604015, Chinese National High Technology Plan ("863" Plan) with grant no. 2005AA121560 and the Foundation of French-Chinese for Sciences and their Applications (FFCSA 2005-2006). This paper is also supported by France Telecom R&D.

Proceedings of the Sixth International Conference on Networking (ICN'07)
0-7695-2805-8/07 $20.00 © 2007 IEEE

flows are missed in sampling algorithm since there are a few packets in the mouse flow and no packet is sampled with high probability. Actually, it is very significative to detect and analyze mouse flows especially for anomalous behavior detecting. More sophisticated and comprehensive methodologies of traffic flow measurement should be proposed to meet the requirement of growing of Internet link speed and complexity of applications.

In this paper, A 3-phase traffic flow measurement methodology is proposed for these purposes. And the methodology has been implemented in NetPro100, NetPro200 and NetPro3000 which are probes of the distributed network traffic and performance measurement system developed by ourselves. Experiments with test instrument and on backbone link of operational network have proved the efficiency of the methodology.

The remainder of this paper is organized as follows. Section 2 discusses the design of hash function based on experiment of IP address distribution on the backbone link. In Section 3, we present the 3-phase measurement methodology for traffic flow. The implementation and methodology performance are introduced in Section 4. Related work is briefly discussed and compared with our approach in Section 5. Section 6 concludes the paper.

2. Hash Function

As mentioned in Section 1, in a period all packets with the same 5-tuple {*srcIP, dstIP, srcPrt, dstPrt, ToP*} belong to the same flow. Given the list of concurrent flows on a link under measurement $F = \{f_1, f_2, ..., f_l\}$, where l is the number of concurrent flows, traffic flow measurement methodology compares each transmitted packet by the 5-tuple with the flows in the list *F*. So, the flows list *F* is the set of rules of packet classification. If the 5-tuple of arriving packet equals to that of f_i , the record of f_i is updated according the packet. Otherwise, a new flow record f_j is inserted into *F* and other following packets belonging to f_j will update the record. So, unlike the traditional packets identification with the determinate rule of classification, the rule of classification is dynamic in traffic flow measurement, where a classification rule is added into the rule table when a new session is generated and deleted from the table when a session is closed. The heavy and dynamic classification rules which are modified with high frequency while the continual packet arrival bring up great challenges to the traffic flow measurement. It is impossible even with the most powerful hardware to map every packet into the list of millions flows using the methodology with the time complexity *O(l)* in line speed where *l* is the number of concurrent flows, for up to OC-192/OC-768 backbone link. Hash function is an established technique to make these network measurements feasible [18].

The digest size, probability of conflict and computing time complexity are three key factors for hash functions. The digest size of hash function ties up close with the probability of conflict. There is less possible for conflict with larger digest size which needs more memory to store them, e.g. a digest size of 32 bits requiring 4Gb memory. It is difficult to have so large memory in measurement infrastructure to store such size of digest besides millions of flow records especially in network processor platforms where the digest usually is stored in SRAM to speed up the packet mapping. A tradeoff between these factors should be thought over.

Supposed there are *n* possible hash values $H = \{h_1, h_2, ..., h_n\}$ for the hash function and *m* samples during the period. To validate the randomicity of the hash function, we define the scatter coefficient of hash function as Formula (1), where n_i is the number of samples with the hash value v_i in the period.

$$c = \frac{\sum_{i=1}^{n} \left| n_i - \frac{m}{n} \right|}{m} \qquad (1)$$

The samples may be IP pairs, flows or packets. Accordingly, the scatter coefficients of hash function can be defined as c_i, c_f and c_p. From the definition of c, it is easily to know that the hash function is more random with less *c* and $0 \le c \le 2$.

We define *Hash_index[i]* as formula (2) to verify the uniformity of byte *i* of source and destination IP address.

$$Hash\_index[i] = (srcIP\_i) << 8 + (dstIP\_i) \qquad (2)$$

The scatter coefficient of hash function based on the first and second byte of IP address is near to 2. The experiment can't be performed on line since it is impossible to map packets into flows links in real time with high scatter coefficient. The average scatter coefficients of c_p, c_i, c_f based on the last byte of IP address are 1.246, 0.532 and 0.657, correspondingly 1.511, 1.229 and1.264 based on the third byte. Obviously, the last byte of IP address has the least scatter coefficient and the hash function based the last byte of IP address has the best randomicity. So, we develop our approach based on the last byte for the hash function.

3. 3-phase Traffic Flow Measurement Methodology

A 3-phase traffic flow measurement methodology is proposed in this section. The packets transmitted on the link are captured by the traffic measurement device. Through hash table, search tree and linear link table, the packets are mapped into theirs flows finally.

3.1 Packet Mapping

As analyzed in Section 2, the last byte of the IP address of packet has the most randomcity. A hash function is constructed with the source and destination IP addresses of the captured packet as formula (3), where *srcIP_4* and *dstIP_4* denote the last byte of source IP address and destination IP address respectively.

$$Hash\_index = (srcIP\_4) << 8 + (dstIP\_4) \quad (3)$$

A hash digest table is built with the value of hash index [0..65535]. Once a packet arrives, its hash index is calculated according to formula (3). Different IP pairs with same *srcIP_4* and *dstIP_4* have same hash index. The tri-tree is used to resolve the hash conflict. Unlike the hash table being built statically, the tri-tree is constructed dynamically, and every hash index has a tri-tree. The tri-tree is an ordered tree with *T_key* to identify the order of nodes with the same hash index in the tree. *T_key* is defined as formula (4) with the third byte of *srcIP* and *dstIP*.

$$T\_key = (srcIP\_3 \& 0x0f) << 12 + (dstIP\_3 \& 0x0f) << 8 + (srcIP\_3 \& 0xf0) + (dstIP\_3 \& 0xf0) >> 4 \quad (4)$$

All flows with the same hash index value and *T_key* are organized as a linear link table.

In the initial status when the program start-up, the hash table is allotted in memory, but the search tri-tree and flow linear table are null. The mapping procedure for every packet is demonstrated in Figure 1.

The traffic flow measurement methodology is described as Figure 2.

3.2 Metrics Export

All concurrent flows are stored in the flow item of linear link table. A timer is set up to determine the flow session is closed or active. Every a defined period, says 5 seconds, the time of last packet arriving in the flow record of every flow is compared with current time. If the time difference is larger than 60 seconds, the flow is regarded ended. The ended flows are moved from the linear link to an ended flow list for advanced analysis, e.g. the distribution of workload of flows, lasting time of session, application types of flows. If the linear link is empty for a tree node, the tree node is deleted. Flow Metrics of concurrent flows can be deduced by traversing the linear link table periodically and stored into database for long-term analysis.

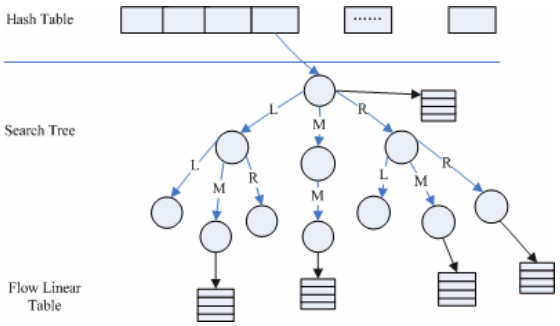


Figure 1. The procedure of packet mapping

```
1: pkt = capturePacket();
2: if (pkt = = Packet_IPv4) {
3:      pkt_header(srcIP, dstIP, srcPort, dstPort, ToP, pkt)
4:      hash_index = Hash (srcIP, dstIP);
5:      t_key = T_key (srcIP, dstIP);
6:      if ((tree_node=SearchTree (tri_tree[hash_index], t_key) ) =
        NULL)
7:           tree_node=InsertTreeNode(pkt, tri_tree[hash_index]);
8:      if (flow=SearchFlow (tree_node.linear_link, pkt)) =NULL)
9:           flow=InsertNewFlow(tree_node.linear_link, pkt);
10:     updateFlow(flow, pkt);
11: }
```

Figure 2. The traffic flow measurement algorithm

4. Implementation and Evaluation

The methodology is implemented in our traffic measurement probes NetPro 100 (Intel Xeon 2.8GHz CPU, 1GB memory, GE measurement interface), NetPro 200 (Intel Xeon 3.0GHz CPU*2, 4GB memory, OC-48 POS measurement interface) and NetPro 3000 (Motorola C-5 network processor and Broadcom 1250 CPU 800MHz, 1 GB memory, OC-12 ATM measurement interface; Broadcom 1250 CPU 800MHz, 1GB memory, GE measurement interface). Methodology performance is validated both in lab with SmartBit 2000 of Spirent which is traffic generation instrument and in the carrier's backbone link, say link *A*. In this section, we focus on verification of the efficiency of our methodology, uniformity of hash value distribution and simplicity of implementation. Experiments on the carrier's backbone, the packet trace of Abilene-I backbone [19] and traffic generation instrument are performed.

4.1 Experiments on Backbone Links

The implementation is tested with link *A* on line and the packet trace of Abilene backbone off line. The link A and the Abilene backbone are both OC-48 POS links. The packet trace of Abilene is about 2.9GB size and lasts 10 minute on August 14, 2002 as shown in Table 2. Also the packet trace of Link *A* is as Table 1.

The scatter coefficient based on the different bytes of IP address for Abilene and link *A* is shown as Table 3, and the average number of flows with every no null hash node as Figure 3. There are about 4.7 tree nodes and 6.7 items in the linear link table with each no-null hash node for link A. So, the average comparing operation is less than 6.7 during the packet mapping for every packet. As to the link of Abilene, there are less tree nodes and items for every no-null hash node. The result also shows the rationality of the last byte of IP address used. The hash based on the last byte in Abilene has the best scatter coefficient as the same as in link A. But, the coefficients are all larger than in link A since more concurrent IP pairs in link A decreases the coefficients.

The major time in our approach is spent on searching along the tri-tree and flow linear link. We define the length of search path H to denote the number of mapping in the tri-tree and link as formula (4), where HT denotes the height of tree and HL denotes the length of link that a pack traverses during the mapping.

$$H = H_T + H_L \qquad (4)$$

Table 1. Traffic of Link A

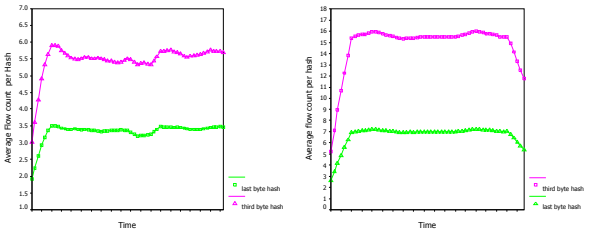
	Minimum	Maximum	Mean
Traffic (pps)	164964	743410	468390
Traffic (Mbps)	669.72	2751.68	1790.30
Concurrent IP Pairs	90486	401223	251123
Concurrent Flows	164107	740991	457540

Table 2. Traffic of Abilene

	Minimum	Maximum	Mean
Traffic (pps)	74332	90449	79596.85
Traffic (Mbps)	466.835	676.106	541.789
Concurrent IP Pairs	22012	79296	74438
Concurrent Flows	29802	122387	113161

Table 3. Scatter Coefficient of Ablience and Link A

	C_p	C_i	C_f
Last byte hash (Abliene)	1.8087	1.060	1.1883
Third byte hash (Abliene)	1.8523	1.4246	1.4916
Last byte hash (LinkA)	1.5183	0.7938	0.8833
Third byte hash (LinkA)	1.7521	1.6855	1.6874

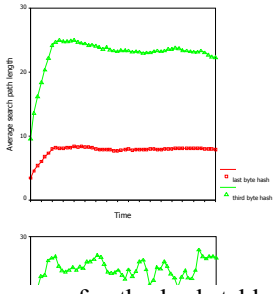


(a) The Flow Count(Abilene) (b) The Flow Count(LinkA)

Figure 3. Flows per Hash Node

Figure 4 shows the average length of search path in link A and Abilene. In the initialization period of system startup, most flows are not constructed in the data structure and H is low. H increases when more flow records are inserted. After several minutes, all concurrent flows have been inserted into the structure and the number of concurrent flows keeps correspondingly stable. The average length of search path in link A is 7.7, while it is 7.6 in Abilene. That is to say, every packet takes about 7.7 times comparing operation to find its flow in 2.5G link. Additionally, the value of H in different link keeps steady correspondingly.

The utilization ratio of NetPro200 CPU during the peak traffic is about 7.6%. The memory for storing concurrent flows including 3 parts: the hash table, the tri-tree and the flow records in the linear link table. The number of digest item in hash table is 65535, and each hash node is a pointer to a the tri-tree node. The memory for the hash table is 65536*4 =256KB. The node of tri-tree also stores the pointer to the flow record link. The memory for the tri-tree and the flow linear link table is correlative with the number of concurrent. Additional, the memory for flow linear link table depends on the properties defined in the flow record. Given 40 bytes for every flow record and 1M concurrent flows in the link, the memory for the flow linear link table is 40MB. The requirement of this scale of memory can be met in many hardware platforms both in general purpose CUP platforms and special network processor platforms.

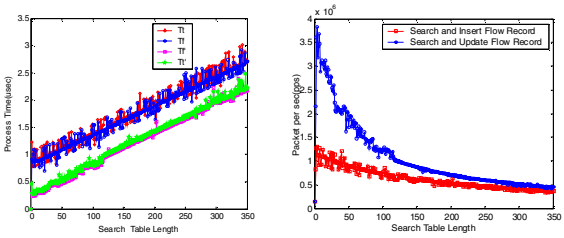


4.2 Experiments with Instrument

In order to verify the efficiency and capability of our approach, the implementations are tested with traffic generating instrument(SmartBits2000) which can generate expected packets to control the length of search path H through changing the height of the search tri-tree and length of the linear link. Implementations in different platforms have the similar experiment results, so only experiment result on NetPro100 is introduced.

SmartBits2000 continually generates packet groups at 1 Gbps traffic rate with different IP pairs into the implementations and each group is consisted of 350 packets. The length of search path H is 350 and all packets can find and update their affiliated flows after the first packet groups. Notice, according to the methodology the average flow number is about 65536*350= 22937600 which is greater than the flow number in common backbone link. Given T^t and T^f denote the total time of packet parsing, which includes packet capture, protocol analysis and flow mapping, and the time of packet mapping respectively under the condition of mapping unsuccessfully, and Tt' and $T^{f'}$ under the condition of mapping successfully. The processing time is showed as Figure 5(a) according to different length of flow links.

From Figure 5, we can know the time of flow mapping holds most the total time of packet parsing. It spends about 0.5 μs to add a new flow record and additional 0.005 μs when the length of flow link adds one. Average process times of mapping successfully and unsuccessfully are about 1.3 μs and



(a) Mapping time (b) Processing Capacity

Figure 5. Mapping time and processing capacity according to H

1.8 μs respectively at the length of search path 350. The capability of packet process of the implementation can be deduced as Figure 5(b). The average H is about 7.7 in OC-48 backbone from Section 4.2. So, even NetPro 100 platform can handle about 3.5Mpps packet in theory without regard to bandwidth of PCI.

5. Related Work

Traffic flow information is critical for network research and operation. There are two working groups in IETF IPFIX (IP Flow Information Export) and RTFM (Realtime Traffic Flow Measurement) are concerned with this field, and many network equipment vendors offer systems for example Cisco's Netflow [15]. Although there has been significant progress in this area, developing an efficient and exact methodology of traffic flow measurement for high speed backbone link remains a challengeable task.

Packet sampling attracts great effort and is increasingly employed for traffic flow measurement in order to reduce the consumption of resources on high speed backbone. The sampling policy of 1 out of n is used in Cisco's Netflow [15]. In order to improve the accuracy of sampling and sample as many packets as the system can perform, we present a best effort adaptive sampling method for flow-based traffic monitoring which the sampling probability is adapted according to the traffic rate [13]. More sampling algorithms are proposed for elephant flows measurement since elephant flows contribute about 80% traffic, which is very important to traffic engineering and accounting [12, 14]. The major measurement is it is less possibility to record small flows limitation of sampling method for high speed link flow which have less than 10 packets. So, sampling is inaccurate or only account for the elephant flows. But, these small flows take up to 80% of all flows and can provide critical information for anomaly detection, e.g. DoS/DDoS detection. Therefore, per-flow traffic being measured is critical regardless of its size, small or large.

To address the imprecision of sampling, Aghishek Kumar et al employed an efficient per-flow traffic measurement method with a data structure called Space Code Bloom Filter (SCBF) [20]. Unlike the traditional Bloom filter, SCBF uses a filter made up of one groups of hash functions. Each group can be viewed as a traditional Bloom filter. One group of hash functions is chosen randomly and the bits are set to 1 according to the hash value of arrived packet. When the SCBF becomes full, it will be paged to persistent storage devices. A query concerning the size of a flow can be made to a SCBF page stored on the disk. The result of the query is approximate number of packets in the flow during the measurement epoch recorded by that SCBF page. Other metrics of flows besides packets number, for example packet size distribution and application type, which are very important for network operation, are difficult to be measurement with SCBF. 18 GB per hour for a OC-192 link to store the SCBF pages is

needed. To generate the statistics report of flow, say weekly report, heavy SCBF pages need to be queried. Farther more, the false positive is avoidless in SCBF. In our 3-phase flow measurement approach, the flow metrics can be added easily. The storage for flow in probe is stable according to time. The metrics are analyzed in probe and stored into database periodically, which makes report generating easily.

6. Conclusion

Accurate flow measurement is important in a number of network applications. However, current solutions such as maintaining per-flow state or sampling are either not accurate or not scalable for new metrics. A 3-phase flow measurement method is proposed in this paper. The method can meet the requirement of very high speed link and output the real time report and history report of flow metrics flexibly. The flow metrics is scalable reconfigurable and It is also convenient to analyze the host behavior with our approach in real time by traversing the linear link table of a node of the tri-tree. The method has been implemented in different platforms with GE, OC-48 POS and 10GE interface. Experiments with Carrier's backbone, instrument and trace show efficiency of the method.

7. References

[1] Kuai Xu, Zhili Zhang, and Supratik Bhattacharyya, Profiling Internet Backbone Traffic: Bahavior Models and Applications, SIGCOMM 2005, August 21-26, 2005, Philadelphia, Pennsylvania, USA.

[2] Gaogang Xie, Yinhua Min, Dafang Zhang, Router Performance Analysis Based on a Periodical Logarithmic Traffic Model, Chinese Journal of Computers, Vol.25, No.12, Dec.2002

[3] A. Lakhina, M. Crovella, and C. Diot, Characterization of Network-Wide Traffic Anomalies in Traffic Flows, In Proc. Of ACM SIGCOMM Internet Measurement Conference, 2004

[4] Cristian Estan , George Varghese, New directions in traffic measurement and accounting, Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications, August 19-23, 2002, Pittsburgh, Pennsylvania, USA

[5] Abdesselem Kortebi, Luca Muscariello, Sara Oueslati and James Roberts, Minimizing the overhead in implementing flow-aware networking, in Proceedings of the 2005 Symposium on Architecture for Networking and Communications Systems, Princeton, NJ, USA, 2005

[6] Case J. D., Fedor, M., Schoffstall M. Letal, Simple network management protocol, RFC 1157, 1990

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

[8] P. Phaal, S. Panchen, and N. McKee. InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks. IETF, RFC 3176, September 2001.

[9] C. Estan, G. Varghese, and M. Fisk. Bitmap algorithms for counting active flows on high speed links. In SIGCOMM Internet Measurement Conference, 2003.

[10] Hyany-Ah Kim, David R. O'Hallaron, Counting network flows in real time, In IEEE GLOBECOM, 1-5 Dec. 2003

[11] Jianhua Yang, Gaogang Xie, and Zhongcheng Li, An Efficient Algorithm for Flow Measurement, Journal of Electronics, 2006

[12] N. Duffield, C. Lund, and M. Thorup. Estimating Flow Distributions from Sampled Flow Statistics. IEEE/ACM Transactions on Networking, v.13 n.5, p.933-946, October 2005

[13] Jianhua Yang, Gaogang Xie, and Zhongcheng Li, A Best Effort Adaptive Sampling Method for Flow-Based Traffic Monitoring, Journal of Computer Research and Development, 43(3), 402-409, 2006

[14] Tatsuya Mori, Masato Uchida, Ryoichi Kawahara, Jianping Pan, and Shigeki Goto, Identifying elephant flows through periodically sampled packets, IMC 2004

[15] Cisco Systems. Netflow services solutions guide. white paper, http://www.cisco.com/univercd/cc/td/doc/cisintwk/intsolns/netflsol/nfwhite.pdf, July 2006.

[16] Gaogang Xie, Jianhua Yang, Junfeng Wang and Zhongcheng Li, A Methodology of Effective Measurement for Link Traffic, Proceeding of International Conference on Communication Technology, April 2003,Beijing

[17] K. Claffy, H.W. Braun, and G. Polyzos, A Parameterizable Methodology for Internet traffic flow profiling, IEEE Journal of Selected Areas in Communications, 1995

[18] M. Molina, S. Niccolini, N.G. Duffield, A Comparative Experimental Study of Hash Functions Applied to Packet Sampling, ITC-19, Beijing, 2005

[19] Public OC-48 backbone trace of Abilene-I, ftp://pma.nlanr.net/traces/long/ipls/1/

[20] Abhishek Kumar, Jun Xu, Jia Wang, Oliver Spatschek, Li Li, Space-Code Bloom Filter for Efficient Per-Flow Traffic Measurement, ACM/USENIX Internet Measurement Conference (IMC-03), Miami, Oct 2003.