An Early Traffic Sampling Algorithm

Hou Ying(✉), Huang Hai, Chen Dan, Wang ShengNan, and Li Peng

National Digital Switching System Engineering & Techological R&D center,
ZhengZhou, 450002, China
ndschy@139.com, hh@mail.ndsc.com.cn, cd@mail.ndsc.com.cn

Abstract. The first several packets of a flow play key role in the on-line traffic managements. Early traffic sampling, extracting the first several packets of every flow, is raised. This paper proposes a structure named CTBF, combination of counting Bloom Filter and time Bloom Filter. Based on it, the algorithm is designed to realize automatically removing the space occupied by the timeout flow. The analyses and experiments demonstrate that the sampling accuracy of CTBF is better than that of LRU and Fixed-T algorithm in the same space.

Keywords: Traffic classification · Bloom Filter · Early Traffic Sampling

1 Introduction

The first several packets of a flow play key role in early traffic classification, traffic monitor and early warning etc. Early traffic classification has become an essential means for network security. Researchers collect and analyze the first several packets of the flow to identify the application of the traffic [1][2]. In [3], Zhang.H.L has proved that it can get relatively high accuracy of traffic classification through the statistical features of the first four packets. With the increasing of network bandwidth, the costs of storage and calculation increase sharply. Thus, the early traffic sampling, how to collect the early packets in establishment stage of a flow, is proposed, especially in high-speed network.

The early traffic sampling is different with the classical uniform random sampling and fixed periodic sampling. It can be defined as follows:

Let F denote the sequence of packets $\{f_1, f_2, \ldots f_N, f_{N+1}, \ldots f_M\}$, N is the number of packets to be sampled. The sample probability of the *i*th packets is p_i, and

$$p_i = \begin{cases} 1, & \text{when i} \le \text{N} \\ 0, & \text{when i>N)} \end{cases} \tag{1}$$

As data preprocessing, the accuracy of the early traffic sampling directly determines the accuracy of the subsequent traffic identification. The core of the early

This research was supported by a research grant from the National Natural Science Foundation of Chinese government [61309019].

V.C.M. Leung et al. (Eds.): TridentCom 2014, LNICST 137, pp. 425–433, 2014.
DOI: 10.1007/978-3-319-13326-3_41

traffic sampling in high speed network is how to quickly and correctly locate and record the information of each flow. The early traffic sampling has remained elusive.

The contributions of this paper are: First, We raise the meaning and give the definition of early traffic sampling. Second, we propose a CTBF (Counter and Timer Bloom Filter) structure suitable of early traffic sampling. At last, the accuracy, space complexity and time complexity of CTBF are analyzed.

This paper is organized as follows. In section 2, we review the previous work in traffic sampling algorithm. Section 3 presents the structure CTBF and describes the principle. Section 4 focuses on the theoretical analysis. Section 5 gives the experimental results of evaluation in accuracy, space and time complexity. Section 6 is the summary of this paper.

2 Related Works

For early traffic sampling, the common approach is to maintain per-flow state table and record the number of the sampled packets. When receiving a packet, locate the position of the corresponding flow in the table by hash functions, query the number of the sampled packets and compare the number with N to judge whether to sample. Drawback of this method is that hash collisions will lead to leakage and the cost is too high to resolve the conflicts with linked list. So it is not suitable for high-speed network.

CBF, Count Bloom Filter [4], usually used in long flows identification, can also be used for early traffic sampling. CBF can improve packet processing speed and reduce the computational complexity. Drawback of this method is: as the IP flow length in the internet obeys heavy-tailed distribution, where a few flows with large bytes occupy most of the network traffic, and in the life cycle of long flow, the counter of bloom filter must remain valid. So as the time goes by, more and more counters become nonzero, resulting in the increasing of leakage sample probability. For normal TCP flow, we can judge the end to clear the counter by FIN/RST. But in the real network, there are more and more UDP and abnormal TCP flows. Usually they rely on the timeout mechanism to judge the end. In the field of network measurement, many researchers have focused on how to set up a reasonable timeout mechanism for these flows. If the timeout is set longer, the end flows occupy memory and increase the processing load of the system. And the shorter timeout may lead to a single flow be mistaken for multi flows.

In [5], Claffy judged the end of flow by the fixed-T mechanism. The timeout is set to be 64s. The experiments verified that this method has a good effect in most cases. But when the short flow peak appears (DDoS attack or worm outbreak), short flows cannot be timely released. That will increase the consumption of system resource. Some researchers designed adaptive timeout mechanism to judge the end of flow [6][7].In [6], Ryu B proposed MBET algorithm, according to the number and the interval of packets to dynamically adjust the timeout of a flow. That algorithm judges the end of flows and releases the resource as soon as possible. But adaptive timeout mechanisms require history information to calculate the flow characteristics, thereby

adjusting the timeout. That is too complicated and cannot be applied to high speed networks.

LRU (least recently used) algorithm was proposed in [8] to detect the long flows. That can be used in the early traffic sampling. The core of the algorithm is that the least recently used flow is replaced when a new flow arrives. This algorithm only maintains the current active link and need not periodically scan the table to release the flow. That reduces the cost of system. But this algorithm needs hash function to locate the flow table. The complexities of time and space to solve the hash conflict are large. And when a large number of sudden small flows arrive, the active flows may be replaced.

In addition, the time Bloom filter flow sampling algorithm [9] extracts the first packet of per flow. That cannot meet the needs of early flow sampling.

3 Description of CTBF

CTBF structure consists of k independent hash function $h_1, h_2, ..., h_k$ and two vectors, V_1 and V_2. The numerical value of each hash function are independent and the range is {1, 2,... , m}. Each dimension of the vector V_1 is set to be a counter, denoted as *C(i)*. Each dimension of the vector V_2 is set to be a timer, denoted as *t(i)*. Both of the initial values are set to zero.

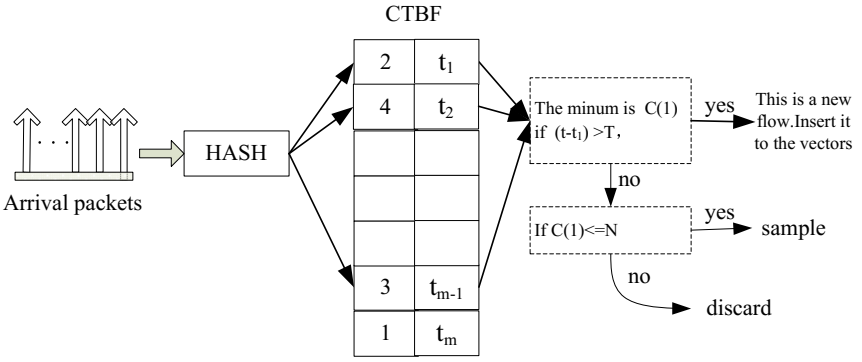

Fig. 1. CTBF structure

Figure 1 is the schematic diagram of the CTBF structure and the sampling algorithm. Assuming that N is the sampling number of per flow and T is a preset timeout of flow interval. The main process of the CTBF algorithm is:

1) When a packet arrives at time *t*, extract flow identification (five-tuple: source IP address, source port number, destination IP address, destination port number , protocol type) as the input of hash function and get k hash results $h_j(s)(1 \le j \le k)$.

2) Calculate $\Delta t_j = t - t(h_j(s))$, $1 \le j \le k$.

3) update the timer in V_2: $t(h_j(s)) = t$.

4) When one $\Delta t_j \geq T$, the packet is the first of a new flow. Sample this packet and update the counter in V_1: $c(h_i(s)) = 1$, where i satisfy $\Delta t_i \geq T$ and $1 \leq i \leq k$.

5) If for any j（$1 \leq j \leq k$）, $\Delta t_j \leq T$, then the flow has begun sampling. Judge the sampling number by $c\_min = \min(c(h_j(s)))$. If $c\_min < N$, sample and update the counters in V_1: $c(h_j(s)) = c(h_j(s))+1$, $1 \leq j \leq k$. Otherwise discard this packet.

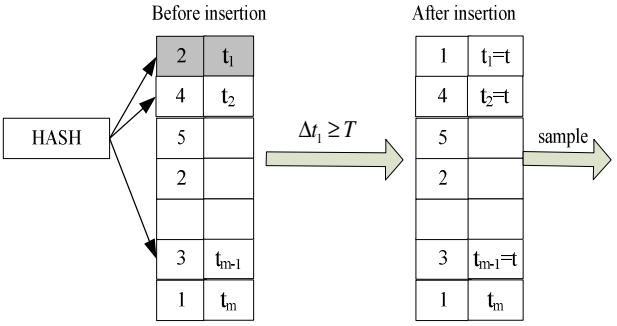

Fig. 2. When first packet of a new flow arrives at t moment

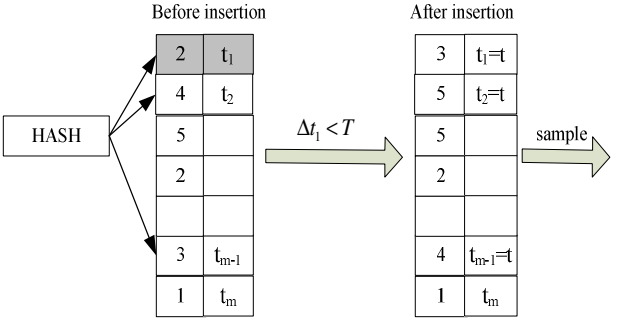

Fig. 3. When a packet of an existing flow arrives at t moment

Figure 2 and figure 3 show the vector's modification when a new flow arrives and when a packet of an existing flow arrives at t moment.

4 Algorithm Analyses

Early traffic sampling algorithm is evaluated from space complexity, accuracy and computational complexity. This several aspects restrict each other. The general goal is

reducing computation complexity and space complexity in the range of accepted false probability.

4.1 False Positive Probability of CTBF

The false positive probability of early traffic sampling algorithm is defined as: when the i-th packet, i<N, arrives, the packet doesn't be sampled. As a concise synopsis data structure, there is false positives probability in Bloom filter. According to the principle of the CTBF algorithm, the algorithm makes mistakes in the following circumstances: when first packet of a new flow arrives, for any j（$1 \le j \le k$），$\Delta t_j \le T$. The following is the theoretical analysis of false positive probability of the CTBF algorithm.

Suppose there are n concurrent flows and m is the length of vector V_1. Since the threshold N is very small, to simplify the analysis, assume the hash functions' result is completely random, and the concurrent flows number don't change before a new flow been sampled. So the probability that the counter in the vector equals zero is $p = (1 - \frac{1}{m})^{nk}$. The probability has nothing to do with sampling threshold. A new flow is false judged when all $\Delta t_j \le T$（$1 \le j \le k$）. Then the new flow will not be sampled because it is judged as an existing flow. Therefore, the false positive probability of a new flow is:

$$p_{ctbf} = (1-p)^k = \left(1 - (1 - \frac{1}{m})^{kn}\right)^k \approx \left(1 - e^{-kn/m}\right)^k$$

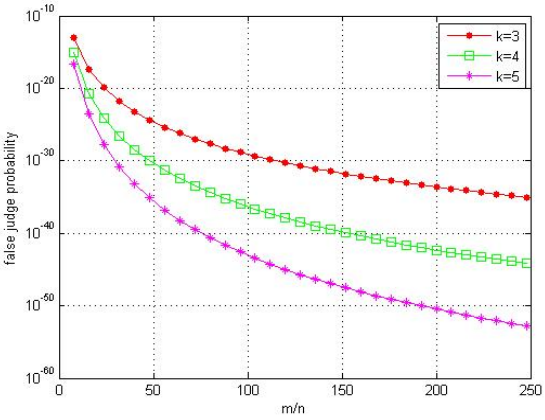

Fig. 4. False positive probability with diffident m/n, in different k

Figure 4 shows the change of p_{CTBF} with the ratio of m/n, when k is 3,4 and 5. It can be seen that p_{CTBF} monotonically decreases with m/n increasing. And the larger k makes the smaller p_{CTBF}.

The figure also shows that the false positive probability is associated with n, the number of concurrent flows. So it is associated with timeout threshold T. Therefore the choice of timeout threshold T influences the false positive probability.

4.2 Time Complexity

The algorithm can be divided into the following sections:

T_h: Time of calculating the k hash functions.
T_q: Time of determining whether $\Delta t_j \leq T$（$1 \leq j \leq k$）.
T_i: Time of verifying the vectors.

For a packet, CTBF algorithm needs computing the k hash functions, judging whether it is a new flow according to Δt_j and modifying V_2, and then deciding whether sampling by V_1 and modifying V_1. So the average handling time of every packet in CTBF algorithm is:

$$T_{ctbf} = T_h + 2 \times (T_q + T_i)$$

Considering $T_q + T_i$ is far less than T_h, so time of CTBF algorithm is almost the time of calculating the k hash functions

4.3 Space Complexity

The space complexity of Bloom filter is measured by the total bits occupied by the vector. When the vector length m is fixed, the algorithm storage space positively correlated with a single counter-digit. However, if the counter-digit is too small, it will lead the counter to be overflow and impact false positives. According to the [4], the width of V_1 is set to be 16bits, assuring the counter cannot overflow in acceptable false probability. The width of V_2 is set to be 32bits and the unit is set to be 100ms. Then the time vector overflow time is 4971 days after algorithm running. That can satisfy the conventional measurement requirements.

So the space of CTBF algorithm is: 16×m+32×m=48×mbit.

5 Experimental Results

Our experiments are based on the passively collected data opened by “National Laboratory for Applied Network Research” (NLANR)[10]. The form of the file is ERF. We use the real trace collected from campus net of Waikato. To protect the privacy, there are only the head of packets in the data file. The duration of the trace is 24 hours. See Table 1 for details.

Table 1. Information of the experimental trace

Data set	File name	duration	Number of flows	Number of Packets
Waikato network	20110407-000000-0	24 hours	31169027	331999280

Set k=4, sampling number N=4. The vectors' length was set to 1000000 and occupied space was 6M bytes. We calculated the accuracy of CTBF when T is from 8 seconds to 96 seconds. The sampling experimental results are shown in Figure 5.

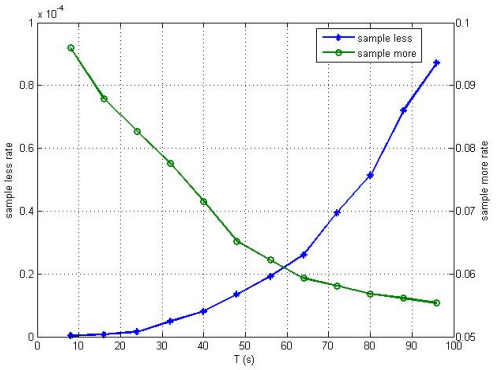

Fig. 5. Sampling results of CTBF with different T

From figure 5 we can see that with T increasing, the less sampling probability increases. That is because the longer the timeout, the more the number of concurrent flow. It is equivalent to m/n decrease, resulting in less sampling. On the other hand, with the decrease of T, some flow may be sampled several times. The reason is that one flow may be truncated into several short flows, resulting in over sampling. We can see that when T=64, the result is more balanced.

The next experiment is compare the sampling result of CTBF, LRU, and Fixed-T with the same space and T=64s. The results are shown in Table 2. The experimental results show that, under the same conditions, the CTBF algorithm over sampling rate is about 4 times less than other algorithms. This is because the other two algorithms need to store five-tuple and linked list pointer, limited by the space. When in the same space, the flow table records are small, resulting in active flow be swapped out. That leads to over sampling. On the other hand, the less sampling rate of CTBF algorithm is between that of LRU and Fixed_T. The reason is in LRU, even though the active flows have been swapped out, the subsequent packets of the flow produce a new flow record and sample again. So LRU algorithm can only produce over sampling but cannot produce less sampling. But in Fixed-T algorithm, when the space is not enough, the packet is discarded. That easily leads to less sampling. So the comprehensive accuracy of CTBF is better than that of the other two algorithms.

Table 2. Information of the experimental trace

	CTBF	LRU	Fixed_T
Over sample（10e-2）	1.72	4.11	3.96
Less sample（10e-2）	0.0026	0	4.93

We evaluated the time complexity of the CTBF, LRU and Fixed-T, base on the Waikato campus dataset. The experiments are conducted on a laptop with Intel i5 CPU, 2.50GHz frequency and 4Gbytes RAM.

To avoid the affect of the disc IO on the conduct time, we read the first hour data of the set to the memory in advance and test with the algorithms. We got PPS (Packets Per Second) of the algorithms when the number of hash functions is 1-7.

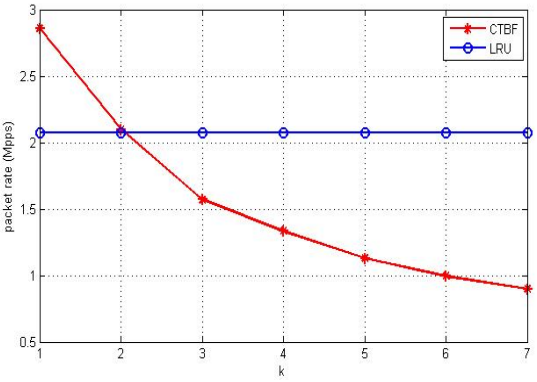

Fig. 6. The rate of CTBF and LRU in different hash functions

The results are shown in figure 6. The latter two algorithms' conduct rates are almost equal. With the increasing of hash functions, the handle rate of CTBF decreases rapidly. Serially handling with hash functions occupies the most time in the emulation and consumes lots of the resource of CPU. When designing an actual system, we can design parallel processing in hardware to raise the rate.

6 Conclusions

The contribution of the paper is the proposed of early traffic sampling. And this paper presents a structure and algorithm suitable for early traffic sampling. The structure consists of two vectors, counting Bloom Filter and timer Bloom Filter. The two vectors cooperate to realize early traffic sampling. The counting Bloom Filter for high rate sampling counting saves storage space. The timer Bloom filter automatic releases the space when timeout, avoiding scanning the flow table regularly. The accuracy, space complexity and time complexity are analyzed and experiments are conducted on the real trace from internet. The experimental results show that when in the same space, CTBF gets better comprehensive accuracy than LRU and Fixed_T algorithm.

References

1. Bernaille, L., Teixeira, R., Akodkenou, I.: Traffic classification on the fly. ACM SIGCOMM Computer Communication Review **36**(2), 23–26 (2006)
2. Li, W., Canini, M., Moore, A.W., Bolla, R.: Efficient application identification and the temporal and spatial stability of classification schema. Computer Networks **53**(6), 790–809 (2009)

3. ZHANG, H.-L., LU G.: Machine Learning Algorithms for Classifying the Imbalanced Protocol Flows: Evaluation and Comparison.Journal of Software, 23(6):1500−1516 (2012)
4. Fan, L., Cao, P., Almeida, J., Broder, A.Z.: Summary Cache: A Scalable Wide-area Web Cache Sharing Protocol. IEEE/ACM Transactions on Networking **8**(3), 281–293 (2000)
5. Claffy, K.C., Braun, H.W., Polyzos, G.C.: A parameterizable methodology for Internet traffic flow profiling. IEEE Journal on Selected Areas In Communications **12**(8), 1481–1494 (1995)
6. Ryu, B., Cheney, D., Braun, H.W.: Internet flow characterization: adaptive timeout strategy and statistical modeling. In Proc, Passive and Active Measurement workshop (2001)
7. Cai, J., Zhang, Z., Zhang, P., et al.: An adaptive timeout strategy for profiling UDP flows. Networking and Computing (ICNC), 2010 First International Conference on. 44–48, IEEE (2010)
8. Smitha, InkooKim, NarasimhaReddy, A.L.: Identifying Long-term High-bandwidth Flows at a Router. In: Proceedings of the 8thInternational Conference on High Performance Computing. Hyderabad, India, 361–371(2001)
9. Kong, S., He, T., Shao, X., An, C., Li, X.: Time-Out Bloom Filter: A New Sampling Method for Recording More Flows. In: Chong, I., Kawahara, K. (eds.) ICOIN 2006. LNCS, vol. 3961, pp. 590–599. Springer, Heidelberg (2006)
10. NLANR. National Laboratory for Applied Network Research [EB/OL]. http://pma.nlanr.net/