INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2024, VOL. 70, NO. 4, PP. 943-951
Manuscript received September 18, 2024; revised October 2024. doi: 10.24425/ijet.2024.152081

Adaptive sampling method for network traffic security monitoring based on queuing theory

Maciej Sosnowski, and Piotr Wiśniewski

***Abstract*—Present network monitoring systems need to cope with the ever-increasing amount of traffic in modern high-speed networks. These systems often perform sophisticated deep packet inspection (DPI) for anomaly detection, denial-of-service attacks detection and mitigation, intrusion detection and prevention, etc. Since DPI is resource-intensive, the monitoring devices are often not able to analyze all incoming traffic at link speeds. Consequently, sampling is employed to reduce the traffic volume and thus limit packet losses caused by resource exhaustion. Classical sampling methods select packets based on a fixed limiting parameter, regardless of the computational resource utilization of the monitoring device.**

This paper proposes a novel sampling approach for network traffic security monitoring that is based on an analytical model of the monitoring device. The model allows for testing adaptive sampling strategies that adjust the instantaneous sampling rate according to the input queue occupancy. The queue occupancy is used to drive the adaptation as it indicates the current relationship between available computational resources and the input traffic volume. Consequently, our approach maximizes the DPI ratio while simultaneously ensuring that the probability of packet loss due to resource exhaustion remains negligible. Analytical and simulation results are presented to demonstrate the impact of the proposed method on system parameters, along with a comparative studies.

***Keywords*—sampling; DPI; network monitoring; system state distribution**

I. Introduction

DEEP packet inspection (DPI) is a commonly used state-of-the-art method employed in a variety of network monitoring systems, such as for anomaly detection, denial-of-service (DoS) attacks detection and mitigation, intrusion detection and prevention, quality of service (QoS) monitoring, traffic classification, and accounting.

These systems need to cope with the ever-increasing amount of traffic of current high-speed networks. Since DPI is resource-intensive, the monitoring devices are often not able to analyze the whole incoming traffic at link speeds. Consequently, sampling is employed to reduce the traffic volume and, thus, limit packet losses caused by resource exhaustion. Sampling algorithms, essentially, select packets from monitored traffic for further DPI analysis. The rest of the traffic is forwarded without inspection. Sampling techniques aim to balance scalability and accuracy.

The classical sampling methods select the packets based on preconfigured static limiting parameter. These methods can be classified into count-based and time-based [1],[2]. Count-based approaches define numbers/counts of packets which are selected. A systematic count-based method selects every m-th packet, probabilistic method selects a packet with given probability, while *n-out-of-N* algorithm selects random set of n packets every N incoming packets. Time-based solutions define times/intervals when the packets are selected. A systematic time-based approach selects packet every l seconds and probabilistic method randomly selects inter-selection time.

Consequently, count-based algorithms yield a DPI rate proportional to the incoming rate, whereas time-based algorithms provide a uniform DPI rate. Note that all classical sampling methods select packets based on a fixed limiting parameter, regardless of the computational resource utilization of the monitoring device. Assessing the optimal value of the limiting parameter might be difficult [3] due to its fixed nature [4]. This may lead to temporary situations where: 1) the incoming packet volume is relatively low, resulting in underutilization of the monitoring device's computational resources, causing many packets to unnecessarily bypass DPI inspection, or 2) some packets are lost during traffic surges due to resource overutilization caused by DPI.

To maximize the number of analyzed packets, adaptive methods have been proposed that adjust the sampling rate (limiting parameter) based on the computational resource utilization of the monitoring device [3]-[6]. The [4] proposes a NetFlow extension that adapts the sampling rate based on memory usage in predefined time bins. The authors argue that under traffic surges, the NetFlow router's memory and processing power may become exhausted, and thus memory utilization is used as a feedback signal to adjust the sampling rate. At the start of each time bin, the sampling rate is set to the maximum value (at which the processor can operate under worst-case conditions), and is then adaptively reduced based on consumed memory resources. The [7] proposes probabilistic sampling method for NetFlow for anomaly detection. The sampling rate is adjusted per flow based on its characteristics, but it is note dependent on the computational resource utilization of the monitoring device.

The [5] proposes to use a prediction model to adapt the sampling rate based on the incoming traffic features and

This work was partially supported by the Polish National Centre for Research and Development under Project CYBERSECIDENT/381651/II/NCBR/2018 (TAMA project).

Maciej Sosnowski and Piotr Wiśniewski are with Warsaw University of Technology, Institute of Telecommunications, Poland (e-mail: maciej.sosnowski@pw.edu.pl, piotr.wisniewski2@pw.edu.pl) and National Institute of Telecommunications, Poland.

foreseen CPU usage. The traffic features are selected to predict the CPU usage and drive the adaptation rate. Earlier, the adaptive sampling based on CPU utilization was proposed in [6] as one of the two approaches (first based on CPU utilization, second on packet interarrival time). The CPU utilization was not directly measured but calculated by dividing the desired processing time by the average time since the beginning of the trace.

The [3] proposes to adapt the sampling ratio on base of the queue occupancy using the proportional–integral–derivative (PID) controller. The sampling ratio is dynamically adjusted to keep the queue occupancy in desired region. Thus, that this solution does not directly monitor the resource utilization like memory or CPU (but uses queue occupancy to guide adaptation similarly to our approach). The proposition analyzes the beginning of connections and is designed to run on multicore commodity hardware.

Other adaptive sampling approaches try to adjust sampling strategy to the traffic characteristics, not considering the computational capacity of monitoring device. The [8] proposes an adaptive sampling methodology that adjusts the time interval between consecutive samples based on aggregated time-serial evolution trend. The approach aims to decrease samples and reflect more information about network traffic burstiness. The [9] adaptively adjust the sampling parameters based on the estimated traffic rate. The [10] proposes an adaptive packet sampling technique tailored for botnet detection. It exploits network characteristics of botnet command and control traffic to inspect higher number of packets related to bots. The sampling probability is adaptively adjusted to keep a target sampling rate without monitoring computational resources. The method aims to inspect smaller amount of suspicious traffic and thus improve scalability.

In this paper, we propose a novel sampling approach based on an analytical model of the monitoring device. The model employs queuing theory of embedded Markov chain. It allows for dynamic adaptation of the instantaneous sampling rate in according to input queue occupancy. Generally, when the queue occupancy is relatively low, the monitoring device has sufficient resources to analyze all traffic. When the occupancy grows, there is a higher risk of resource exhaustion (resulting in queue overflow), thus lower DPI ratio is preferred. Our approach bounds the probability of packet loss (due to resource exhaustion) while simultaneously maximizing the DPI ratio. The following section II presents our adaptive sampling approach with explanation of the analytical model of DPI device with adaptive sampling. Section III provides simulation results to demonstrate the impact of the proposed method on system parameters, along with a comparative studies. Finally, section IV sums up the paper and describes future works.

II. MODEL OF DPI SYSTEM WITH ADAPTIVE SAMPLING

This chapter introduces a queuing model of a DPI monitoring device with adaptive sampling. We assume that packet batches are recursively taken for service, and for each batch, a sampling algorithm determines the number of packets to undergo DPI based on the instantaneous queue occupancy. We consider the queue occupancy to be sufficient information to drive the instantaneous sampling rate. Such information is readily available on monitoring devices, regardless of where the actual resource bottleneck lies (e.g., CPU speed, memory access speed). Moreover, fine-grained measurement of CPU usage is a challenging task [5] and may be unreliable in current solutions like DPDK [11], where the CPU is typically fully utilized regardless of the traffic volume[1].

Our aim is to maximize the DPI ratio, R_{DPI}, while bounding probability of packet loss, P_{loss}, to negligible (acceptable) level. The P_{loss} can be expressed as the mean packet loss ratio (the ratio of lost packets to total incoming packets), while the R_{DPI} is the ratio of packets undergoing DPI to all forwarded packets. Note that optimization of these factors is contradictory. The higher the R_{DPI} is, the more time is required to analyze the traffic, thus the higher is the probability of packets losses due to the queue overflow. Consequently, our method adapts the instantaneous sampling rate based on the queue occupancy. When the occupancy is low, there are probably sufficient resources to DPI all packets. Whereas, when it is relatively high, just a fraction of packets should undergo DPI to reduce the risk of packet loss.

Keeping the P_{loss} at negligible level is more important than analyzing the whole traffic. Simultaneously, the higher the traffic volume analyzed, the better, especially for security applications, as it lowers the risk of missing a malicious packet. Therefore, our approach allows for increasing the instantaneous sampling rate when there is enough computational capacity to process the majority of the current traffic volume with DPI.

To calculate P_{loss} and R_{DPI} for different adaptive sampling strategies (different instantaneous sampling rates as a function of buffer occupancy), we propose a model of DPI monitoring device with adaptive sampling. Specifically, we analytically describe the system state at time instances immediately after finishing packet batch service employing embedded Markov chains. Our solution enables calculation of the system state distribution (queue occupancy distribution), P_{loss} and R_{DPI} for different adaptive sampling strategies. The following sections provide a detailed description.

A. Batch service time

We assume that the DPI monitoring device employs burst-oriented optimization techniques to aggregate the cost of processing each packet individually. Consequently, we model the device as a queuing system with batch service. The incoming packets are enqueued individually and serviced in batches of preconfigured maximum size B (ranging from one to K packets). After finishing a service of a batch of packets, the batch is dequeued (removed from the queue and sent), and the new one is taken for service [12].

The DPI monitoring device is assumed to be an inline device that forwards all incoming packets from input network interface cards (NICs) to output NICs performing DPI on a subset of packets (selected by a sampling algorithm). It may be a multi-core, multi-NIC device, like DPDK monitoring device

[1] The main packet processing function requests the packets from network interface card in a loop yielding full CPU utilization even where there are no incoming packets.

described in [13] where each NIC port is assigned to a few CPU logical cores, each bounded to a number of RX queues. The queue may be selected by hashing the IP addresses and port numbers, ensuring no packet reordering occurs [11],[13].

We assume that the batch service time depends on the number of packets in the queue at the start of the service, l. Passing (skipping) a packet requires a fixed processing time T_{skip}, while performing DPI on a packet takes a longer fixed time T_{DPI}. This assumption allows us to easily model different adaptive sampling strategies. For example, when queue occupancy is relatively low, the entire packet batch can undergo DPI analysis (thus the service time equals the number of packets multiplied by T_{DPI}). Otherwise, only a fraction of packets is selected for DPI (the service time equals number of selected packets multiplied by T_{DPI} plus number of skipped packets multiplied by T_{skip}). We assume, that T_{DPI} describes the total processing time of a packet undergoing DPI, and it is independent of packet size. It corresponds to computational complexity of DPI analysis and computational power of the monitoring device. Similarly, we assume that T_{skip} describes the time required to forward a packet form input NIC to output NIC, and it is also independent of packet size. The values of T_{DPI} that T_{skip} can be measured on a real device or assumed a priori.

The adaptive sampling strategy is described with a vector $\boldsymbol{v}$ that specifies the number of packets subjected to DPI based on $l, l \in \{1, 2, \dots, K-1\}$ (number of packets present in the queue at the start of service). The value of $\boldsymbol{v}(l)$, ranging from 0 to $min(l, B)$, describes the number of packets undergoing DPI. The $min(l, B)$ describes the number of packets taken for service (size of the batch), while $min(l, B) - \boldsymbol{v}(l)$ describes the number of packets skipping DPI, when there are l packets enqueued at the service start. Consequently, $\boldsymbol{v}$ describes the instantaneous sampling rate in function of queue occupancy l.

Note that, while each batch for $l \geq B$ has a size of B, the values of $\boldsymbol{v}(l)$ may vary depending on l. For instance, let us consider that $B = 4$, $\boldsymbol{v}(5) = 3$ and $\boldsymbol{v}(8) = 1$. Then when $l = 5$ a batch of four packets is taken for processing, with three of them undergoing inspection. Whereas, when $l = 8$, again four packets are taken to service, but only one is inspected.

Let $T_{batch}(l)$ represent the service time of a batch of packets taken for processing when there are l packets in the queue at the start of the service. Then:

$$T_{batch}(l) = \boldsymbol{v}(l) \cdot T_{DPI} + \big(min(l, B) - \boldsymbol{v}(l)\big) \cdot T_{skip} \quad (1)$$

Note, that $T_{batch}(l)$ does not have to be proportional to l and may even be non-monotonic.

B. System state description

We define the system state S as the number of packets in queue observed in moments just after finishing a service. Let P_s denote the steady-state probability that the system is in state $s, s \in \{1, 2, \dots, K-1\}$ in these moments. Note that the system has capacity of K packets, but the maximum reachable system state S (and also the maximum possible batch size B) equals $K-1$. This comes from the fact that during service period, there is at least one packet in service (still occupying a packet slot in the queue), that will be dequeued immediately after the service period ends, which is the moment when the system state is observed. Note that the states $S = \{1, 2, \dots, K-1\}$ together state transition probabilities (between these states) constitute an embedded Markov chain.

Consider that the number of packets being in the system at the beginning of a service period, denoted as l, and $l \in \{1, 2, \dots, K-1\}$, equals the system state just after the previous service period ends only for states $S \geq 1$. When the system reaches state $S = 0$, the service process stops, and another service period will start immediately after the first new packet arrival, hence:

$$P(l = k) = \begin{cases} P_0 + P_1, & for\ k = 1 \\ P_k, & for\ k > 1 \end{cases} \quad (2)$$

In Fig. 1 an example of system state evolution is presented, i.e., how the number of packets in the system changes for an sample arrival process. The system has $K = 7$ and $B = 4$. The x-axis represents the passage of time, and moments t_1 to t_5 indicate the end of a batch service, which are the points at which the system state is observed, i.e., $t_0: S = 5, t_1: S = 4$, $t_2: S = 1, t_3: S = 0$, $t_4: S = 1$, $t_5: S = 0$. It is particularly important to note that when the system empties (e.g., at t_3), the next observed moment will always be the completion of service for the one packet that arrived into the empty system. If packets arrive in a full system, they are lost (indicated by the red arrow on the *Arrival process* track).

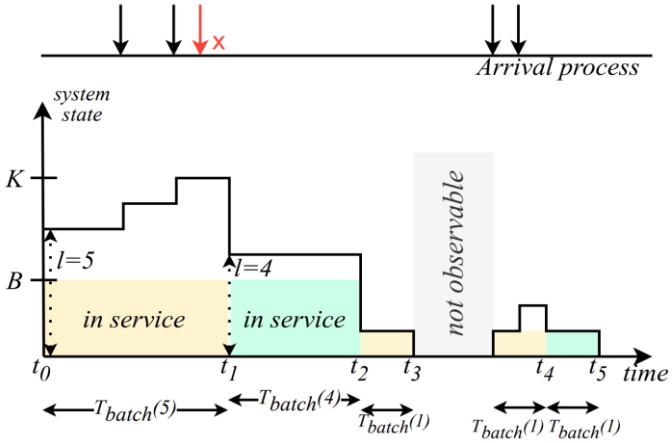

Fig. 1. An exemplary evolution of system state.

In the next three subsections we present system state analysis for three cases. First in subsection *C*, we start with simple special case when batch size equals the system capacity and all enqueued packets are taken into service. Then in subsection *D*, the general case with arbitrary batch size B is investigated. Finally, in subsection *E*, we provide analysis for system state as seen by arriving packets that is used to calculate the packet loss probability P_{loss}. Moreover, in subsection *F* we derive equation for the packet inspection ratio R_{DPI}.

C. System state analysis for special case when $B = K - 1$

Let us start with the simplified analysis of the system state, for the special case when $B = K - 1$, i.e. maximum batch size equals the system capacity[2], and all packets from the queue are taken into service at once when a new service period begins.

[2] As previously described, the actual maximum batch size taken into service (and maximum system state S) equals $K - 1$.

The system can reach state $S = s$, where $s \in \{0, 1, \dots, K-1\}$, only if the number of free packet slots in the queue, at the beginning of the service is at least s. In other words, at the start of the service, there are l packets in the system, such that $l \in \{1, \dots, K-s\}$. During the service of these l packets, a certain number of packets arrives to the system. Note that the service time of a batch consisting of l packets is given by (1) and all $T_{batch}(l)$ values are constants derived from vector $\boldsymbol{v}$ and times T_{DPI} and T_{skip}.

Consequently, the probability P_s is the total probability, taking into account all the possible queue occupancies at service start $P(l = k)$. Thus P_s can be expressed as:

$$P_s = \sum_{k=1}^{K-s} P(l = k) \cdot P_{tr}(s, k), \text{ for } s = 0,1,..,K-1 \quad (3)$$

where $P(l = k)$ is given by (2), and $P_{tr}(s, l)$ represents the probability of transitioning to state s when there were l packets in the system at the start of service. In fact, to reach state $S = s < K - l$, *exactly* s packets must arrive within the time of $T_{batch}(l)$ (time that depends on l). Whereas to reach state $S = s = K - l$ *at least* s packets must arrive within $T_{batch}(l)$. Thus $P_{tr}(s, l)$ can be expressed as:

$$P_{tr}(s, l) = \begin{cases} P_{arr}(s, T_{batch}(l)), & \text{for } s < K - l \\ P_{arr+}(s, T_{batch}(l)), & \text{for } s = K - l \end{cases} \quad (4)$$

where $P_{arr}(m, T_{batch}(l))$ and $P_{arr+}(m, T_{batch}(l))$ are the probabilities that within the time $T_{batch}(l)$ *exactly* m and *at least* m, packets arrive to the system. The $P_{arr}(m, T_{batch}(l))$ distributions can be obtained from arrival process (Poissonian process is assumed), whereas $P_{arr+}(m, T_{batch}(l))$ can be calculated as:

$$P_{arr+}(m, T_{batch}(l)) = 1 - \sum_{i=0}^{m-1} P_{arr}(i, T_{batch}(l)) \quad (5)$$

By substituting (2), (4) and (5) into (3), a system of equations is obtained, where the only unknowns are the values of P_s. One of these equations, e.g., for $s = K - 1$, can be removed and replaced with the normalization condition,

$$\sum_{s=0}^{K-1} P_s = 1,$$

allowing the system state distribution (the values of P_s for $s \in \{0, 1, \dots, K-1\}$ to be calculated.

D. System state analysis for general case $B \leq K - 1$

In the general case, the maximum batch size B may be lower than $K - 1$, so not all packets present in the system at the beginning of a service period are necessarily taken for service. Suppose there are l, $l \in \{1, 2, \dots, K-1\}$, packets in the system when service starts. Then, $min(l, B)$ packets are taken for service.

If no packets arrive in the system during the service, the system will reach state $s_{min} = max(l - B, 0)$. However, if some packets do arrive, this state may increase by at most $K - l$ (the number of free slots in the system at the start of service). This means, that for given B, K, and l the system can reach only states s that satisfy:

$$max(l - B, 0) \leq s \leq max(l - B, 0) + (K - l) \quad (6)$$

Therefore, after some algebra on (6), it can be shown, that a particular state s may be reached only for specific values of l:

$$\begin{cases} l \leq min(s + B, K - 1), & for\ s \leq K - B \\ l \leq K - s, & for\ K - B < s < K - 1 \end{cases} \quad (7)$$

Equation (3) and (4) are no longer valid for the general case $B \leq K$. Because of (7), eq. (3) needs to be split into two cases:

$$P_s = \begin{cases} \sum_{k=1}^{min(s+B,K-1)} P(l = k) \cdot P_{tr}(s, k), & for\ s \leq K - B \\ \sum_{k=1}^{K-s} P(l = k) \cdot P_{tr}(s, k), & for\ K - B < s \leq K - 1 \end{cases} \quad (8)$$

If, for a given value of l, it is possible to transition to state s, this transition will occur if, during the service of the packet batch, $m = s - max(0, l - B)$ packets arrive in the system. To distinguish between cases when *exactly* or *at least* m packets must arrive, it needs to be determined whether the considered state s is the highest possible state achievable for the given l. Since the system has K slots, there are l packets at the start of service, and $s - max(0, l - B)$ packets arrive, s is the highest possible state for $l + s - max(0, l - B) = K$, i.e., when:

$$s = K - l + max(0, l - B)$$

Thus, the transition probabilities to state s for different values of l can be defined as follows:

$$P_{tr}(s, l) = \begin{cases} P_{arr+}\big(s - max(0, l - B), T_{batch}(l)\big), \\ \qquad for\ s = K - l + \max(0, l - B) \\ P_{arr}\big(s - max(0, l - B), T_{batch}(l)\big), \\ \qquad on\ other\ cases \end{cases} \quad (9)$$

for cases, when the transition is possible, i.e., l and s satisfy (7). The remaining part of the reasoning is identical to case presented in subsection II.C.

E. System state seen by arriving packets and packet loss probability

For certain applications, it may be useful to obtain the distribution of the number of packets in the system as observed by the arriving packets, $P_{s_{arr}}(n)$, for $n \in \{0,1, \dots, K\}$. This distribution can be derived from the system state distribution, P_s, along with the distributions of the arrival probability of *exactly* m packets, $P_{arr}(m, t)$, or *at least* m packets, $P_{arr+}(m, t)$, within a given time period t. Note, that $P_{s_{arr}}(K)$ equals packet loss probability, P_{loss}.

Let $E_{arr}(n)$ describe the mean number of packets arriving during the service time (or arriving to an empty system) that observe n packets in the system. It can easily be observed that

$$E_{arr}(0) = P_0.$$

When the system is emptied (with probability P_0), always one packet arrives in the system, and this packet sees the system in state 0. Cases for $n > 0$ concern packets arriving during the service of the previous batch of packets. If m packets arrive in the system, where there are l packets at the start of service, the first (i.e., always one) of these m packets see the system in state l, the second (i.e., always one) in state $l + 1$, and so on. If $m > K - l$, then all subsequent packets, starting from the one indexed $K - l + 1$, see the system in state K – these packets are lost. Thus, the cases for $n < K$ and $n = K$ need to be considered separately.

For example, an arriving packet can observe $n = 2 < K$ packets in the system if $l = 2$ and it is the first of $m \geq 1$ packets entering the system during the service, or if $l = 1$ and it is the second of $m \geq 2$ incoming packets. Therefore:

$$E_{arr}(2) = P(l = 1) \cdot P_{arr+}\big(2, T_{batch}(1)\big) + \\ + P(l = 2) \cdot P_{arr+}\big(1, T_{batch}(2)\big)$$

In general, for $1 \leq n \leq K - 1$, to observe state n, for given l, at least $m = n - l + 1$ packets must arrive. Hence, $E_{arr}(n)$ can be expressed as:

$$E_{arr}(n) = \sum_{k=1}^{n} P(l = k) \cdot P_{arr+}(n - k + 1, T_{batch}(k)) \quad (10)$$

For the case of $n = K$, it must be noted that from an arriving set of m packets, all packets with an index greater than $K - l$ will observe the system in state K. Unlike the cases for $n < K$, more than one packet from the set may observe this state. As a result, the distribution $P_{arr+}(m, t)$ cannot be applied directly. Instead, for each size of arriving packets set, the probability of set being this size should be multiplied by $l + m - K$, i.e. the number of packets that, for that set size, observe the system in state K:

$$E_{arr}(K) = \sum_{k=1}^{K-1} P(l = k) \cdot \\ \cdot \left(\sum_{m=K-k+1}^{\infty} (k + m - K) \cdot P_{arr}\big(m, T_{batch}(k)\big) \right)$$

Now, having all $E_{arr}(n)$ it is possible to determine mean number of packets arriving to the system during the service time (or arriving to an empty system), $E_{sum_{arr}}$, and $P_{s_{arr}}(n)$:

$$E_{sum_{arr}} = \sum_{i=0}^{K} E_{arr}(i),$$

$$P_{s_{arr}}(n) = \frac{E_{arr}(n)}{E_{sum_{arr}}}, for\ n = 0, \dots, K \quad (11)$$

while $P_{loss} = P_{s_{arr}}(K)$.

It is worth noting that the parameter B does not appear in the above relations. It is irrelevant which portion of packets is currently being serviced, as they still occupy slots in the system. However, the value of B affects the batch service times, $T_{batch}(l)$, which are treated as known constants assumed from the system definition.

F. Packet inspection ratio

The last parameter to be determined is the packet inspection ratio, R_{DPI}, which indicates the portion of packets arriving in the system that will be inspected. R_{DPI} can be calculated using two variables that describe the system over one service cycle: by dividing the expected number of packets subjected to inspection by the average number of packets entering the system ($E_{sum_{arr}}$). This calculation method accounts for packets that reached a fully occupied system and were lost, as they are not subject to inspection. The number of packets inspected when there are l packets in the system at the start of service is defined by presumed vector $\boldsymbol{v}$, hence:

$$R_{DPI} = \frac{\sum_{l=k}^{K-1} P(l = k) \cdot \boldsymbol{v}(k)}{E_{sum_{arr}}} \quad (12)$$

In particular, if the values in vector $\boldsymbol{v}$ satisfy $v(l) = min(l, B)$ for each $l \in \{0,1, \dots, K - 1\}$, it implies that all packets present in the system are inspected, and $R_{DPI} = 1 - P_{loss}$.

G. Model validation

To verify the accuracy of the model, a discrete event simulator was implemented in MATLAB. The software was also used for equations solving and numerical calculations.

Let us present results for an exemplary system with parameters $K = 6$, $B = 4$, $\boldsymbol{v} = [1, 2, 3, 4, 4]$, $T_{skip} = 1$, $T_{DPI} = 3$, fed by Poissonian input stream with $\lambda = 0.5$. We compare the analytical and simulation values. The simulation results were obtained from five independent simulation runs, each with different random generator seed. In each run, the system's performance was evaluated for one million packets. Subsequently, 95% confidence intervals were calculated. The results confirm the accuracy of the analytical methods used to determine the distribution of the system state at service completion, S (Table I), the state distributions observed by incoming packets, S_{arr} (Table II), as well as the P_{loss} and R_{DPI} values. Note that the vector $\boldsymbol{v}$ indicates that all served packets were inspected, so in this case R_{DPI} should equal $1 - P_{loss}$.

TABLE I
STATE DISTRIBUTION

S	0	1	2	3	4	5
P	.0587	.1236	.2549	.4609	.0985	.0034
$Sim.$	.0590	.1239	.2548	.4605	.0984	.0034
95%	.0011	.0019	.0004	.0025	.0005	.0001

TABLE II
DISTRIBUTION OF STATES SEEN BY ARRIVING PACKETS

S_{arr}	0	1	2	3	4	5	6
P	.0155	.0375	.0854	.1837	.1824	.1519	.3437
$Sim.$	.0156	.0376	.0854	.1837	.1823	.1517	.3437
95%	.0004	.0007	.0005	.0002	.0003	.0003	.0009

- P_{loss}: 0.3437 vs 0.3435 ± 0.0012
- R_{DPI}: 0.6563 vs 0.6565 ± 0.0012

All subsequent results presented in the article were checked for consistency with simulation results.

III. RESULTS

A. System evaluation

In this chapter, the system parameters will be presented based on the selected vectors $\boldsymbol{v}$. In the first case, a system with parameters $K = 12, B = 4$, $T_{skip} = 1$, and $T_{DPI} = 5$ for $\lambda = 0.5$ is analyzed using five exemplary vectors $\boldsymbol{v}$:

a) No packet inspection.
b) Inspect one packet when there is one packet in the system.
c) Inspect at most two packets when there are at most two packets in the system.
d) Inspect at most two packets when there are at most two packets in the system. If there are more than two packets in the system, inspect one from a batch.

e) Inspect all packets from the batch if there are more than five packets in the system.

The system state distributions, P_s, and the system state distributions seen by arriving packets, $P_{s_{arr}}$, are presented in Fig. 2 and Fig. 3, while P_{loss} and R_{DPI} are shown in Table III. Case b) shows that the most conservative method of selecting packets for inspection can result in a very high R_{DPI} parameter with practically no packet loss. Cases c) and d) demonstrate the possibility of increasing the R_{DPI} parameter at the cost of introducing packet loss, which seems acceptable for case c), but not for case d). In cases a) and e), the distributions P_s and $P_{s_{arr}}$ are very similar. In the overwhelming majority of instances, the system remains in one of the lower states ($s < 6$). For case e), inspection is defined when the system is in a state above 6. Inspecting the entire batch of packets in these higher states very often leads to packet loss. As a result, the R_{DPI} value is very low relative to the introduced P_{loss}.

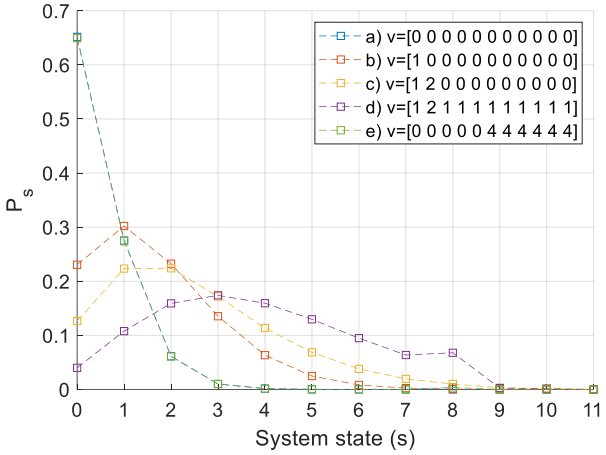

Fig. 2. Impact of vector $\boldsymbol{v}$ on the system state distribution.

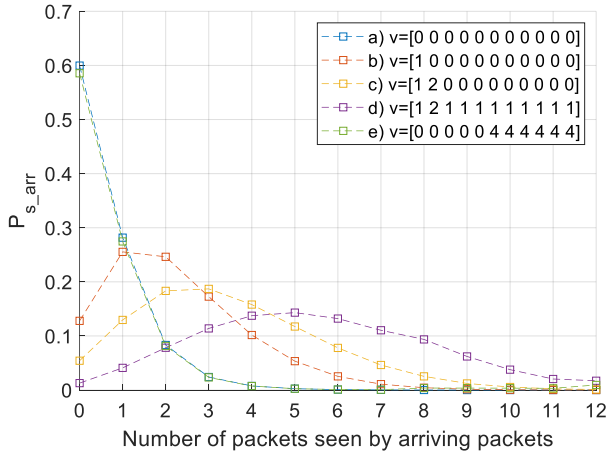

Fig. 3. Impact of vector $\boldsymbol{v}$ on the system state distribution seen by arriving packets.

TABLE III
IMPACT OF VECTOR $\boldsymbol{v}$ ON PLOSS AND RDPI

$\boldsymbol{v}$	P_{loss}	R_{DPI}
[0 0 0 0 0 0 0 0 0 0 0]	< 0.0001	0
[1 0 0 0 0 0 0 0 0 0 0]	0.0001	0.2955
[1 2 0 0 0 0 0 0 0 0 0]	0.0011	0.3418
[1 2 1 1 1 1 1 1 1 1 1]	0.0172	0.3717
[0 0 0 0 0 4 4 4 4 4 4]	0.0094	0.0114

In order to present the system behavior for a capacity close to real conditions, simulation studies were conducted for $K = 128$ and $B = 32$. Analytical solutions for such a large value of K are not feasible within a reasonable time frame. However, the consistency of analytical and simulation results obtained for $K \leq 12$ validates the accuracy of the simulator. Each simulation run involved the observation of ten million packets, repeated five times. The differences in the results of individual runs are negligibly small for the assumed expected accuracy.

The system was evaluated for $\lambda = 0.96$, and the vector $\boldsymbol{v}$ was presumed as:

a) $\boldsymbol{v} = [1{:}8, 119\text{x}0]$, meaning that as long as there are no more than eight packets in the system at the start of service, all of them undergo inspection, and if there are more than eight packets, none are inspected; and

b) $\boldsymbol{v} = [1{:}12, 32\text{x}12, 83\text{x}0]$, meaning that as long as there are no more than twelve packets in the system at the start of service, all of them undergo inspection; when the number of packets is between 13 and 44, only 12 are inspected, and when there are more than 44 packets in the system, none are inspected.

In Fig. 4, Fig. 5, and Fig. 6 the impact of vector $\boldsymbol{v}$ on the system state and on the system state seen by arriving packets, compared to the case with no inspection, is shown. One can see that the values of P_S for $s > 96$ are negligible small. Especially for the vector $\boldsymbol{v} = [1{:}12, 32\text{x}12, 83\text{x}0]$, an outstanding value can be observed for $s = 96$, followed by a drop to nearly zero. This results from the fact that, in an overloaded system, when all slots become occupied, 32 packets leave the system at the end of the service, changing the number of packets in the system from 128 to 96 – then, the system state is observed. States above 96 are achievable, for example, if there were 20 packets in the system at the start of service, and 108 packets arrived during their service. As a result, there would be 108 packets at the end of the service. However, assuming a Poisson arrival process, the probability of such an event is negligible (though theoretically possible).

Values of P_{loss} and R_{DPI} are shown in Table IV. There is almost no difference between R_{DPI} for the two non-zero $\boldsymbol{v}$ vectors, while P_{loss} becomes non-negligible for the second case.

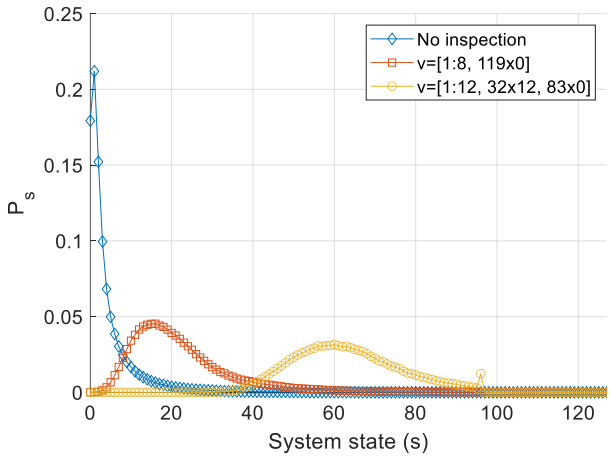

Fig. 4. Impact of vector $\boldsymbol{v}$ on the system state distribution.

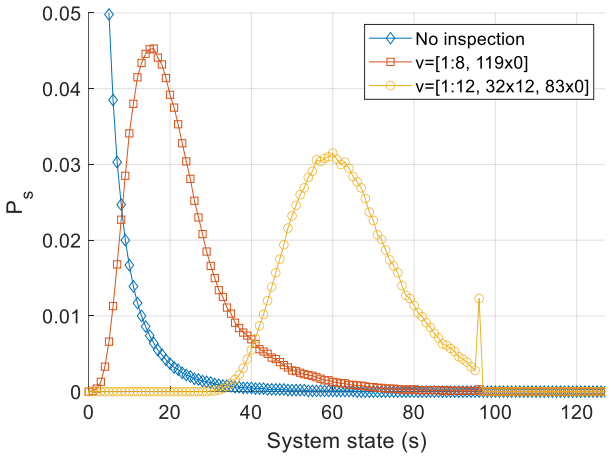

Fig. 5. Impact of vector $\boldsymbol{v}$ on the system state distribution (close-up).

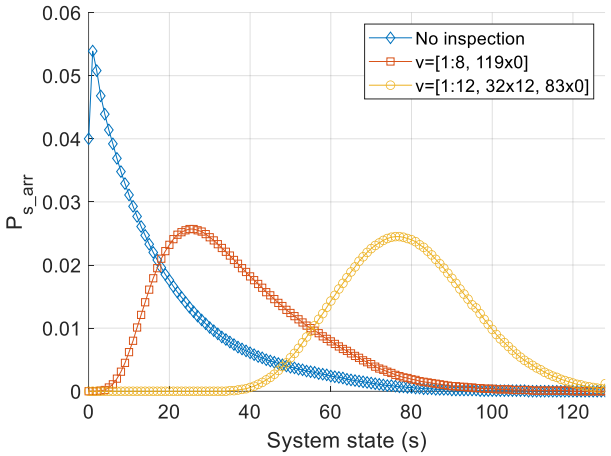

Fig. 6. Impact of vector $\boldsymbol{v}$ on the system state distribution seen by arriving packets.

TABLE IV
IMPACT OF VECTOR V ON PLOSS AND RDPI

$\boldsymbol{v}$	P_{loss}	R_{DPI}
$[127x0]$	< 0.0001	0
$[1{:}8, 119x0]$	< 0.0001	0.020
$[1{:}12, 32x12, 83x0]$	0.001	0.021

B. *Comparison of sampling methods*

In this section, the results obtained from the proposed adaptive sampling method are compared with those from the system where packet sampling is done in random manner (i.e. each packet is taken to the inspection with certain set probability, P_{DPI}). Note, that results for this random sampling method were obtained from simulations, however the confidence intervals were negligible small thus are not presented.

Let us focus on an exemplary system with parameters $K = 6, B = 4, T_{skip} = 1$, $T_{DPI} = 3$, fed by Poissonian input stream with parameter $\lambda \in \{0.2, 0.3, 0.4, 0.6, 0.8, 1\}$. For this system, different vectors $\boldsymbol{v}$ (the first method) and different P_{DPI} values (the second method) were used. Consequently R_{DPI} and P_{loss} are presented as functions of λ in Fig. 7 and Fig. 8.

As expected, results for both methods are the same for two boundary cases: when all or none packets are taken to the inspection (*all* / $\boldsymbol{v} = [1, 2, 3, 4, 4]$ / $P_{DPI} = 1$ and *none* / $\boldsymbol{v} = [0, 0, 0, 0, 0]$ / $P_{DPI} = 0$; solid lines with circle markers). The proposed dynamic sampling method was also evaluated for two additional vectors, $\boldsymbol{v} = [1, 1, 1, 1, 1]$ and $\boldsymbol{v} = [1, 2, 2, 2, 2]$ (lines with square markers). For each of these vectors, the P_{DPI} parameter used in the random sampling method was selected to achieve the same packet inspection ratio R_{DPI} for $\lambda = 1$. For the first vector, this value corresponded to $P_{DPI} = 0.37$, and for the second, $P_{DPI} = 0.68$ (lines with diamond markers).

Comparing the packet loss level graphs (Fig. 8), one can see that the values for the corresponding cases are very similar over the entire range of the λ parameter. In the case of random sampling, each served (i.e. not lost) packet is inspected with a probability of P_{DPI}. Therefore, in the absence of losses, the R_{DPI} parameter would equal P_{DPI}. The decrease in the R_{DPI} value as the load increases is due to a growing proportion of packets being lost.

On the other hand, in the case of dynamic sampling, for the selected vectors $\boldsymbol{v}$, when the system load decreases, the proportion of packets subjected to analysis increases significantly. This is because, when only a small number of packets are taken to service, a large portion of them is analyzed (specifically, one out of one). For instance, for the vector $\boldsymbol{v} = [1\ 1\ 1\ 1\ 1]$, only 1 packet from a batch is always inspected (with a maximum batch size of $B = 4$). However, under low load conditions, such as $\lambda = 0.2$, the batches typically consist of just one packet, so the R_{DPI} is relatively high in this cases.

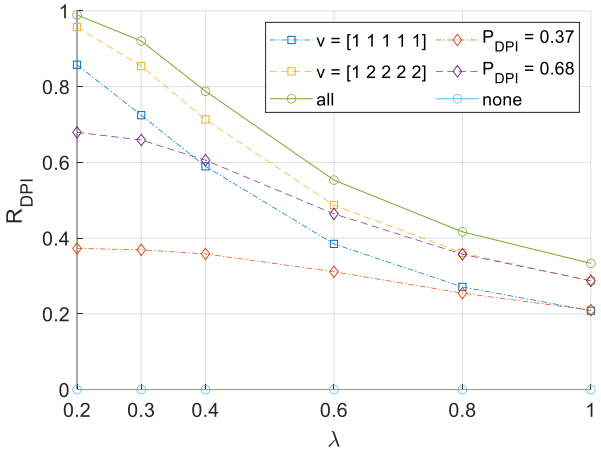

Fig. 7. Comparison of sampling methods: R_{DPI}.

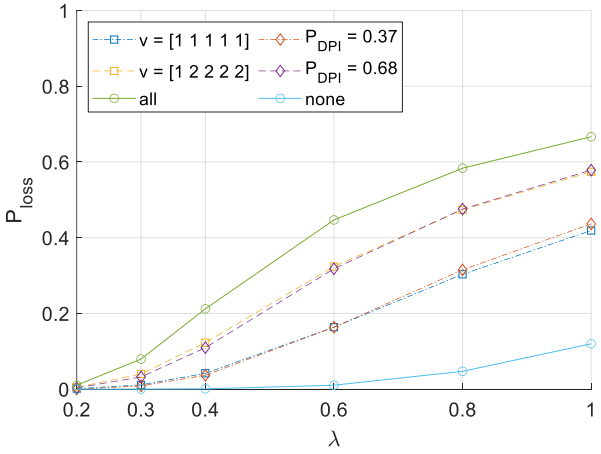

Fig. 8. Comparison of sampling methods: P_{loss}.

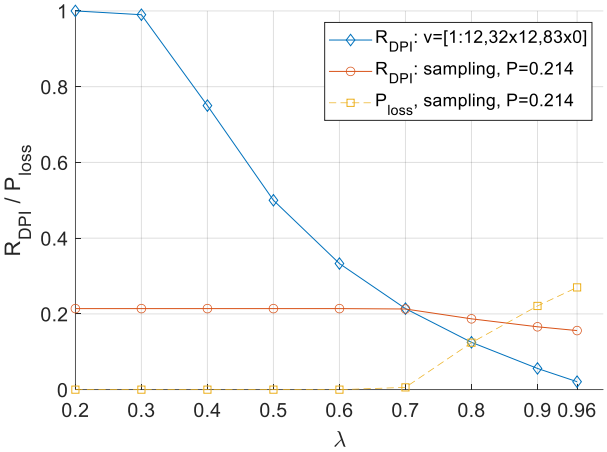

Fig. 9. Comparison of sampling methods: P_{loss} and R_{DPI}.

Same as in previous chapter, in order to present the system behavior in terms of the R_{DPI} and P_{loss} parameters for a capacity close to real conditions, simulation studies were conducted for $K = 128$ and $B = 32$ (each simulation run involved ten million packets, repeated five times). In the simulation, packet arrival followed a Poisson process, and the maximum allowable system load was assumed as $\lambda = 0.96$. At this load level, without packet inspection, the P_{loss} parameter is below 0.0001. Then, a vector $\boldsymbol{v} = [1{:}\,12, 32\text{x}12, 83\text{x}0]$ (see chapter III.A) was selected so that for the maximum allowable load of $\lambda = 0.96$, it did not exceed the arbitrarily set packet loss probability, $P_{loss} = 0.001$, while R_{DPI} is maximized. It must be noted, that many different vectors $\boldsymbol{v}$ satisfy this condition. This phenomenon will be addressed in future works.

Figure 9 shows the R_{DPI} parameter for selected vector $\boldsymbol{v}$ as a function of system load, λ (blue line, R_{DPI}*: vector* $\boldsymbol{v}$). For all values of $\lambda \leq 0.9$, the packet loss level, P_{loss}, remains below 0.0001. Similar to what was observed in Fig. 7, there is a significant increase in the R_{DPI} parameter as the load decreases - the lower the system state, the larger the proportion of packets that are inspected. It is worth noting that the R_{DPI} values for certain λ values, such as λ / R_{DPI}: 0.8 / 0.125, 0.6 / 0.333, 0.5 / 0.5, 0.4 / 0.75, suggest the existence of some property of the system that is not yet fully understood.

In the figure, to comparison, the results for the sampling method with a fixed probability of $P_{DPI} = 0.214$ are presented (R_{DPI}*: sampling,* $P = 0.214$). This value of P_{DPI} corresponds to the R_{DPI} value for a system load of $\lambda = 0.7$, which was assumed as the typical system load. Same as observed in Fig. 7 and Fig. 8, one can see that R_{DPI} for this random sampling method cannot exceed P_{DPI} when the load is lower than $\lambda = 0.7$. However, the lack of adaptation at higher loads results in a higher R_{DPI} than in the proposed method, but this comes at the cost of introducing very large, unacceptable losses.

CONCLUSIONS AND FUTURE WORKS

This paper presents a general queuing model of the DPI monitoring device with adaptive sampling. The proposed approach dynamically adjusts the sampling rate according to system queue occupancy, striving to maximize the percentage of packets undergoing inspection while keeping the probability of packet loss negligibly small. By employing an analytical model of the monitoring device, the method allows for the calculation of key system parameters, such as system state distribution, packet loss probability, and the DPI ratio, under different traffic conditions. The results demonstrate that the adaptive method significantly improves system performance compared to classical sampling methods. This work offers insights into optimizing network monitoring efficiency, especially in high-traffic environments.

In our future works we plan to investigate and adjust our model to a DPDK-based network monitoring device. Specifically, we will adapt the model to reflect the network packet processing chain on multicore commodity hardware (a standard "COTS: Commercial Off-The-Shelf" Linux x86 server) with a DPDK-supported NICs running DPDK application that performs DPI for network security (e.g. as described in [13]). We also plan to assess the performance of our adaptive solution based on measurements on DPDK-based COTS server.

There is also a phenomenon, mentioned in Chapter III.B, whereby many vectors $\boldsymbol{v}$ result in the same maximum achievable R_{DPI} (assuming negligible packet loss). For example, for a system with $K = 128, B = 32$, $T_{skip} = 1$, $T_{DPI} = 3$, and a load of $\lambda = 0.6$, the maximum achievable R_{DPI} is 0.333. This value can be achieved e.g. when the vector $\boldsymbol{v}$ is in the form $[127x\ min(l, N)]$ (i.e., as long as there are no more than N packets in the system, all packets are inspected, and if there are more than N, only N are inspected), for any $N \in \{3, 4, \ldots, 9\}$ (for $N < 3, R_{DPI} < 0.333$; for $N > 9, R_{DPI} > 0.333$, but packet losses occur). In Fig. 10, the first 60 values of system state distribution obtained for three vectors $\boldsymbol{v}$, for $N = \{3, 6, 9\}$, are shown. Shifting the distribution mass toward higher states is undesirable, for example, in terms of delays, and if it does not increase the R_{DPI} parameter, it should be avoided. We plan to describe the phenomenon, derive maximum achievable R_{DPI} (with negligible packet loss) for a given system, and develop a multi-criteria method for selecting vector $\boldsymbol{v}$, which will differentiate vectors $\boldsymbol{v}$ that are identical in terms of the P_{loss} and R_{DPI} parameters, for example, based on their impact on packet waiting times.

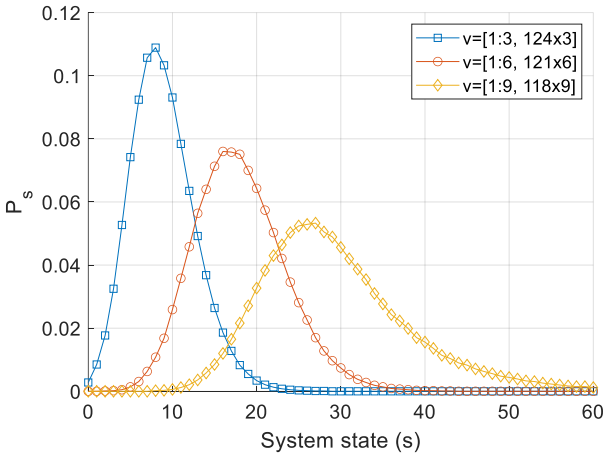

Fig. 10. Comparison of vectors $\boldsymbol{v}$ providing the same R_{DPI} and no packet loss.

REFERENCES

[1] N. Duffield, "Sampling for Passive Internet Measurement: A Review," Statististical Science vol. 19, no. 3, pp. 472-498, 2004. https://doi.org/10.1214/088342304000000206

[2] G. Roudière and P. Owezarski, "Evaluating the Impact of Traffic Sampling on AATAC's DDoS Detection," in Proceedings of the 2018 Workshop on Traffic Measurements for Cybersecurity (WTMC '18). Association for Computing Machinery, New York, USA, pp. 27–32, 2018. https://doi.org/10.1145/3229598.3229605

[3] L. Braun, C. Diekmann, N. Kammenhuber and G. Carle, "Adaptive load-aware sampling for network monitoring on multicore commodity hardware," 2013 IFIP Networking Conference, New York, USA, pp. 1-9, 2013. https://doi.org/10.48550/arXiv.1604.02322

[4] C. Estan, K. Keys, D. Moore, and G. Varghese, "Building a better NetFlow," in Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '04), Association for Computing Machinery, New York, USA, pp. 245–256, 2004. https://doi.org/10.1145/1015467.1015495

[5] P. Barlet-Ros, G. Iannaccone, J. Sanjuàs-Cuxart, D. Amores-López, and J. Solé-Pareta, "Load shedding in network monitoring applications," in Proceedings of the USENIX Annual Technical Conference, ATC'07, Berkeley, USA, pp. 1-14, 2007. https://dl.acm.org/doi/10.5555/1364385.1364390

[6] J. Drobisz and K. J. Christensen, "Adaptive sampling methods to determine network traffic statistics including the Hurst parameter," in Proceedings of 23rd Annual Conference on Local Computer Networks. LCN'98, Lowell, USA, pp. 238-247, 1998. https://doi.org/10.1109/LCN.1998.727664

[7] Z. Jadidi, V. Muthukkumarasamy, E. Sithirasenan and K. Singh, "A probabilistic sampling method for efficient flow-based analysis," Journal of Communications and Networks, vol. 18, no. 5, pp. 818-825, 2016. https://doi.org/10.1109/JCN.2016.000110

[8] B. Zeng, D. Zhang, W. Li, M. Zhang and Q. Hong, "An Adaptive Sampling Methodology for Internet Traffic Data Measurement," 2009 International Conference on Communication Software and Networks, Chengdu, China, pp. 215-218, 2009. https://doi.org/10.1109/ICCSN.2009.135

[9] Wenhong Ma, J. Yan and Changcheng Huang, "Adaptive sampling methods for network performance metrics measurement and evaluation in MPLS-based IP networks," in Proceedings of CCECE 2003 - Canadian Conference on Electrical and Computer Engineering, Toward a Caring and Humane Technology, vol. 2, Montreal, Canada, pp. 1005-1008, 2003. https://doi.org/10.1109/CCECE.2003.1226065

[10] J. Zhang, X. Luo, R. Perdisci, G.i Gu, W. Lee, and N. Feamster, "Boosting the scalability of botnet detection using adaptive traffic sampling", in Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS '11). Association for Computing Machinery, New York, USA, pp. 124–134, 2011. https://doi.org/10.1145/1966913.1966930

[11] M. Jin, C. Wang, P. Li and Z. Han, "Survey of Load Balancing Method Based on DPDK," 2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS), pp. 222-224, 2018. https://doi.org/10.1109/BDS/HPSC/IDS18.2018.00054

[12] N. T. Bailey, "On queueing processes with bulk service," Journal of the Royal Statistical Society: Series B (Methodological), vol. 16, no. 1, pp. 80-87, 1954. https://doi.org/10.1111/j.2517-6161.1954.tb00149.x

[13] P. Wiśniewski, M. Sosnowski, W. Burakowski, "On Implementation of Efficient Inline DDoS Detector Based on AATAC Algorithm," International Journal of Electronics and Telecommunications, vol. 68, no. 4, pp. 889-898, 2022. https://doi.org/10.24425/ijet.2022.143899