

Analytical-simulation modeling of a finite queue for asynchronous HTTP request analysis

Alexander Chizhov^{1,*†}, Andrii Fesenko^{2†}, Agnieszka Gajewska^{3†} and Vladyslav Chizhov^{1,†}

¹Scientific Cyber Security Association of Ukraine, Mykhaila Donsia Str., 2A, Kyiv, 03161, Ukraine

²State Scientific and Research Institute of Cybersecurity Technologies and Information Protection, Maksym Zalizniak Str. 3/6, Kyiv, 03142, Ukraine

³University of the National Education Commission, Podchorazych Str., 2, Krakow, 30-084, Poland

Abstract

This article studies an asynchronous pipeline for HTTP event analysis in application-level protection systems. Such systems often use rules, statistical features, or machine learning models to detect attacks. If this analysis is performed directly in the main request path, it can increase latency for legitimate users. Therefore, the article considers a separate queue-based processing pipeline with several parallel workers. The main task is to choose the number of workers and the buffer size so that the system does not lose too many events and can still produce a useful reaction in time. The analytical M/M/c/K queue model is used for the initial estimation of system parameters. A discrete-event simulation model is then used to study bursty traffic, different service-time distributions, and the probability of timely useful processing. The proposed approach can support practical configuration of asynchronous HTTP analysis systems under reaction-time limits.

Keywords

web application security, cybersecurity, WAF, SIEM, DDoS, HTTP request analysis, asynchronous event-driven processing, queue-based architecture, queueing theory, M/M/c/K model, simulation modeling

1. Introduction

Application-level protection of web applications increasingly requires the analysis of HTTP requests using rules, statistical features, and machine learning models. Such analysis may be sufficiently resource-intensive, so its execution directly in the critical path of request processing can increase the latency of legitimate traffic and degrade service quality.

One practical way to mitigate this effect is to move the analysis to a separate asynchronous pipeline. In such an architecture, an HTTP request or an event associated with it is quickly transferred to a queue, while the subsequent analysis is performed by a group of parallel workers outside the main service path [1, 2, 3]. This makes it possible to reduce the impact of the protection logic on legitimate traffic, but it creates a new engineering task: it is necessary to select the queue configuration and the number of workers correctly.

For application protection systems, not only the very fact of subsequent event analysis is important, but also whether a protective reaction can be formed in time. If the conclusion is obtained too late, it may be useful for logging or for further offline analysis, but it will no longer be able to limit the current attack [4, 5]. Therefore, the effectiveness of such an architecture is evaluated not only through the probability of event loss, but also through the fraction of events whose analysis results can still be used for an operational reaction within the given reaction time.

The aim of this work is to build and study a queue model for deferred analysis of HTTP events that makes it possible to justify the choice of the number of workers and the buffer size for a given load,

WCH'26: Fifth International Workshop on Cyber Hygiene, June 25–26, 2026, Krakow, Poland

*Corresponding author.

†These authors contributed equally.

✉ alex.definch@gmail.com (A. Chizhov); andrii.fesenko@npp.kai.edu.ua (A. Fesenko); agnieszka.gajewska@uken.krakow.pl (A. Gajewska); v.chizhov.v@gmail.com (V. Chizhov)

ORCID 0000-0002-3992-8522 (A. Chizhov); 0000-0001-5154-5324 (A. Fesenko); 0000-0002-4620-0222 (A. Gajewska); 0009-0002-6646-904X (V. Chizhov)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

mean analysis time, and allowable reaction time. For this purpose, two approaches are combined: the analytical $M/M/c/K$ model, which provides an initial evaluation of stationary characteristics, and simulation, which makes it possible to take into account the bursty nature of the load, non-exponential processing time, and the metric of the operational usefulness of the analysis result.

2. Analysis of related works

Work [6] directly considers two basic directions of Web Application Firewall construction: signature-based analysis and analysis based on machine learning models. In [7], this approach is presented already at the level of a specific ML-based WAF implementation. For this study, such sources are important because they show the increasing complexity of application-level analysis of HTTP requests: verification increasingly relies not only on simple rules, but also on computationally more complex procedures of feature construction and inference.

Survey [8] provides a broader context for such solutions by generalizing data mining and machine learning methods in intrusion detection tasks. From the viewpoint of our formulation, this is important not because [8] describes WAF queues specifically, but because it shows a general trend: higher detection accuracy is often achieved at the cost of more complex analysis. In [6, 7, 8], however, attention is focused primarily on attack detection methods and classification quality, whereas the question of event residence time in an asynchronous processing system and the choice of this system's parameters is not posed separately.

Work [9] makes concrete why such a time-sensitive formulation is fundamentally important for application-layer DDoS. It is not about exhausting the network channel as such, but about overloading back-end database resources and the associated application logic. For this class of attacks, it is not enough simply to detect malicious activity; the analysis result must be obtained early enough so that it can still affect the current attack.

The architectural prerequisites for moving such analysis to a separate asynchronous pipeline are supported by works [10, 11]. In [10], the transition from data stream processing to complex event processing is considered, where stream intake and subsequent processing are logically separated. In [11], a similar idea can be seen in SIEM systems, where events are collected, aggregated, and analyzed outside their primary source. However, these works do not consider the task of selecting the number of parallel workers and the buffer capacity for HTTP events under an allowable reaction-time constraint.

For the formal description of such a system, it is natural to turn to the queueing-theory apparatus [12, 13, 14]. These sources provide basic results for systems with finite capacity, multi-server service, losses at the input, mean waiting time, and time spent in the system. The $M/M/c/K$ model is suitable as the first analytical level of description of deferred analysis of HTTP events, where c specifies the number of workers and K specifies the total system capacity.

Thus, the available body of work covers three related but distinct dimensions. Works [6, 7, 8, 9] describe the content and complexity of application-level attack detection, [10, 11, 15] justify the feasibility of asynchronous event-driven processing, and [12, 13] provide the mathematical apparatus for queue analysis. In this body of sources, the task of choosing the configuration (c, B) for a finite queue of deferred HTTP analysis with the simultaneous consideration of the loss probability P_{loss} and the probability of timely useful processing P_{useful} for a given reaction time T_{react} is not formulated directly. The present work is aimed precisely at filling this gap.

3. Problem statement

Let us consider an asynchronous analysis pipeline to which events formed on the basis of HTTP requests arrive. The input flow is described by the intensity $\lambda(t)$. Event processing is performed by a group of c parallel workers. If S is a random variable of the analysis time of one event by one worker, then the mean processing intensity of one worker is

$$\mu = \frac{1}{E[S]}$$

In the application context, the analysis time can be represented as the sum of the main components:

$$S = T_{parse} + T_{features} + T_{inference} + T_{decision}$$

where

- T_{parse} - time required for parsing an HTTP packet
- $T_{features}$ - time spent on forming parameters for analysis
- $T_{inference}$ - neural network inference time
- $T_{decision}$ - time for decision making

The system has a finite capacity K , which includes both the events already being processed and the events waiting in the queue. If the queue buffer size is B , then

$$K = c + B$$

The basic drop-on-full policy is considered: if at the moment of arrival there are already K events in the system, the new event is not accepted into the system and is considered lost for asynchronous analysis. Alternative displacement or prioritization policies are not studied in this work.

To evaluate practical effectiveness, a reaction-time limit T_{react} is introduced, which specifies the maximum allowable interval from the event arrival to the completion of its analysis. For an individual event e_i , let t_i^{arr} denote the arrival time, t_i^{start} the processing start time, and t_i^{done} the analysis completion time. The total time spent by the event in the system is

$$T_{sys,i} = t_i^{done} - t_i^{arr}.$$

Let A_i be the event that e_i is accepted into the system. Then the event is considered timely useful if it was accepted and its analysis was completed no later than T_{react} :

$$A_i \cap \{T_{sys,i} \leq T_{react}\}.$$

Accordingly, in this work the probability of timely useful processing is defined as

$$P_{useful} = P(A \cap \{T_{sys} \leq T_{react}\}),$$

where discarded events are automatically considered non-useful for operational protection. Similarly, the following are defined:

$$P_{late} = P(A \cap \{T_{sys} > T_{react}\}),$$

$$P_{loss} = P(\bar{A}).$$

For simulation-based evaluation, a decomposition of the total number of events is used:

$$N_{total} = N_{useful} + N_{late} + N_{dropped}$$

where:

- N_{total} - the set of all events that arrived in the system;
- N_{useful} - events processed in a time less than or equal to T_{react} ;
- N_{late} - events whose processing took more time than T_{react} ;
- $N_{dropped}$ - events discarded because of queue overflow.

The empirical estimates of the metrics then take the form

$$\hat{P}_{useful} = \frac{N_{useful}}{N_{total}},$$

$$\hat{P}_{late} = \frac{N_{late}}{N_{total}},$$

$$\hat{P}_{loss} = \frac{N_{dropped}}{N_{total}}.$$

In this formulation, the following balance holds:

$$\hat{P}_{useful} + \hat{P}_{late} + \hat{P}_{loss} = 1.$$

Formally, the task consists in choosing the configuration (c, B) for given $\lambda(t)$, S , and T_{react} under the conditions

$$P_{loss} \leq \epsilon_{loss},$$

$$P_{useful} \geq p_{min}.$$

A practical feature of the problem is that the cost of one additional worker is usually substantially higher than the cost of one additional position in the queue buffer. Therefore, in this work, configuration minimality is interpreted lexicographically: first the number of workers c is minimized, and among configurations with the same c , the buffer size B is minimized. This approach reflects the practical asymmetry between computational resources and buffer memory.

Thus, the task is to evaluate the parameters of a finite queue for asynchronous analysis of HTTP events for application protection systems, taking into account not only losses due to overflow, but also the timeliness of the protective decision.

4. Analytical model

For the initial evaluation of the parameters of the asynchronous pipeline, the $M/M/c/K$ queueing model is used. It describes a system with finite capacity K , to which events arrive with intensity λ , while their processing is performed by c parallel workers with exponential service time [12, 16, 17, 1].

Let

$$a = \frac{\lambda}{\mu}$$

Denote by P_n the stationary probability that there are n events in the system. For states in which the number of events is smaller than the number of workers, we have

$$P_n = P_0 \frac{a^n}{n!}, \quad 0 \leq n < c$$

For states in which all workers are busy and the excess events wait in the queue,

$$P_n = P_0 \frac{a^n}{c! c^{n-c}}, \quad c \leq n \leq K$$

The normalization condition has the form

$$\sum_{n=0}^K P_n = 1,$$

from which

$$P_0 = \left[\sum_{n=0}^{c-1} \frac{a^n}{n!} + \sum_{n=c}^K \frac{a^n}{c! c^{n-c}} \right]^{-1}$$

The probability of event loss due to system saturation is equal to the probability of the full-capacity state:

$$P_{loss} = P_K$$

Since part of the events is discarded at the input, the effective intensity of the event flow accepted by the system is

$$\lambda_{eff} = \lambda(1 - P_{loss})$$

The mean number of events in the queue is

$$L_q = \sum_{n=c}^K (n - c) P_n$$

The mean waiting time of an accepted event in the queue can then be computed by Little's law:

$$W_q = \frac{L_q}{\lambda_{eff}}$$

The mean total time spent by an accepted event in the system is

$$W = W_q + \frac{1}{\mu}$$

In this model, it is also convenient to use the utilization factor

$$\rho = \frac{\lambda}{c \cdot \mu}$$

which shows the relation between the arrival intensity and the total mean productivity of the workers. For a system with finite capacity, the value $\rho \geq 1$ does not mean infinite queue growth, but it indicates a high risk of system saturation and increasing losses.

The analytical $M/M/c/K$ model directly provides the stationary characteristics P_{loss} , λ_{eff} , L_q , W_q , and W [12]. It is not used in this work for the exact analytical computation of P_{useful} , because this metric requires not only the mean residence time, but also the distribution of T_{sys} together with the explicit consideration of discarded events as automatically non-useful. The analytical stage is therefore used for the initial narrowing of the range of allowable values of c and B , while the final evaluation of configuration suitability is performed by simulation.

5. Limitations of the analytical model and the role of simulation

Despite its analytical transparency, the $M/M/c/K$ model has a number of limitations that are important to consider when interpreting the results.

One of the key assumptions of this model is that the input flow is Poisson. For stationary load, this is an acceptable basic approximation, but real attacking HTTP traffic often has a bursty, controlled, or non-stationary nature [18, 19]. Under such conditions, describing the flow only through the mean value λ hides important transient processes.

No less important is the assumption of an exponential service-time distribution. For asynchronous analysis of HTTP events, this is also only an approximation, because the analysis time may be almost deterministic, moderately variable, or, on the contrary, may have a pronounced right tail [13, 20].

In addition, the $M/M/c/K$ model is oriented primarily toward stationary average characteristics. Because of this, it does not fully reflect short-term spikes, buffer filling during an attack, system recovery after overload, and other transient modes that are fundamentally important for application protection.

Another simplification is that all events are considered homogeneous jobs. In real systems, they may differ by source, risk level, type of HTTP operation, or priority. In this work, such a simplification is deliberate, since the goal is precisely the selection of queue-system parameters, rather than the construction of a multi-class traffic model.

Simulation is used as the second stage of the study. It does not replace the analytical model, but makes it possible to verify analytical estimates under conditions closer to the real operation of the system. Within the simulation, it is possible to specify a time-varying flow intensity $\lambda(t)$, model short attack bursts, use different analysis-time distributions S , track the time trajectory of each event, and directly evaluate $\hat{P}_{useful}$, $\hat{P}_{late}$, and $\hat{P}_{loss}$.

The analytical model and simulation perform complementary functions: the former provides a mathematically transparent basis for the initial evaluation of system parameters, while the latter makes it possible to verify their practical suitability from the viewpoint of whether the system manages to form a reaction within the allowable time.

6. Simulation model

A discrete-event simulation model of the asynchronous pipeline for HTTP-event analysis is used [20]. The system state changes only at the moments of a new event arrival, the start of its processing, the completion of analysis, or rejection at admission because of system saturation. This approach makes it possible to record the time trajectory of each event and to determine directly whether the protective decision was formed within the given reaction time.

The simulation model describes a system with c parallel workers, a queue buffer of size B , and total capacity

$$K = c + B.$$

As in the analytical formulation, the drop-on-full policy is used. If at the moment of arrival of a new event there are already K events in the system, such an event is not accepted and is assigned to the set of lost events.

For each accepted event e_i , the moments t_i^{arr} , t_i^{start} , and t_i^{done} are recorded, and the individual analysis time S_i is generated according to the specified distribution. The analysis completion time is computed as

$$t_i^{done} = t_i^{start} + S_i.$$

After that, the total residence time is determined for each completed event:

$$T_{sys,i} = t_i^{done} - t_i^{arr}.$$

An event is classified as useful if $T_{sys,i} \leq T_{react}$ and as late if $T_{sys,i} > T_{react}$. Discarded events are directly placed into the dropped category and are considered non-useful for operational protection.

T_{react} is treated as an externally specified application requirement. If λ_{attack} is the intensity of the attacking flow and N_{bad}^{max} is the maximum allowable number of malicious events before the moment of reaction, then

$$T_{react} = \frac{N_{bad}^{max}}{\lambda_{attack}}.$$

For part of the experiments, $T_{react} = 3$ s was used. For $\lambda_{attack} = 1000$ events/s, this corresponds to a harm budget of $N_{bad}^{max} = 3000$ events. In the final burst scenario, the same reaction time $T_{react} = 3$ s is adopted, but already for $\lambda_{attack} = 1200$ events/s, which corresponds to an increased allowable harm

budget of $N_{bad}^{max} = 3600$ events. This clarification is important for the correct interpretation of the final results.

Within the work, two types of input flow are considered.

Stationary scenario:

$$\lambda(t) = \lambda = const.$$

It is used for comparison with the analytical $M/M/c/K$ model in the case of exponential service time.

Short attack-burst scenario:

$$\lambda(t) = \begin{cases} \lambda_{normal}, & t < t_1, \\ \lambda_{attack}, & t_1 \leq t \leq t_2, \\ \lambda_{normal}, & t > t_2. \end{cases}$$

It is this second scenario that makes it possible to evaluate the behavior of a finite queue under conditions in which the input flow temporarily exceeds the total productivity of the workers.

The event-analysis time S is specified by one of three distributions:

$$S \sim Exp(\mu),$$

$$S = const,$$

$$S \sim Normal^+(m, \sigma).$$

The exponential case is used to validate the simulator against $M/M/c/K$, the deterministic one to model an almost stable analysis time, and the truncated-normal one to model moderate variability while preserving positivity of the processing time.

During each run, the main metrics $\hat{P}_{loss}$, $\hat{P}_{late}$, $\hat{P}_{useful}$ are recorded, as well as diagnostic characteristics: the mean waiting time in the queue, the mean total time spent in the system by accepted events, and the maximum queue length Q_{max} .

The generalized simulation algorithm has the following form:

1. Generate event arrival times according to the selected scenario $\lambda(t)$.
2. For each new event, increase N_{total} .
3. If there are already K events in the system, increase $N_{dropped}$.
4. If a free worker is available, start processing immediately and generate S_i .
5. If all workers are busy but there is space in the queue, add the event to the buffer.
6. After processing completion, pass the next event from the queue to the worker, if it exists.
7. For each completed event, compute $T_{sys,i}$ and classify it as useful or late.
8. After the experiment is completed, compute the final empirical estimates and diagnostic characteristics.

The configuration (c, B) is considered acceptable if the following conditions are satisfied simultaneously:

$$P_{loss} \leq \epsilon_{loss},$$

$$P_{useful} \geq p_{min}.$$

Thus, the simulation model complements the analytical stage and makes it possible to move from mean stationary characteristics to the evaluation of the timeliness of the protective reaction at the level of individual events.

Table 1Comparison of Simulation and Analytical Estimates for the Stationary Flow $\lambda = 900$ Events/s

Configuration	P_{loss}^{sim}	P_{loss}^{theor}	W_q^{sim}, s	W_q^{theor}, s	W^{sim}, s	W^{theor}, s
$c = 20, B = 100$	0,258606	0,259259	0,145448	0,145714	0,175427	0,175714
$c = 20, B = 500$	0,257125	0,259259	0,741446	0,745714	0,771425	0,775714
$c = 20, B = 1000$	0,255274	0,259259	1,479767	1,495714	1,509746	1,525714
$c = 30, B = 100$	0,000000	0,000001	0,004601	0,004713	0,034578	0,034713
$c = 30, B = 500$	0,000000	0,000000	0,004601	0,004714	0,034578	0,034714
$c = 30, B = 1000$	0,000000	0,000000	0,004601	0,004714	0,034578	0,034714
$c = 40, B = 100$	0,000000	0,000000	0,000027	0,000029	0,030005	0,030029
$c = 40, B = 500$	0,000000	0,000000	0,000027	0,000029	0,030005	0,030029
$c = 40, B = 1000$	0,000000	0,000000	0,000027	0,000029	0,030005	0,030029

7. Results of simulation verification and computational experiments

7.1. Validation of the simulator on a stationary flow

At the first stage, the correctness of the simulation implementation was verified by comparison with the analytical $M/M/c/K$ model for a stationary Poisson flow and exponential processing time.

For the light regime with $\lambda = 300$ events/s, $E[S] = 30$ ms, $T_{react} = 3$ s, and configurations $c \in \{20, 30, 40\}$, $B \in \{100, 500, 1000\}$, almost complete agreement was obtained: in all nine configurations, $P_{loss} = 0$, $P_{late} = 0$, $P_{useful} = 1$, and the mean time spent in the system was about 0,03 s.

A denser stationary test with $\lambda = 900$ events/s is more illustrative. Its results are given in Table 1.

The worst absolute errors in this test were

$$\max |\Delta P_{loss}| = 0,003985$$

$$\max |\Delta W_q| = 0,015947 \text{ s}$$

$$\max |\Delta W| = 0,015968 \text{ s}$$

The simulator correctly reproduces the analytical characteristics of $M/M/c/K$ for a stationary Poisson flow and exponential service time and can be used for further simulation experiments.

7.2. Effect of buffer size in an overloaded stationary regime

Separately, the dependence of losses and delay on the buffer size was studied for an overloaded stationary regime with the parameters $c = 20$, $\lambda = 900$ events/s, $E[S] = 30$ ms, $T_{react} = 3$ s, and $B = 50, 100, \dots, 1000$.

The obtained results showed that increasing the buffer in such a regime only weakly reduces losses: P_{loss} changes from 0,258791 at $B = 50$ to 0,255274 at $B = 1000$. The mean waiting time in the queue grows very substantially: from 0,070619 s to 1,479767 s. The probability of timely useful processing also changes only weakly – from 0,741209 to 0,744726.

This result is fundamentally important for the formulation of the optimization task. In the studied range $B = 50..1000$, late events still practically do not appear, but even with a sharp increase in the waiting time, the increase in P_{useful} remains minimal. Increasing the buffer by itself does not provide a substantial gain from the viewpoint of the real usefulness of the system if the computational productivity of the workers is already insufficient.

Table 2

Results of the Final Simulation Experiment

c	B	$\hat{P}_{loss}$	$\hat{P}_{useful}$	Criteria satisfied
20	100	0,184235	0,815765	no
20	500	0,169257	0,830743	no
20	1000	0,150534	0,849466	no
20	2000	0,113087	0,842576	no
20	5000	0,000749	0,647182	no
25	100	0,120202	0,879798	no
25	500	0,105224	0,894776	no
25	1000	0,086501	0,913499	no
25	2000	0,049054	0,950946	no
25	5000	0,000000	0,878937	no
30	100	0,057742	0,942258	no
30	500	0,042764	0,957236	no
30	1000	0,024040	0,975960	no
30	2000	0,000000	1,000000	yes
30	5000	0,000000	1,000000	yes
35	100	0,000037	0,999963	yes
35	500	0,000000	1,000000	yes
35	1000	0,000000	1,000000	yes
35	2000	0,000000	1,000000	yes
35	5000	0,000000	1,000000	yes
40	100	0,000000	1,000000	yes
40	500	0,000000	1,000000	yes
40	1000	0,000000	1,000000	yes
40	2000	0,000000	1,000000	yes
40	5000	0,000000	1,000000	yes

7.3. Final burst experiment and selection of an acceptable configuration

The following parameters were used for the final experiment: $E[S] = 30$ ms, $\mu = 33,3$ events/s, $T_{react} = 3$ s, $\lambda_{normal} = 300$ events/s, $\lambda_{attack} = 1200$ events/s, simulation duration 60 s, attack interval 20–30 s, $c \in \{20, 25, 30, 35, 40\}$, $B \in \{100, 500, 1000, 2000, 5000\}$, $\epsilon_{loss} = 0,01$, $p_{min} = 0,95$.

Taking into account the fixed $T_{react} = 3$ s and $\lambda_{attack} = 1200$ events/s, this scenario corresponds to an allowable harm budget of $N_{bad}^{max} = 3600$ events.

Zero values of $\hat{P}_{loss}$ and the value $\hat{P}_{useful} = 1,000000$ in Table 2 mean that within the corresponding simulation run, neither events discarded because of queue overflow nor late events were observed. All reported simulation values should be interpreted as empirical estimates obtained for a fixed load scenario, a given simulation horizon, and a specific experimental configuration, rather than as universal characteristics of the system.

Since the cost of one additional worker is considered substantially higher than the cost of one additional position in the buffer, a lexicographic minimality criterion is used: first c is minimized, and among configurations with the same c , B is minimized. Under this criterion, the minimally acceptable configuration is

$$c = 30, \quad B = 2000$$

For this configuration, the following was obtained:

$$\hat{P}_{loss} = 0,000000, \quad \hat{P}_{useful} = 1,000000$$

The obtained results show that configurations with $c = 20$ and $c = 25$ do not ensure the simultaneous satisfaction of both criteria even with a substantial increase in the buffer. In these cases, increasing B

Table 3

Comparison of Indicators for Different Analysis-Time Distributions

Distribution S	$E[S]$, ms	$\hat{P}_{loss}$	$\hat{P}_{late}$	$\hat{P}_{useful}$
Exponential	30	0,004692	0,000000	0,995308
Deterministic	30	0,004709	0,000000	0,995291
Normal ⁺	30	0,004982	0,000000	0,995018

does reduce $\hat{P}_{loss}$, but it increases the waiting time and the fraction of events that lose operational value. Starting from $c = 30$, the system can already satisfy both criteria, but only with a sufficiently large buffer. For $c = 35$ and $c = 40$, acceptable results are achieved already at $B = 100$, which confirms the decisive role of the number of workers compared with simply increasing the buffer.

7.4. Comparison of different analysis-time distributions

To check how strongly the results depend on the assumption of exponential service time, a separate stationary experiment was performed with fixed parameters: $\lambda = 1000$ events/s, $T_{react} = 3$ s, $c = 30$, $B = 500$, simulation duration 120 s, and mean analysis time $E[S] = 30$ ms.

Three variants were compared:

$$S_1 \sim Exp(\mu),$$

$$S_2 = 30 \text{ ms},$$

$$S_3 \sim Normal^+(30 \text{ ms}, 5 \text{ ms}).$$

In this experiment, the influence of the shape of the analysis-time distribution on the main practical metrics was insignificant. For all three variants, $P_{late} = 0$, P_{loss} remains close to 0,5%, and P_{useful} exceeds 0,995. This indicates that for this configuration and this load regime, the decisive factor is primarily the mean value $E[S]$, rather than the choice among the distributions Exp , $const$, and $Normal^+$.

8. Conclusions

It is shown that for application protection systems it is not sufficient to evaluate asynchronous analysis only through the probability of losses in the queue. A practically important question is different: whether the system manages to form the analysis result when it can still influence the development of the current attack. Therefore, the queue parameters and the number of workers were considered not only from the viewpoint of throughput, but also from the viewpoint of the operational usefulness of the obtained conclusion.

The analytical $M/M/c/K$ model proved useful as a tool for the initial evaluation of stationary characteristics of the system and for narrowing the space of admissible configurations. This model does not give a direct answer to the question of what fraction of events will be processed within the allowable reaction time. For this reason, a simulation stage is necessary for the metrics P_{useful} and P_{late} , because it takes into account not only mean values, but also the time dynamics of system overload and recovery.

The obtained results have a clear practical interpretation. If the system lacks computational productivity, a simple increase in the buffer does not solve the problem, but only changes its form: part of the events stops being lost at the input, but instead waits longer in the queue and no longer manages to be used for an operational reaction. In other words, the buffer absorbs a short spike well, but it cannot replace missing worker productivity. Therefore, the parameter c is more important than B in

this problem: the number of workers determines the system's ability to actually serve the flow during an attack, whereas the buffer only gives it an additional time reserve.

For the final burst scenario with $\lambda_{normal} = 300$ events/s, $\lambda_{attack} = 1200$ events/s, $E[S] = 30$ ms, and $T_{react} = 3$ s, the minimally acceptable configuration under the criterion adopted in the work was $c = 30$, $B = 2000$. This result is important not only in itself, but also as an engineering conclusion: configurations with $c = 20$ and $c = 25$ do not provide the required indicators even with a substantial increase in the buffer, whereas the transition to $c = 30$ moves the system to a regime in which the buffer already begins to work as a means of smoothing peak load, rather than as an attempt to compensate for the lack of resources. For the design of such pipelines, this means that an error in estimating the required number of workers is substantially more dangerous than an error in choosing a moderate queue size.

The comparison of three analysis-time distributions with the same mean value $E[S]$ showed that in the studied regime, the difference among the exponential, deterministic, and truncated-normal cases is small. This indicates that for loads of this type, the decisive factor is primarily the mean analysis time, not the exact shape of the distribution. Practically, this means that at the stage of the initial engineering calculation, it is more reasonable to focus on reducing the mean processing time and providing a sufficient level of parallelism than on overcomplicating the service-time model.

At the same time, the boundary of applicability of the proposed approach is quite clear. The model is suitable for the preliminary selection of the configuration of an asynchronous pipeline if the events can, in the first approximation, be considered homogeneous, if the queue operates under the drop-on-full policy, and if the load is described by a stationary regime or a simple short-burst scenario. If the system has several event classes with different priorities, complex displacement policies, controlled or adaptive attacks, as well as substantially non-stationary flows, such a model is no longer sufficient, and a transition to multi-class or more complex simulation formulations is required.

Further research should be directed to the following areas:

1. Consideration of heterogeneous event classes with different priorities;
2. Study of alternative overflow-handling policies, in particular drop-head, as a separate direction of future work;
3. Extension of the model to cases of more complex non-stationary or controlled attacking flows;
4. More detailed justification of the parameter T_{react} through application scenarios of damage and the allowable budget of attacking events.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

- [1] M. Zaliskyi, R. Odarchenko, S. Gnatyuk, Y. Petrova, A. Chaplits, Method of traffic monitoring for DDoS attacks detection in e-health systems and networks, in: CEUR Workshop Proceedings, volume 2255, 2018, pp. 193–204.
- [2] M. Vihuro, A. Malyar, G. Litawa, K. Kluczevska-Chmielarz, T. Konrad, P. Migo, Lightweight semantic segmentation for fermentation foam monitoring: A comparative study of u-net, deeplabv3+, fast-scnn, and segnet, Applied Sciences 16 (2026) 1487.
- [3] S. Ivanov, O. Korchenko, G. Litawa, P. Oliinyk, O. Oliinyk, Prototype of hydrochemical regime monitoring system for fish farms, Sensors 26 (2026) 497.
- [4] S. Semenov, V. Tsukur, V. Molokanova, M. Muchacki, G. Litawa, M. Mozhaiev, I. Petrovska, Mathematical model of the software development process with hybrid management elements, Applied Sciences 15 (2025) 11667.

- [5] R. Ziubina, G. Litawa, R. Szklarczyk, P. Biel, O. Veselska, J. Nikodem, Detection of viruses using machine learning method, in: PACIS 2020 Proceedings, 2020.
- [6] S. Applebaum, T. Gaber, A. Ahmed, Signature-based and machine-learning-based web application firewalls: A short survey, *Procedia Computer Science* 189 (2021) 359–367. doi:10.1016/j.procs.2021.05.105.
- [7] M. E. Durmuşkaya, S. Bayraklı, Web application firewall based on machine learning models, *PeerJ Computer Science* 11 (2025) e2975. doi:10.7717/peerj-cs.2975.
- [8] A. L. Buczak, E. Guven, A survey of data mining and machine learning methods for cyber security intrusion detection, *IEEE Communications Surveys & Tutorials* 18 (2016) 1153–1176. doi:10.1109/COMST.2015.2494502.
- [9] D. Beckett, S. Sezer, J. McCanny, New sensing technique for detecting application layer ddos attacks targeting back-end database resources, in: 2017 IEEE International Conference on Communications (ICC), IEEE, 2017, pp. 1–7. doi:10.1109/ICC.2017.7997376.
- [10] G. Cugola, A. Margara, Processing flows of information: From data stream to complex event processing, *ACM Computing Surveys* 44 (2012) Article 15. doi:10.1145/2187671.2187677.
- [11] G. González-Granadillo, S. González-Zarzosa, R. Diaz, Security information and event management (siem): Analysis, trends, and usage in critical infrastructures, *Sensors* 21 (2021) 4759. doi:10.3390/s21144759.
- [12] G. Bolch, S. Greiner, H. de Meer, K. S. Trivedi, *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*, 2 ed., Wiley, 2006. doi:10.1002/0471791571.
- [13] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*, Cambridge University Press, 2013. doi:10.1017/CBO9781139226424.
- [14] I. Ostroumov, et al., A probability estimation of aircraft departures and arrivals delays, in: *Lecture Notes in Computer Science, including Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*, volume 12950, 2021, pp. 363–377. doi:10.1007/978-3-030-86960-1_26.
- [15] M. Aleksander, G. Litawa, V. Karpinskyi, Distributed computing system which solves an elliptic curve discrete logarithm problem, in: *Proceedings of the 10th International Conference “The Experience of Designing and Application of CAD Systems in Microelectronics”*, 2009, pp. 378–380.
- [16] S. Smerichevska, L. Ivanenko, Z. Poberezhna, S. Smerichevskyi, T. Konrad, D. Dudko, Matrix-vector approach to assessing the competitiveness of logistics enterprises in the context of economic security and information risks, in: *CEUR Workshop Proceedings*, volume 4024, 2025, pp. 440–451.
- [17] J. S. Al-Azzeh, M. Al Hadidi, R. S. Odarchenko, S. Gnatyuk, Z. Shevchuk, Z. Hu, Analysis of self-similar traffic models in computer networks, *International Review on Modelling and Simulations* 10 (2017) 328–336. doi:10.15866/iremos.v10i5.12009.
- [18] S. Kandula, D. Katabi, M. Jacob, A. W. Berger, Botz-4-sale: Surviving organized ddos attacks that mimic flash crowds, in: *Proceedings of the 2nd Symposium on Networked Systems Design and Implementation (NSDI '05)*, USENIX Association, 2005.
- [19] S. Ranjan, R. Swaminathan, M. Uysal, A. Nucci, E. W. Knightly, Ddos-shield: Ddos-resilient scheduling to counter application layer attacks, *IEEE/ACM Transactions on Networking* 17 (2009) 26–39. doi:10.1109/TNET.2008.92503.
- [20] A. M. Law, *Simulation Modeling and Analysis*, 5 ed., McGraw-Hill Education, 2015.