УДК 621.391

М.М. Климаш, М.І. Кирик, Н.М. Плесканка, І.О. Кагало
Національний університет “Львівська політехніка”

БАГАТОРІВНЕВА МОДЕЛЬ БУФЕРИЗАЦІЇ ДАНИХ У ВУЗЛАХ ОБСЛУГОВУВАННЯ МУЛЬТИСЕРВІСНОГО ТРАФІКУ

Проведено огляд основних принципів буферизації даних у вузлах обслуговування мультисервісного трафіку. Запропоновано багаторівневу модель буферизації даних у вузлах, які обслуговують велику кількість TCP потоків. Здійснено оптимізацію моделі буферизації з метою забезпечення задовільної якості надання послуг у мультисервісній мережі передачі даних.

Ключові слова: буферизація даних, синхронний потік, ковзне вікно, смарт-буфер, алгоритм обслуговування черг, багаторівнева модель.

M. M. Klymash, M. I. Kyryk, N. M. Pleskanka, I. O. Kagalo
Lviv Polytechnic National University

DATA BUFFERING MULTILEVEL MODEL AT THE MULTISERVICE TRAFFIC SERVICE NODE

Given the bursty behavior of cloud applications described above, an easy solution to the incast problem would be to overdesign buffer capacity at each network node. The basic principles of data buffering at the multiservice traffic service node were reviewed. The data buffering multilevel mode in the nodes, that serve a large number of TCP flows, was proposed. Each level of the model has its own characteristics and is relatively independent. However the malfunction of any of them, may adversely excel at the efficiency of other levels. The network interface represents the physical level, which is the lowest. Packets routing are occurs at the protocol level. Two queues provide communication between physical layer (network interface card) and ip module. One is called the backlog and is used for incoming packets, the other – txqueue, for outgoing packets. In the current networks, the basis for data transmission is TCP/IP protocol stack. It provides a set of tools to deliver data from one application to another. Today, the size of the buffers is determined by the dynamics of TCP's congestion control algorithm. In particular, the goal is to make sure that when a link is congested, it is busy 100% of the time; which is equivalent to making sure its buffer never goes empty. A widely used rule-of-thumb states that each link needs a buffer of size B = RTT × C, where RTT is the average round-trip time of a flow passing across the link, and C is the data rate of the link. Arguably, router buffers are the single biggest contributor to uncertainty in the Internet. Buffers cause queueing delay and delay-variance; when they overflow they cause packet loss, and when they underflow they can degrade throughput. Given the significance of their role, we might reasonably expect the dynamics and sizing of router buffers to be well understood, based on a well-grounded theory, and supported by extensive simulation and experimentation. The Smart-Buffer architecture takes into consideration that congestion in a typical data center environment is localized to a subset of egress ports at any given point in time and realistically never happens on all ports simultaneously. This enables its centralized on-chip buffer to be right-sized for overall cost and power; at the same time, the buffer is dynamically shareable and weighted towards congested ports or flows exactly when needed using self-tuning

thresholds. In addition, the centralized buffer can be allocated based on class of service or priority group. Available buffer resources can therefore be partitioned into separate, virtual buffer pools and assigned to special traffic classes. This is especially useful in converged I/O scenarios where some traffic classes (such as storage) may require guaranteed lossless behavior. These properties enable Smart-Buffer technology to strike an optimum balance between silicon efficiency and burst absorption performance – essential design principles in current and next-generation high density. Optimization of the data buffering multilevel model to ensure a satisfactory quality of service in multiservice data network was made in this paper.

Key words: data buffering, synchronous flow, scale window, smart buffer, queuing algorithm, multilevel model.

Вступ. За сучасного рівня розвитку телекомунікаційних мереж інтенсивність мережевого трафіку зростає із кожним днем. Це викликає необхідність у виборі плоскіших мережевих архітектур, оптимізованих для меншої кількості сплесків, менших затримок та з'єднань типу "кожен з кожним". Такі зміни роблять першочерговим питання управління мережевим трафіком та перевантаженням, оскільки традиційні методи, такі як збільшення розмірів буферів у вузлах мультисервісної мережі, потенційно можуть негативно впливати на ключові IT вимоги, такі як продуктивність, затримка і час відповіді сервісів, а також потребувати додаткових витрат.

Проте не слід забувати і те, що все це вимагає відповідних мережевих ресурсів, оскільки, відповідно до вимог часу, сучасні мережі передачі даних повинні будуватися як високонадійні системи, які здатні забезпечити задані показники якості обслуговування (Quality of Service, QoS). Сьогодні вже існує досить велика кількість механізмів обробки трафіку, які сприяють забезпеченню QoS. Одним із основних та найбільш важливих і критичних є буферний ресурс та ресурс управління чергами, а також механізми управління доступом та перевантаженням.

Черги є тим інструментом управління перевантаженням, який допомагає у випадку, коли мережевий пристрій не встигає обробляти пакети та передавати їх на вихід в такому темпі, в якому вони надходять на вхід. Буферизація пакетів під час перевантажень являє собою основний механізм підтримки якості обслуговування пульсуючого трафіку, що забезпечує високу продуктивність роботи мереж передачі даних. Водночас організація черг вноситиме невизначену затримку під час передачі пакетів через мережу, а в деяких випадках і втрату пакетів через переповнення буферів, відведених під організацію черг.

Буферизація даних, безперечно, потрібна, оскільки саме завдяки їй можуть бути поглинені короткочасні сплески активності мережевого трафіку, що пов'язано із властивістю самоподібності мережевого трафіку [1]. Щоб запобігти нескінченному зростанню довжини черги, потрібно відкидати частину пакетів. Які саме пакети будуть відкинуті, вирішуватиме механізм обробки черг. Саме тут ми стикаємось із проблемою вибору оптимального розміру буфера; занадто малий розмір призведе до втрати пакетів, завелика черга буде вносити значну затримку, що недопустимо для сервісів реального часу. Відповідь на це складне питання намагався дати Клейнрок, показавши, що розмір буфера визначатиметься як [2]:

$$N = B \cdot T_{пер} \cdot n_n, \tag{1}$$

де N – розмір буфера; B – пропускна спроможність; $T_{пер}$ – затримка передачі; n_n – кількість потоків.

Останні дослідження показали, що факт оптимальності розміру мережевого буфера, який запропонував Клейнрок, швидше за все, дає можливість визначати верхню межу довжини черги. Багато сучасних маршрутизаторів можуть краще працювати зі значно меншою кількістю буферизованих пакетів.

Модель буферизації даних. Принцип буферизації даних у будь-якому мережевому пристрої досить зручно представляти у вигляді кількох рівнів. В нашому випадку це буде модель, зображена на рис. 1. Кожен з поданих нижче рівнів має свої характеристики і є порівняно незалежним. Однак неправильна робота будь-якого з них може негативно позначитись на працездатності інших рівнів та роботі усього пристрою.

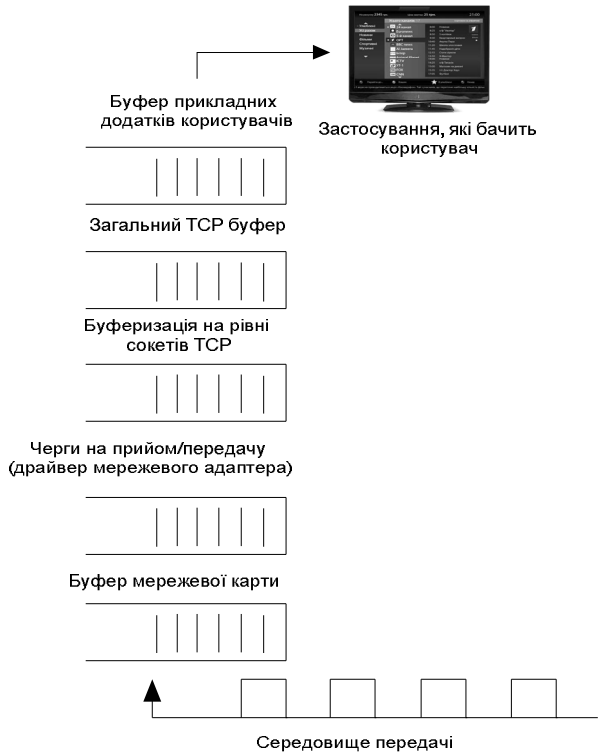

Рис. 1. Модель буферизації даних

Дані, які приходять із середовища передавання, потрапляють безпосередньо на мережевий адаптер обслуговуючого пристрою. Кожен адаптер має два типи черг, а саме:

а) черга на прийом (розміщуються пакети, які надходять із мережі);

б) черга на передавання (розміщуються пакети, які передаються в мережу).

Наступним є рівень черг, на якому відбувається розміщення пакетів у черзі на прийом та передачу, а також визначається механізм формування та опрацювання черг. Цей рівень виконує транспортну роль, доставляючи пакети зі стека протоколів TCP/IP у фізичну частину і навпаки. І хоча налаштування тут тривіальні, проте, передаючи дані на великі відстані, варто звернути увагу на цей рівень.

Третій та четвертий рівні можна об'єднати під рівнем протоколів, в якому розміщується стек TCP/IP. Саме тут відбувається обробка всіх TCP/IP пакетів. Буфер третього рівня виділяється на кожне TCP з'єднання. На четвертому рівні виділяється загальний буфер. І хоча в стеку TCP/IP транспортну роль виконують два протоколи, TCP і UDP, у цій статті описується робота лише TCP протоколу. Зазвичай настроювання TCP зводиться до того, щоб задати правильні значення буферів, неправильний розмір яких може стати причиною вичерпання ресурсів на вузлі або, що менш критично, низької продуктивності протоколу.

Останній рівень – це рівень прикладних сервісів, які використовує кінцевий користувач. Саме тут користувач може побачити результати роботи всіх рівнів та дати оцінку роботи сервісу. Розмір буфера на цьому рівні буде визначатись програмами, якими користується користувач, такими як VLC медіа-плеєр (для перегляду медіа-даних у режимі реального часу), браузер (для перегляду WEB сторінок) тощо.

Принцип розподілу буферного простору. Розглядаючи механізм буферизації даних у вузлах обслуговування мережевого трафіку, необхідно згадати про два аспекти [3–4]:

– принцип розподілу буферного простору;

– оптимальний розмір буфера.

Обчислюючи розмір буферів, варто керуватися такими параметрами:

1. Середня кількість одночасних TCP підключень.

2. Відстань до віддаленої точки (RTT – час проходження пакета).

3. Пропускна здатність каналу.

Багато досліджень та експериментів показують, що розмір буфера визначатиметься як:

$$B = C \cdot RTT, \qquad (2)$$

де B – розмір буфера для обслуговуючого пристрою, C – пропускна здатність каналу зв'язку, RTT – середній час проходження пакета туди і назад. Однак дослідження [5, 6] показують, що такий метод може дати правильний результат у випадку одного потоку із великою інтенсивністю. Проте зазвичай більшість вузлів мультисервісної мережі (комутаторів і маршрутизаторів) одночасно обслуговують велику кількість таких потоків. Сьогодні більшість трафіку використовує протокол TCP, тому кількість потоків, що передаються каналами зв'язку, зростає. Магістральні канали зазвичай працюють на швидкостях 2,5 Гбіт/с або 10 Гбіт/с та передають тисячі потоків [7].

Дослідження показують, що в таких умовах, коли є n потоків, розмір буфера повинен бути не більшим, аніж:

$$B = (C \cdot RTT) / \sqrt{n}. \quad (3)$$

Важливим аспектом загального буферного пулу є можливість встановлювати пороги, що динамічно адаптуються для кожної черги. Це означає, що поріг, після якого трафік починає відкидатися, не вказується статично для кожної черги [8]. Цей поріг динамічно адаптується під умови затору (черги). Динамічне налаштування порогів для кожної черги є ключовим досягненням технології буферизації і дозволяє досягти не тільки хорошого згладжування сплесків трафіку, а й забезпечити рівноправність між чергами [9]. Дослідження показали, що для досягнення еквівалентної здатності обробки сплесків трафіку і рівня втрати кадрів пристрої зі статичним розподілом пам'яті по портах вимагають буфер, майже в п'ять разів більший, ніж пристрої з динамічним загальним буферним пулом та з адаптивними порогами [10].

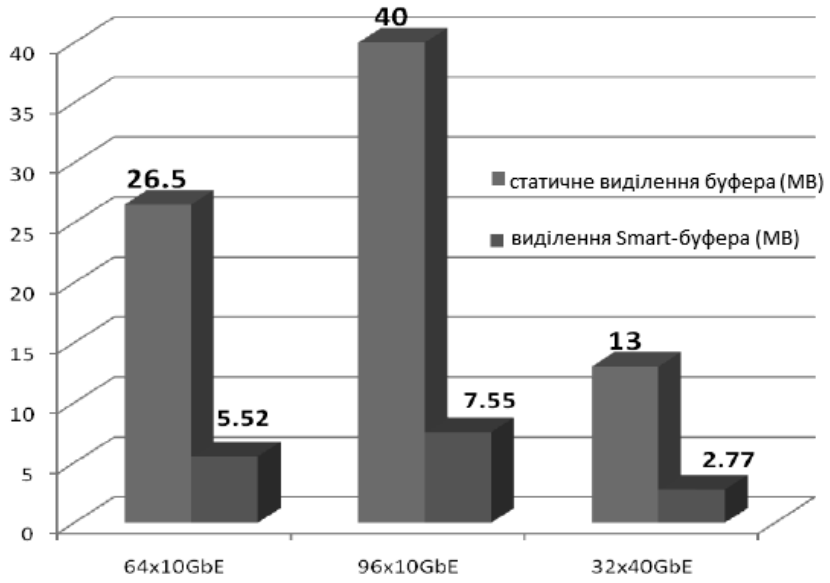

Рис. 2. Порівняння статичної та динамічної (адаптивної) буферизації

Ключовим аспектом технології смарт-буферизації є те, що пороги відкидання пакетів вибираються динамічно на основі умов затору (черги) або, інакше кажучи, на основі розміру буферного пулу, що залишився. Саме завдяки динамічній адаптації під умови затору довші й переповнені черги не можуть заповнити весь буфер, залишивши коротші та вільні черги ні з чим. Така адаптація порога відкидання пакетів називається Fair Adaptive Dynamic Threshold (FADT) [11]. Технологія смарт-буферизації на базі FADT може дати такі переваги:

– *Ефективнішу обробку сплесків трафіку* – це дає гарантію того, що за короткочасного перевантаження алгоритм управління буфером забезпечить достатню кількість ресурсів для збереження пакетів у черзі, зменшивши кількість втрачених кадрів.

– *Рівноправний доступ до спільного буферного пулу* – гарантує, що під час затору порти, які не є перевантаженими, не залишаться без доступу до буферного ресурсу.

– *Незалежну від трафіку продуктивність*, оскільки умови затору з часом змінюються динамічно, залежно від навантаження, вибирати параметри для політики управління буфером важливо так, щоб вони не залежали істотно від природи трафіку.

Буферизація у вузлах із великою кількістю TCP потоків. У вузлах обслуговування магістральних мереж одночасно проводиться обробка великої кількості потоків, тому модель із одним довготривалим потоком є практично нереальною. Для прикладу, канал 2,5 Гбіт/с зазвичай передає більше ніж 10000 потоків у один момент [12]. В роботі розглянемо дві ситуації. По-перше, буде розглянуто випадок, коли всі потоки синхронізовані один з одним. В другому випадку

розглядатимуться потоки, які не синхронізовані один з одним або синхронізовані недостатньо, щоб передаватись абсолютно синхронно.

Синхронізовані потоки

Розглянемо процес передавання двох TCP потоків через вихідний інтерфейс обслуговуючого пристрою. Діаграму зміни розмірів вікна та довжини черги показано на рис. 3.

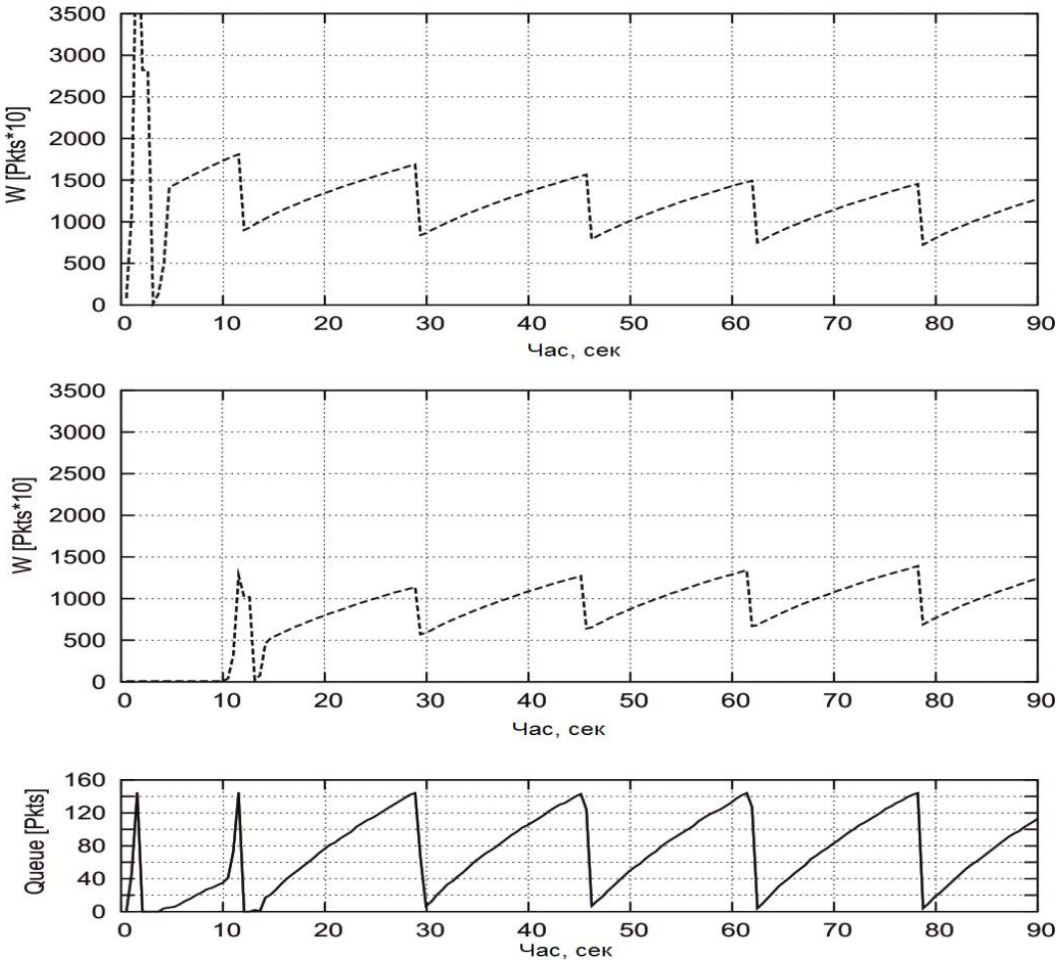

Рис. 3. Діаграма зміни розмірів вікна та довжини черги

Два верхні графіки показують, як змінюється розмір вікна відправника, нижній показує зміну довжини черги обслуговуючого пристрою. Хоча два потоки надходять на обслуговуючий пристрій в різні моменти часу, вони швидко синхронізуються і починають передаватись абсолютно в одній фазі. Ця тенденція синхронізації потоків добре відома та досліджена [13–15].

Набір абсолютно синхронних потоків має такі самі вимоги до розміру буфера, як і єдиний потік. Їх сумарна поведінка описана пилкоподібною залежністю; висота визначається максимальним розміром вікна, необхідного для заповнення шляху туди і назад, що не залежить від кількості потоків. Зокрема, припустимо, що є n потоків, кожен з яких має розмір вікна $W_i(t)$ в момент часу t, а час передачі від відправника до одержувача дорівнює T_i де $i=[1,...,n]$. Розмір ковзного вікна – це максимально допустима кількість переданих байтів, отже:

$$\sum_{i=1}^{n} W_i(t) = 2 \cdot T_s \cdot C + F(t), \qquad (4)$$

де $F(t)$ – наповненість буфера в момент часу t, а T_s – середня затримка передачі.

Для визначення розміру буфера можна розглянути два випадки:

- безпосередньо перед початком передачі;
- відразу після того, як відкидається перший пакет.

Оскільки потоки передаються синхронно, всі потоки матимуть найбільший розмір вікна, $W_{\max}$ в той самий момент часу – коли буфер буде повний:

$$\sum_{i=1}^{n} W_i(t) = W_{\max} = 2 \cdot T_s \cdot C + B \quad (5)$$

Аналогічно, розмір вікна буде найменшим в момент, коли вони одночасно почнуть відкидати пакети. Якщо розміри буфера такі, що після того, як відправник відновив передачу після стану паузи, він пустий, тоді розмір ковзного вікна:

$$\sum_{i=1}^{n} W_{i(\min)} = \frac{W_{\max}}{2} = 2 \cdot T_s \cdot C \quad (6)$$

Отже, можна стверджувати, що розмір буфера визначатиметься як:

$$B = 2 \cdot T_S \cdot C = RTT \cdot C \quad (7)$$

Очевидно, що це твердження справедливе для будь-якої кількості синхронних потоків.

Розсинхронізовані потоки

Несинхронізовані потоки, які передаються через вихідний інтерфейс обслуговуючого пристрою, містять тисячі потоків із різними значеннями RTT. Незначні зміни RTT або часу обробки є достатніми для запобігання синхронізації [16]; відсутність синхронізації потоків продемонстровано в реальних мережах [17, 18].

Щоб зрозуміти різницю між процесом вибору розміру вікна у разі використання синхронізованих та розсинхронізованих потоків, зазначимо, що якщо додати разом багато синхронізованих потоків, ми отримуємо єдиний сумарний потік, який вимагатиме незмінного розміру буфера. З іншого боку, якщо потоки несинхронні, то чим більшою буде їхня кількість, тим меншою стане їх сума. За графічного представлення як пилкоподібного процесу вони будуть згладжувати один одного і відстань від піку до мінімального значення, яка і визначатиме розмір вікна, зменшуватиметься. Отже, враховуючи, що розмір буфера визначатиметься як відстань від піку до западини сукупного розміру вікна, можна очікувати, що розмір буфера має бути меншим у разі збільшення кількості потоків.

Розглянемо набір TCP потоків з випадковими (і незалежними) часом початку і затримкою передавання. Вважатимемо, що вони достатньо розсинхронізовані та процеси вибору розміру вікна незалежні один від одного. Ми можемо моделювати загальний розмір вікна як обмежений випадковий процес, що складається із суми цих незалежних пилкоподібних потоків. Процес агрегації розміру ковзного вікна буде сходитись до гауссівського процесу. Рис. 4 показує, що сукупний розмір вікна насправді сходиться до гауссівського процесу. Графік показує розподіл імовірностей сум ковзних вікон всіх потоків $W = \sum W_i$ із різними значеннями часу передавання та старту. Значення довжини черги в момент часу t визначатиметься як:

$$F(t) = \sum_{i=1}^{n} W_i(t) - (2 \cdot T_s \cdot C) - a \ . \quad (8)$$

Із цієї формули очевидно, що всі надіслані пакети або містяться в буфері обслуговуючого пристрою ($F(t)$), або в мережі $(2 \cdot T_s \cdot C)$, або ж були втраченими. Кількість втрачених пакетів представлено як a . Якщо розмір буфера достатній та протокол TCP працює належно, тоді кількість втрачених пакетів дуже незначна порівняно із $(2 \cdot T_s \cdot C)$, тому довжину черги можна представити як:

$$F = W - 2 \cdot T_s \cdot C \, . \quad (9)$$

Оскільки W характеризується нормальним законом розподілу, F має теж нормальний закон розподілу, зміщений на константу (звичайно, нормальний розподіл обмежений діапазоном, допустимих для F значень). Отже, тепер можна вибрати розмір буфера і знати ймовірність того, що розмір буфера замалий, внаслідок чого втрачатиметься пропускна здатність каналу.

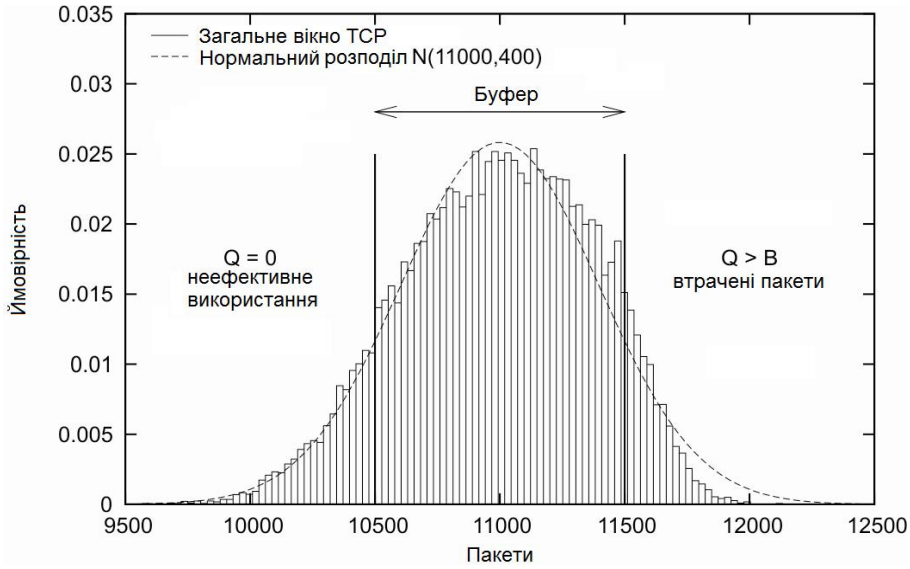

Рис. 4. Розподіл ймовірностей сум ковзних вікон розсинхронізованих потоків

Оскільки це розподіл Гаусса, то можна визначити процес заповнення черги, знаючи середнє значення та дисперсію. Середнє значення є просто сумою середніх значень всіх складових. Для обчислення дисперсії зробимо припущення, що всі потоки мають однакове середнє значення. Кожна зміна TCP вікна може бути змодельована як коливання із нормальним законом відносно середнього розміру ковзного вікна W_i із відповідними мінімальним $\frac{2}{3}W_i$ та максимальним $\frac{4}{3}W_i$ значеннями. Оскільки стандартне відхилення для рівномірного розподілу дорівнює $\frac{1}{\sqrt{12}}$ розмаху, то стандартне відхилення для одного розміру вікна становить:

$$s_i = \frac{1}{\sqrt{12}} \cdot \left(\frac{4}{3}W_i - \frac{2}{3}W_i \right) = \frac{1}{3\sqrt{3}}W_i . \tag{10}$$

Із формули (9) можна отримати:

$$W_i = \frac{W}{n} = \frac{2 \cdot T_s \cdot C + F}{n} \le \frac{2 \cdot T_s \cdot C + B}{n} . \tag{11}$$

Для великої кількості потоків стандартне відхилення суми ковзних вікон W визначається як:

$$s_w \le \sqrt{n} \cdot s_{w_i} . \tag{12}$$

Отже, із виразів (11) та (12), отримаємо:

$$s_w \le \frac{1}{3\sqrt{3}} \cdot \frac{2 \cdot T_s \cdot C + B}{\sqrt{n}} . \tag{13}$$

Знаючи закон розподілу черги, можна знайти, наскільки продуктивно використовується канал передачі для заданого розміру буфера. Коли розмір черги стає меншим від порогового значення, є велика імовірність того, що черга буде пустою і канал передачі простоюватиме. Якщо знатимемо імовірність того, що розмір черги є меншим від порогового значення, тоді отримаємо нижню границю, що свідчитиме про неефективне використання пропускної спроможності каналу передавання. Оскільки закон розподілу черги є нормальним, можна скористатись функцією помилок для обчислення цього значення [19]. Так можна отримати нижнє граничне значення:

$$Effec \ge erf\left(\frac{3\sqrt{3}}{2\sqrt{2}} \cdot \frac{B}{\frac{2 \cdot T_S \cdot C + B}{\sqrt{n}}} \right) . \tag{14}$$

Моделювання та результати експериментів. У попередніх розділах коротко описано теоретичні моделі, які можна використати для опису TCP потоків. В цьому розділі представлено результати, отримані на основі імітаційного моделювання.

Схема, згідно з якою проведено моделювання, зображена на рис. 5.

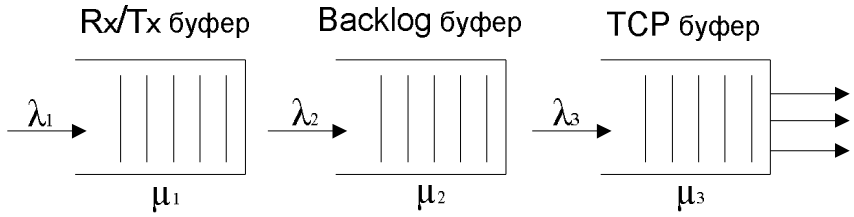

Рис. 5. Схема буферизації даних у стеку операційної системи вузла обслуговування

Дані, які надходитимуть із мережі, спочатку попадатимуть на мережевий інтерфейс обслуговуючого пристрою, на якому і відбуватиметься перший етап буферизації. Вони будуть записуватись у буферну пам'ять мережевого інтерфейсу. В ролі алгоритму обслуговування черги в цьому випадку буде розглядатись FIFO. Інтенсивність надходження та обслуговування пакетів визначатиметься інтенсивністю надходження даних із мережі та параметром fps (frame per second) мережевого інтерфейсу [20].

Після того, як пакет був отриманий мережевою картою, він надходить на обробку в ядро. Перед тим, як перейти на рівень ядра (TCP), пакет поміщається в Backlog чергу. Інтенсивність, з якою пакети надходитимуть в цю чергу, визначатиметься продуктивністю роботи мережевого адаптера обслуговуючого пристрою, оскільки саме він формуватиме вхідний потік для черг цього рівня. FIFO є алгоритмом, який використовується для обробки backlog черги.

Останній етап буферизації, не враховуючи буфери прикладних програм, які використовуються кінцевим користувачем у кінцевих пристроях, є буфер ядра (TCP). На цьому етапі використовуються два типи буферизації даних, а саме:

– буферизація даних для кожного з'єднання;

– загальний буфер для всіх потоків.

Інтенсивність надходження пакетів на цьому рівні визначатиметься інтенсивністю їх обслуговування на попередніх двох рівнях. Алгоритм обслуговування черг тут може бути довільним, а саме FIFO, PQ, FQ, WEQ. Тактова частота процесора обслуговуючого пристрою визначатиме інтенсивність обслуговування пакетів на цьому рівні. Перед тим, як передаватись на обслуговування, дані будуть зберігатись у буферній пам'яті.

Розроблена імітаційна модель дасть змогу визначити вплив параметрів буферизації на кожному із описаних вище рівнів на якість передавання та обслуговування мультисервісного трафіку в телекомунікаційних мережах передачі даних.

Результати моделювання наведено на рис. 6.

Рис. 6 показує, як змінюється рівень заповнення буфера на кожному рівні залежно від інтенсивності вхідного потоку. Найшвидше починає заповнюватись буфер на третьому рівні представленої моделі. Це свідчить про те, що інтенсивність надходження пакетів із нижчих рівнів вища за інтенсивність обслуговування на цьому рівні. Це також можна пояснити і тим, що продуктивність роботи обслуговуючого пристрою на третьому рівні може паралельно розподілятись між кількома процесами. Оскільки значення інтенсивностей обслуговування (в цьому експерименті) на всіх рівнях буферизації є сталими, то можна зрозуміти, що за певного значення інтенсивності надходження пакетів, яке перевишуватиме задану інтенсивність обслуговування, пакети почнуть буферизуватись. Якщо рівень втрат різко зростатиме після досягнення інтенсивності надходження пакетів певного рівня, це означатиме, що слід збільшити

розмір буфера, якщо це можливо, або ж підвищити продуктивність роботи вузла обслуговування. Рівень втрат представлено на рис. 7.

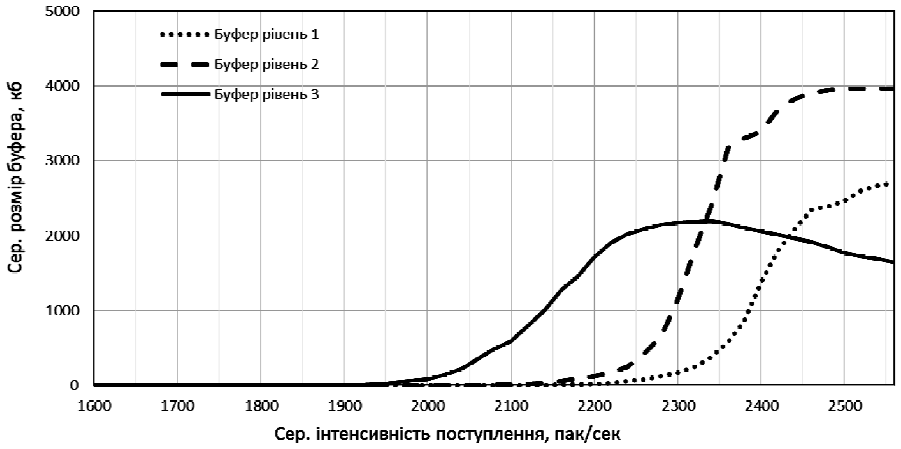

Рис. 6. Залежність розміру буфера від інтенсивності надходження пакетів (інтенсивність обслуговування на кожному рівні $m_1 = 2500, m_2 = 2400, m_3 = 2300$ *; розмір буфера на кожному рівні* $B_1 = 3000, B_2 = 4500, B_3 = 3100$ *)*

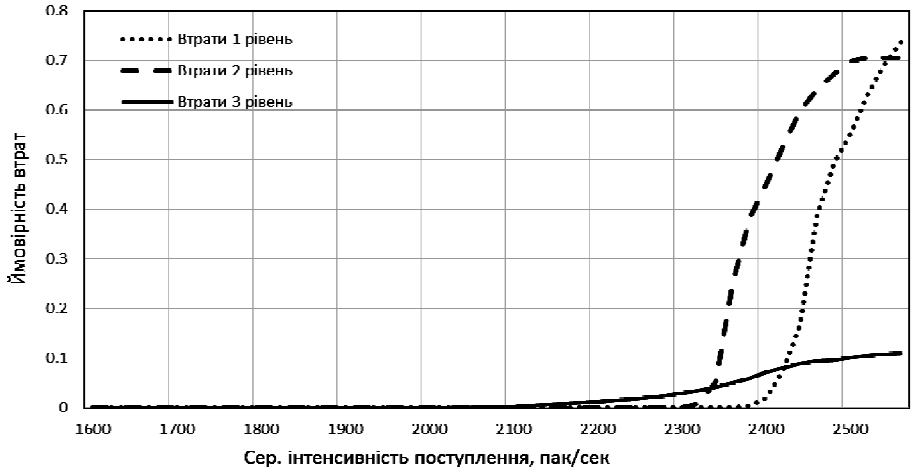

Рис. 7. Залежність ймовірності втрат від інтенсивності надходження пакетів

Рівень втрат на другому та першому рівнях є досить значним за достатньо великої інтенсивності надходження пакетів на вузол обслуговування. Оскільки дамп вхідного потоку, для якого проводився цей експеримент, містить дані реального часу, то це свідчить про те, що такий значний рівень втрат недопустимий для цього типу сервісу. Збільшення розміру буфера дасть змогу понизити рівень втрати пакетів. Однак слід зауважити і той факт, що за більшого розміру буфера зросте час затримки пакетів під час обробки у вузлі обслуговування, що є не надто сприйнятливим для даних реального часу. Іншим способом вирішення цієї проблеми є збільшення продуктивності роботи обслуговуючого пристрою або ж використання технології адаптивної буферизації даних. Результати моделювання для збільшеного розміру буфера подано на рис 8. Оскільки найзначніші втрати на перших двох рівнях, то розмір буфера було збільшено саме там.

На рис. 8 показано, як змінюється завантаженість буфера у разі збільшення інтенсивності надходження пакетів на вузол обслуговування. Відповідно до цих значень розміру буферів рівень втрат зображено на рис. 9.

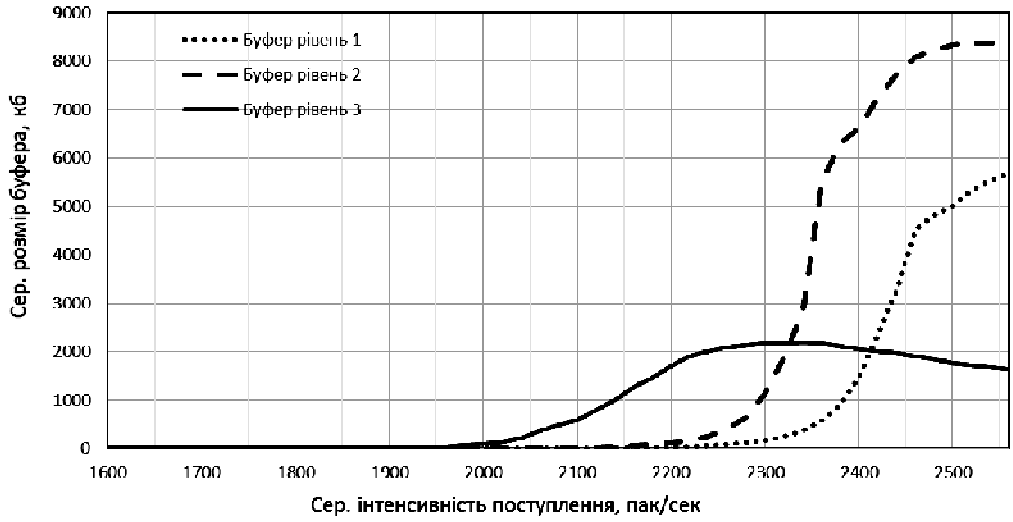

Рис. 8. Залежність розміру буфера від інтенсивності надходження пакетів (інтенсивність обслуговування на кожному рівні $m_1 = 2500, m_2 = 2400, m_3 = 2300$ *; розмір буфера на кожному рівні* $B_1 = 6000, B_2 = 9000, B_3 = 3100$ *)*

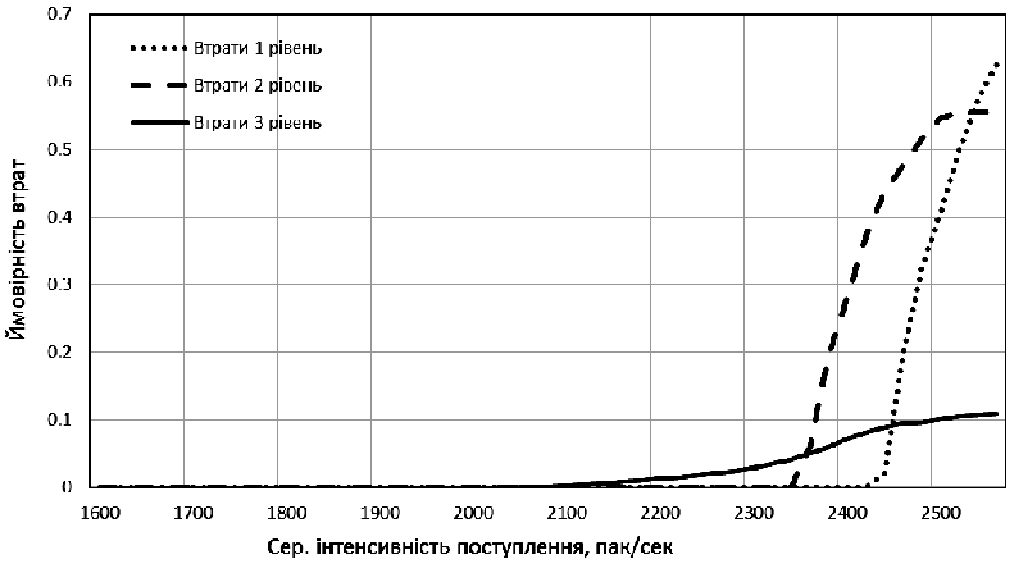

Рис. 9. Залежність ймовірності втрат від інтенсивності надходження пакетів

Як бачимо із рис. 9, рівень втрат дещо знизився порівняно із попередніми результатами, проте не є задовільним для якісних сервісів у телекомунікаційній мережі передачі даних. Оскільки подальше збільшення розміру буфера буде недоцільним, можна зробити висновок, що варто збільшити інтенсивність обслуговування вузла для забезпечення задовільної якості надання послуг. Результати моделювання за збільшених значень інтенсивності обслуговування та розміру буферного простору зображено на рис. 10, 11.

Як можна побачити, розмір буфера та інтенсивність обслуговування на третьому рівні достатні для забезпечення надання задовільних сервісів. На рис. 10 розмір буфера на двох нижніх рівнях є теж задовільним, оскільки дає змогу забезпечити допустимий рівень втрат. Результати залежності втрат подано нижче.

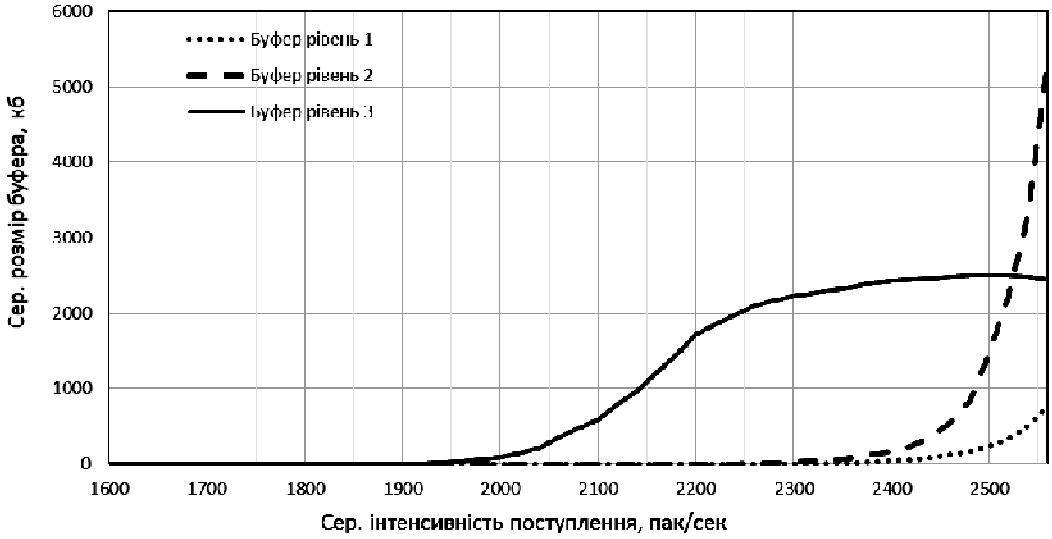

Рис. 10. Залежність розміру буфера від інтенсивності надходження пакетів (інтенсивність обслуговування на кожному рівні $m_1 = 2700, m_2 = 2600, m_3 = 2300$ *; розмір буфера на кожному рівні* $B_1 = 6000, B_2 = 9000, B_3 = 3100$ *)*

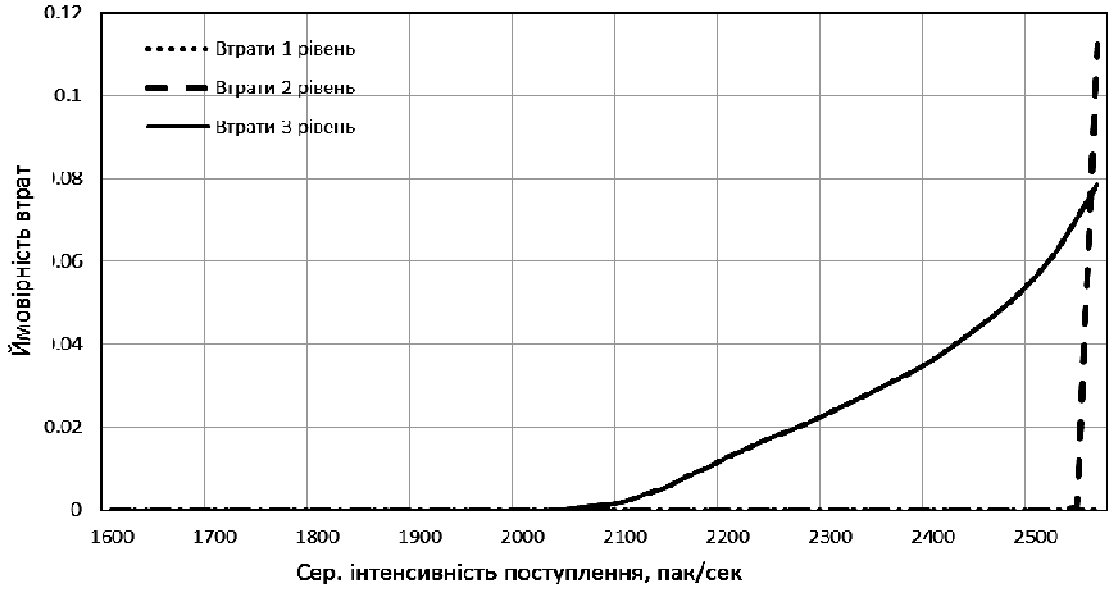

Рис. 11. Залежність ймовірності втрат від інтенсивності надходження пакетів

Як бачимо із представлених результатів, втрати на другому значно перевищують втрати першого та третього рівнів. Це свідчить про те, що інтенсивність надходження пакетів на першому рівні перевищує інтенсивність другого рівня. Через це пакети не встигають опрацьовуватись і поміщаються в буферний простір. В момент, коли рівень заповнення буфера досягає порогового значення, пакети починають втрачатись. Для боротьби з цим явищем можна ще збільшити інтенсивність обслуговування на другому рівні або ж запропонувати використання технології смарт-буферизації даних.

Використання технології динамічної зміни буферного простору залежно від рівня втрат дає можливість раціональніше використовувати буферний простір та мінімізувати втрати. Результати моделювання із використанням технології смарт-буферизації наведено на рис. 12.

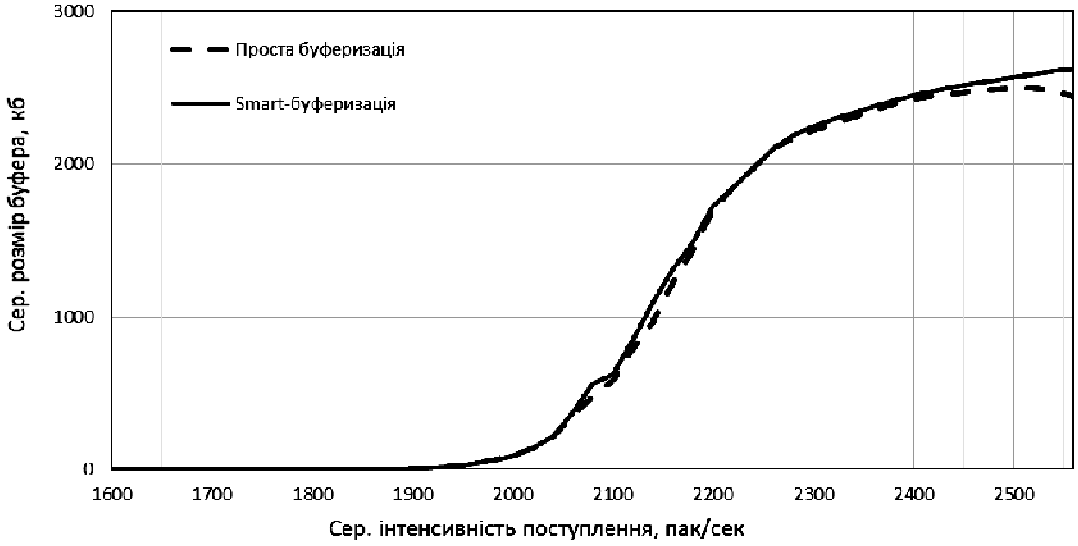

Рис. 12. Порівняння розміру буфера із використанням технології смарт-буферизації та без її використання

Як бачимо із рис. 12, розмір буфера є практично незмінним з використанням технології смарт-буферизації і без її використання. Тепер покажемо, як змінився рівень втрат за однакових вхідних параметрів:

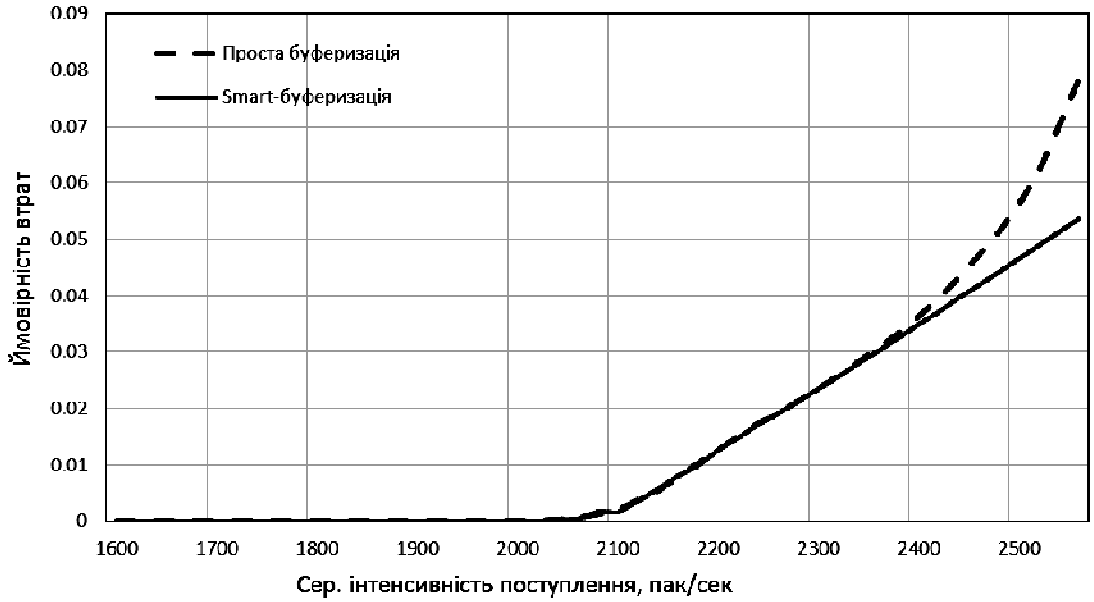

Рис. 13. Залежність ймовірності втрат від інтенсивності надходження пакетів з використанням технології смарт-буферизації та без неї

Результати моделювання показують, що для невеликих інтенсивностей вхідних потоків рівень втрат практично однаковий, однак, якщо зростає інтенсивність надходження пакетів, використання технології смарт-буферизації дає кращі результати. В цьому моделюванні можна побачити, що вже за невеликих значень розміру буфера рівень втрат стає меншим. Рівень втрат знизився на 10 % порівняно із використанням звичайних методів буферизації даних. За вищих значень інтенсивностей надходження пакетів та відповідно рівнів заповнення буферного простору ця різниця буде збільшуватись.

Висновки. У роботі проаналізовано основні методи та принципи буферизації даних у вузлах обслуговування мультисервісного трафіку. Встановлено, що багато фактів оптимальності розміру мережевого буфера, швидше за все, дають можливість визначати верхню межу довжини черги. Представлено модель багаторівневої буферизації даних та описано способи взаємодії на кожному із рівнів. Показано залежності розміру буфера та ймовірності втрат від інтенсивності надходження пакетів на кожному рівні моделі. Запропоновано технологію динамічної зміни буферного простору залежно від рівня втрат, яка дає можливість раціональніше використовувати буферний простір та мінімізувати втрати пакетів. Результати моделювання показали, що використання смарт-буферизації дає можливість знизити рівень втрат на 10 % порівняно із використанням звичайних методів буферизації даних.

1. Кирик М.І, Плесканка Н.М, Андрухів Т.В., Червенець В.В. Дослідження буферизації мультимедійного трафіку в мережах передачі даних // Вісник НУ "Львівська політехніка", № 738 "Радіоелектроніка та телекомунікації", Львів, 2012, – С.100–106. 2. Клейнрок Л. Теория массового обслуживания / Пер. с англ. И. И. Грушко; ред. В. И. Нейман. – М.: Машиностроение, 1979. – С. 292–320. 3. Intel® Processors and Chipsets for Embedded Applications [Електронний ресурс]/ 2013. Режим доступу до статті:http://www.intel.com/content/www/us/en/intelligent-systems/embedded-processors-which-intel-processor-fits-your-project.html/applnots/ap450.pdf. 4. A Map of the Networking Code in Linux Kernel 2.4.20 [Електронний ресурс]/2013. Режим доступу до статті: http://pages.cpsc.ucalgary.ca/~xiaof/linux/linux_net_map.pdf. 5. Deepak Kakadia, Understanding Tuning TCP [Електронний ресурс]/2011. Режим доступу до статті: http://www.oracle.com/us/sun/042646.pdf. 6. Iyer S., Kompella R. R., and McKeown N. Analysis of a memory architecture for fast packet buffers. In Proceedings of IEEE High Performance Switching and Routing, Dallas, Texas, May 2001. 7. Fraleigh C. J. Provisioning Internet Backbone Networks to Support Latency Sensitive Applications. PhD thesis, Stanford University, Department of Electrical Engineering, June 2002. 8. Floyd, S. and Jacobson, V. Random Early Detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking Aug. 1993. 9. Jim Gettys, Kathleen Nichols, Bufferbloat: Dark Buffers in the InternetCommunications of the ACM, Vol. 55 No. 1, Pages 57-65. 10. Sujal Das,Rochan Sankar, Broadcom Smart-Buffer Technology in Data Center Switches for Cost-Effective Performance Scaling of Cloud Applications. April 2012. 11. Abhijit K. Choudhury, Ellen L. Hahne. Dynamic Queue Length Thresholds for Shared-Memory Packet Switches. Bell Laboratories Holmdel, NJ, USA. May 2013. 12. C. J. Fraleigh. Provisioning Internet Backbone Networks to Support Latency Sensitive Applications. PhD thesis, Stanford University, Department of Electrical Engineering, June 2002. 13. Y.J.Anna Gilbert, N.McKeown. Congestion control and periodic behavior. In LANMAN Workshop, March 2001. 14. S.Floyd, V.Jacobson. Random early detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking, 1(4):397–413, 1993. 15. L.Zhang, D.D.Clark. Oscillating behaviour of network traffic: A case study simulation. Internetworking: Research and Experience, 1:101–112, 1990. 16. L. Qiu, Y. Zhang, and S. Keshav. Understanding theperformance of many tcp flows. Comput. Networks, 37(3-4):277–306, 2001. 17. G.Iannaccone, M.May, C.Diot. Aggregate traffic performance with active queue management and drop from tail. SIGCOMM Comput. Commun. Rev., 31(3):4–13, 2001. 18. C.J.Fraleigh.Provisioning Internet Backbone Networks to Support Latency Sensitive Applications. PhD thesis, Stanford University, Department of Electrical Engineering, June 2002. 19. Milton Abramowitz and Irene A. Stegun, eds. Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables. New York: Dover, 1972, Nikolai G. Lehtinen "Error functions", April 2010. 20. H. Bilic, Y. Birk, I. Chirashnya and Z. Machulsky. Deferred Segmentation for Wire-Speed Transmission of Large TCP Frames over Standard GbE Networks. In Hot Interconnects IX, pages 81–85, Aug. 2001.