УДК 621.391

Варфоломеєва О. Г., Мороз О. О. Державний університет телекомунікацій

ДОСЛІДЖЕННЯ МЕТОДІВ УПРАВЛІННЯ ПОТОКОМ В МЕРЕЖАХ NGN ЗА ПРОТОКОЛОМ TCP/IP

Проведено аналіз особливостей архітектури телекомунікаційних мереж в умовах переходу до Глобальної інформаційної інфраструктури. Розглядаються питання, пов'язані з ефективним управлінням потоком з застосуванням розподілених механізмів управління за протоколом TCP. Визначені основні вимоги до механізмів управління потоком за протоколом TCP/IP, виконано аналіз особливостей реалізації механізмів, методів та алгоритму управління потоком в NGN.

Ключові слова: *телекомунікаційна мережа, система управління, TCP, AIMD, адитивне збільшення, мультиплікативне зменшення, повільний старт, управління потоком, перевантаження*

Вступ. **N**GN (**N**ext Generation **N**etwork – мережа наступного покоління) – це мультисервісна мережа телекомунікацій, яка підтримує інтеграцію послуг передавання мови, даних і мультимедіа та базується на IP-мережі (на відміну від ISDN). Основна відмінність мереж наступного покоління від традиційних мереж в тому, що вся інформація, яка циркулює в мережі, розбита на дві складові. Це сигнальна інформація, що забезпечує комутацію абонентів та надання послуг, і безпосередньо дані користувача, що містять корисну інформацію, призначену абоненту (голос, відео, дані). Шляхи проходження сигнальних повідомлень і даних користувача можуть не збігатися [1].

TCP (Transmission Control Protocol) – основний протокол Інтернету. Однією з його головних задач – боротьба з перевантаженнями в мережі (network congestion), коли виникають затори з пакетів [2]. Управління здійснюється шляхом взаємного підстроювання швидкості відправки запитів. Механізм TCP надає потік даних з попередньою установкою з'єднання, здійснює повторний запит даних в разі втрати даних і усуває дублювання при отриманні двох копій одного пакета, гарантуючи тим самим, на відміну від UDP, цілісність переданих даних і повідомлення відправника про результати передачі [3].

TCP використовує форму управління потоком даних, яка називається протоколом із змінним вікном (sliding window). Це дозволяє відправнику передати кілька пакетів, перед тим як він зупиниться і буде чекати підтвердження. При цьому дані передаються швидше, так як відправник не повинен зупинятися і чекати підтвердження кожного разу після відправки пакета.

Управління потоком. В мережах NGN основним протоколом являється протокол TCP/IP завдяки якому відбувається взаємодія між мережами та найголовніше управління потоком в цих мережах. В TCP/IP існує декілька механізмів управління мережею передачі даних. Для початку розглянемо режими зупинки і очікування: відправник передає дані і зупиняється, очікуючи підтвердження, одержувач відправляє підтвердження про отримання даних, після чого передається наступна порція даних и так далі до повної передачі даних. Але справа в тому, що повідомлення передаються по мережі швидко, але не миттєво, тому в середовищі передачі даних може знаходитись деякий об'єм даних. Цей об'єм невеликий для локальних мереж з маленькою затримкою, але в великих сучасних мережах з високошвидкісним з'єднанням та великим терміном очікування підтвердження приводить до зниження продуктивності мережі, що є недоцільним. Тому для прискорення і оптимізації процесу передачі великих об'ємів даних протокол TCP визначає метод управління потоком, який називається методом ковзного вікна, що дозволяє відправникові посилати черговий сегмент, не чекаючи підтвердження про отримання в пункті призначення попереднього сегменту.

Розглянемо базові принципи метода ковзного вікна. Протокол TCP формує підтвердження не для кожного конкретного успішно отриманого пакету, а для всіх даних від початку посилки до деякого порядкового номера *ACKSN* (Acknowledge Sequence Number).

Якщо *ISN* – початковий порядковий номер пакету (а точніше, байту), переданий під час встановлення з'єднання, *SN* – порядковий номер пакету, а *ACKSN* – порядковий номер підтвердження, то після встановлення з'єднання:

$$SN = ISN + 1.$$

Якщо після встановлення з'єднання відправник передав декілька TCP-сегментів, в яких в цілому було передано *n* байт, то як підтвердження успішного прийому, перших *n* байт, висилається порядковий номер підтвердження

$$ACKSN = SN + n + 1 = (ISN + 1) + n + 1.$$

Це означає, що всі дані в байтовому потоці під номерами від $(ISN + 1)$ до $(ISN + 1) + n$ успішно отримані.

Разом з посилкою відправникові порядкового номера *ACKSN* одержувач оголошує також розмір вікна. Це означає, що відправник може посилати дані з порядковими номерами від поточного *ACKSN* до *(ACKSN+розмір_вікна-1)*, не чекаючи підтвердження з боку одержувача. Якщо не буде отримано нове підтвердження (новий *ACKSN*), відправник посилатиме дані, поки він залишається в межах оголошеного вікна. Після цього посилка даних буде припинена до отримання чергового підтвердження і нового розміру вікна. Розмір вікна вибирається так, щоб підтвердження встигали приходити вчасно і зупинки передачі не відбувалося. Розмір вікна може динамічно змінюватися одержувачем. Для тимчасової зупинки посилки даних досить оголосити нульове вікно. Але навіть в цьому випадку через певні проміжки часу будуть від правляться сегменти з одним октетом даних. Це робиться для того, щоб відправник гарантовано дізнався про те, що одержувач знов оголосив ненульове вікно, оскільки одержувач зобов'язаний підтвердити отримання пробних сегментів, а в цих підтвердженнях він вкаже також і поточний розмір свого вікна. У протоколі TCP ковзне вікно використовується для регулювання трафіку і запобіганню переповнювання буфера.

Управління потоком даних за методом ковзного вікна (sliding window) дозволяє відправникові передати декілька пакетів, перш ніж він зупиниться і чекатиме підтвердження. При цьому дані передаються швидше, оскільки відправник не повинен зупинятися і чекати підтвердження кожного разу після відправки пакету.

Наступний метод управління потоком – це механізм повільного старту. Алгоритм повільного старту полягає в тому, що здійснюється дослідження, з якою швидкістю нові пакети повинні відправлятися в мережу, причому ця швидкість повинна відповідати швидкості, з якою прийшли підтвердження з віддаленого кінця.

При роботі з повільним стартом відправляючому TCP додається ще одне вікно: вікно перевантаження (Congestion Window, CWnd). Коли встановлюється нове з'єднання з вузлом, що знаходиться в іншій мережі, розмір вікна перевантаження встановлюється рівним розміру одного сегменту MSS (Maximum segment size - розмір сегменту оголошений віддаленим кінцем). Кожного разу, коли приймається ACK, вікно перевантаження збільшується на один сегмент. (Розмір CWnd вимірюється в байтах, проте при повільному старті розмір завжди збільшується на розмір сегменту. Відправник може передати об'єм даних величиною до мінімального розміру вікна переповнювання і оголошеного вікна. За допомогою вікна перевантаження відправник здійснює управління потоком, тоді як за допомогою оголошеного вікна потоком управляє одержувач.

Відправник починає роботу, відправивши один сегмент і чекаючи підтвердження (ACK) на цей сегмент. Коли ACK отриманий, вікно перевантаження збільшується з одного сегменту до двох, і в цьому випадку можуть бути відправлені два сегменти. Коли кожен з цих двох сегментів підтверджений, вікно перевантаження збільшується до 4 . Таким чином, здійснюється експоненціальне збільшення. У певній точці досягається максимум передачі

для даного з'єднання (об'єднаній мережі), в цьому випадку проміжний маршрутизатор починає відкидати пакети [7].

Розглянемо, як використовується розмір вікна, як здійснюється управління потоком за допомогою вікон і повільного старту, а також як це впливає на пропускну спроможність TCP з'єднання, по якому передаються не інтерактивні дані [7].

У момент часу 0 відправник посилає один сегмент (рис. 1). Оскільки відправник працює з повільним стартом (вікно перевантаження встановлене в один сегмент), він повинен чекати підтвердження на цей сегмент, перш ніж продовжити роботу.

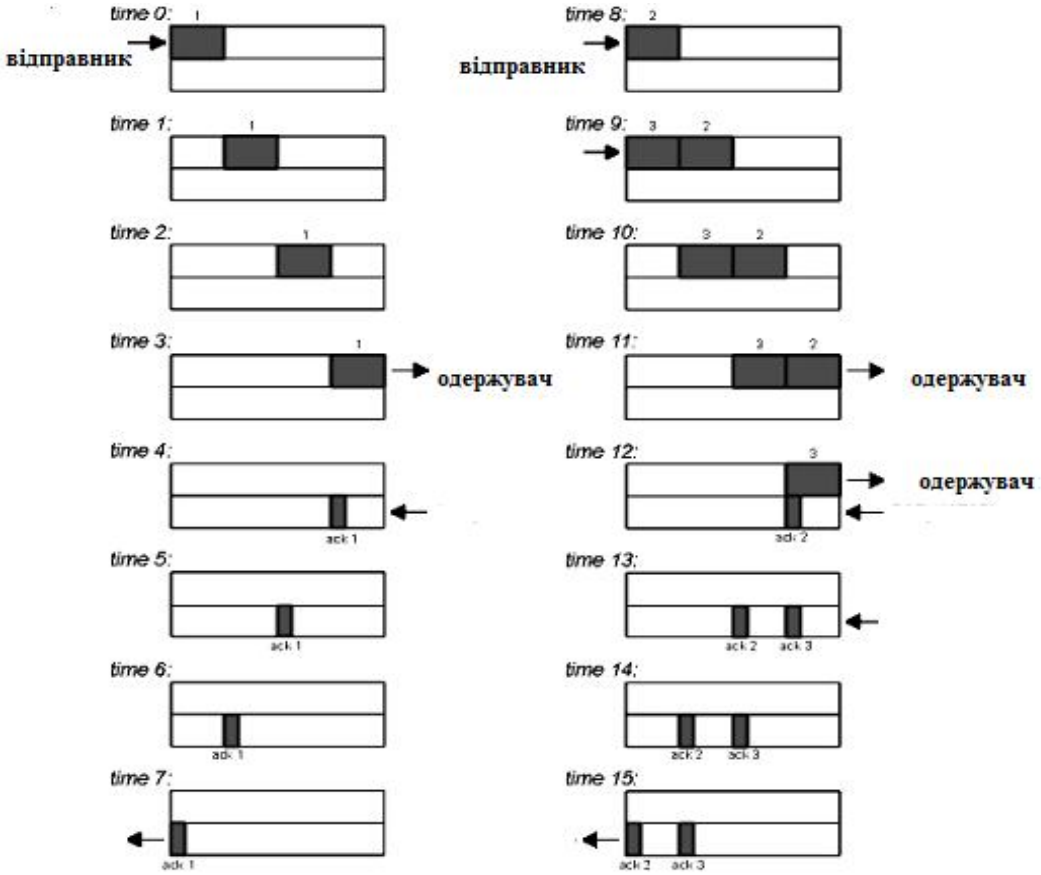

Рис. 1. Зміна пропускної спроможності при передаванні даних

У моменти часу 1, 2 і 3 сегмент проходить по одному проміжку часу вправо. У момент часу 4 одержувач читає сегмент і генерує підтвердження. У моменти часу 5, 6 і 7 підтвердження рухається по одному проміжку часу вліво, назад до відправника. Таким чином, час повернення (RTT – *round-trip time*) складає 8 проміжків часу.

Коли одержувач приймає ACK, він може передати два сегменти (які ми пронумерували як 2 і 3) в моменти часу 8 і 9. Вікно перевантаження зараз складає два сегменти. Ці два сегменти рухаються вправо у напрямку до приймача, де генеруються підтвердження в моменти часу 12 і 13. Проміжки часу між підтвердженнями (ACK) ідентичні проміжкам між сегментами даних [4]. Це поведінка TCP називається самонастроюванням за часом (self-clocking). Оскільки одержувач може генерувати ACK лише тоді, коли дані отримані, по проміжках між підтвердженнями можна визначити швидкість прибуття даних до приймача. Переповнювання може виникнути, коли дані прибувають з швидкого каналу (локальна мережа), а потім вирушають в повільний канал (глобальна мережа). Переповнювання також може виникнути, коли декілька вхідних потоків прибувають на маршрутизатор, вихідна пропускна спроможність якого менше ніж сума вхідних даних.

Регулювання трафіку в TCP передбачає існування двох незалежних процесів: контролю доставки, керованого одержувачем за допомогою параметра Розмір вікна (Window), і

контролю перевантаження, керованого відправником за допомогою Вікна перевантаження (Congestion Window, CWnd), а також Порогу повільного старту (Slow Start Threshold, SSThreth) [4]. Для запобігання втрати пакетів використовується алгоритм запобігання перевантаження**.** Припущення, на якому будується цей алгоритм, полягає в тому, що втрата пакетів сигналізує про те, що в якому-небудь місці мережі між джерелом і призначенням з'явилося перевантаження (а не будь-яке інше пошкодження). Існують дві ознаки, по яких можна визначити, що пакети втрачаються: підтвердження не повертається в строк і отримання дубльованих підтверджень. Запобігання перевантаження і повільний старт – це незалежні один від одного алгоритми, проте, коли виникає перевантаження, необхідно уповільнити швидкість передачі пакетів по мережі, а потім використовувати повільний старт, аби почати все з початку. На практиці ці алгоритми використовуються разом. У кожного відправника є два вікна: вікно, надане одержувачем, і вікно перевантаження CWnd. Розмір кожного з них відповідає кількості байтів, яка відправник має право передати. Відправник керується мінімальним з цих двох значень. При установці з'єднання відправник встановлює розмір вікна перевантаження рівним розміру максимального використовуваного в даному з'єднанні сегменту. Потім він передає один максимальний сегмент. Якщо підтвердження отримання цього сегменту прибуває перш, ніж закінчується час очікування, до розміру вікна додається розмір сегменту, тобто розмір вікна перевантаження подвоюється, і посилаються вже два сегменти. У відповідь на підтвердження отримання кожного з сегментів проводиться розширення вікна перевантаження на величину розміру поточного вікна. Допустимо, розмір вікна дорівнює n сегментам. Якщо підтвердження для всіх сегментів приходять вчасно, вікно збільшується на число байтів, відповідне n сегментам. По суті, підтвердження кожної послідовності сегментів призводить до подвоєння вікна перевантаження [5].

Цей процес експоненціального зростання продовжується до тих пір, поки не буде досягнутий розмір вікна одержувача або не буде вироблена ознака тайм-ауту, що сигналізує про перевантаження в мережі. Наприклад, якщо пакети розміром 1024, 2048 і 4096 байт дійшли до одержувача успішно, а у відповідь на передачу пакету розміром 8192 байти підтвердження не прийшло у встановлений термін, вікно перевантаження встановлюється рівним 4096 байтам. Поки розмір вікна перевантаження залишається рівним 4096 байтам, довші пакети не посилаються, незалежно від розміру вікна, що надається одержувачем. Цей алгоритм називається повільним стартом. Незважаючи на назву він не такий вже і повільний, скоріше експоненціальний.

Запобігання перевантаженню і повільний старт вимагають, аби для кожного з'єднання були визначені дві змінні: вікно перевантаження – CWnd, і розмір порогу повільного старту – SSThresh. Таким чином, окрім вікон одержувача і перевантаження, як третій параметр в нім використовується порогове значення, яке спочатку встановлюється рівним 64 Кбайт. Коли виникає ситуація тайм-ауту (підтвердження не повертається в строк), нове значення порогу встановлюється рівним половині поточного розміру вікна перевантаження.

Але потім, так само як і у попередньому випадку, не використовується алгоритм повільного старту, тому що це приводить до зниження продуктивності мережі, що є недоцільним. Проте цього разу використовується метод адитивного збільшення тобто вікно збільшується лінійно, на один сегмент для кожної наступної передачі. При сигналі про перевантаження розмір вікна перевантаження зменшується в двічі, тобто розмір вікна множиться на 0.5 – це називається мультиплікативне зменшення. По суті, передбачається, що можна урізувати удвічі розмір вікна перевантаження, після чого поступово нарощувати його [5].

Приклад, приведений на рис. 2, ілюструє цей алгоритм.

Алгоритм управління перевантаженням може бути представлений у вигляді наступної послідовності кроків:

Крок 1. Ініціалізація змінних: вікно переповнювання заданого з'єднання CWnd встановлюється рівним одному сегменту (MSS); а розмір порогу повільного старту SSThresh дорівнює 65535 байт.

Крок 2. Порівняння значень вікна переповнювання CWnd і вікна, що оголошене одержувачем (Window). Вибирається менше значення.

Крок 3. Пересилка даних і отримання підтвердження.

Крок 4. Якщо отримання чергового блоку даних підтверджене до настання тайм-ауту, значення збільшується експоненціально (подвоюється) .

Крок 5. Якщо підтвердження не отримане, нове значення порогу встановлюється рівним половині поточного розміру вікна перевантаження, після чого запускається процедура адитивного збільшення, тобто подальше зростання відбувається лінійно з приростом на кожному кроці. Коли досягаємо перевантаження розмір вікна зменшується у двічі, відбувається мультиплікативне зменшення, після чого знову запускається процедура адитивного збільшення, і так поки не передамо всі дані.

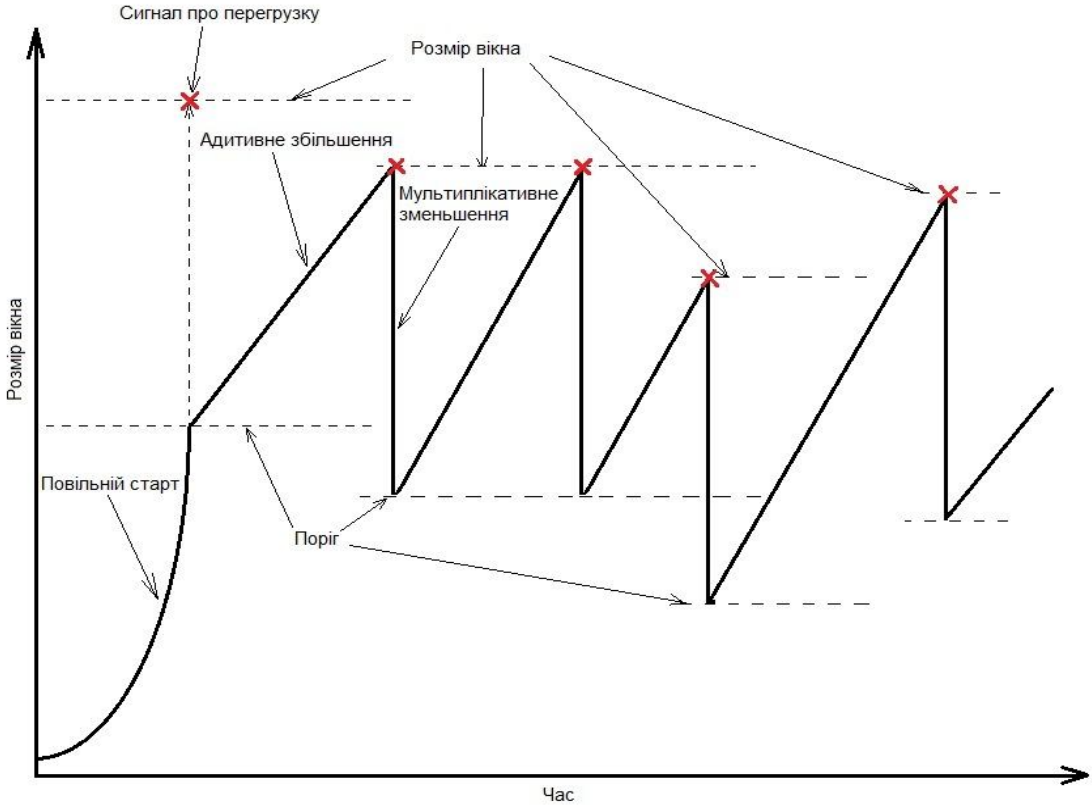

Рис. 2. Повільний старт і AIMD в TCP

Запобігання перевантаження – це спосіб контролювати і управляти потоком даних з боку відправника, тоді як оголошення вікна – це спосіб контролювати потік даних з боку одержувача [2]. Перший процес відстежує заповнення вхідного буфера одержувача, другий - реєструє перевантаження каналу і пов'язані з цим втрати, а також знижує інтенсивність трафіку.

Висновки

На основі проведеного аналізу можна зробити висновок що протокол TCP/IP хоч і не ідеальний але, на відмінно від UDP, є більш зручним механізмом для управління потоком в мережах NGN, і це саме те (простота, надійність і якість) що потрібно операторам та провайдерам телекомунікаційних мереж.

Для управління потоком в мережах NGN є три ключові механізми управління потоком за протоколом TCP/IP:

– *механізм ковзного вікна*: дозволяє відправникові посилати черговий сегмент не чекаючи підтвердження, поки він залишається в межах оголошеного вікна;

– *механізм повільного старту*: дозволяє визначити швидкість прибуття даних до приймача, самонастроюється за часом (self-clocking) и управляє потоком не допускаючи переповнення.

– *алгоритм запобігання перевантаження*: на основі механізму ковзного вікна для запуску використовується повільний старт, для визначення розміру вікна та порогу передачі даних, далі збільшується лінійно щоб не допустити великої кількості втрачених даних та не затопити мережу, далі управління відбувається за принципом AIMD.

Механізми протоколу TCP гарантують забезпечення доставки даних, цілісність переданих даних і повідомлення відправника про результати передачі.

Механізми управління ковзним вікном TCP досить складні, а деталі їх різняться для різних версій протоколу TCP. Удосконалення управління ковзним вікном TCP є областю активних досліджень Більшість пропонованих розширень TCP намагаються зробити управління ковзним вікном менш консервативним з метою поліпшення продуктивності додатків. Проте, основні операції керування ковзним вікном залишаються тими ж.

Список використаної літератури

1. Росляков А. В. / Сети следующего поколения NGN / А. В. Росляков. – Москва : Эко-Трендз, 2008. – 400 с.

2. Правило В. В. / Дослідження методів управління телекомунікаційними мережами в умовах перевантаження / В. В. Правило / І-а науково-технічна конференція Харківського університету повітряних сил, 17 лютого 2005 р. – С. 16.

3. Ализар А. Алгоритми прискорення TCP RemyCC [Електронний ресурс] / А. Ализар // – Режим доступу : https://habrahabr.ru/post/187278/

4. Дуглас Камер. Сети TCP/IP, том 1. Принципы, протоколы и структура / Дуглас Камер. – Москва : «Вильямс», 2003. – 417с.

5. Кришнамурти Б. Web-протоколы. Теория и практика / Б. Кришнамурти, Рексфорд Дж. – Москва : «Издательство БИНОМ», 2002 г. – 592 с.

6. Колченко Г. Ф. Построение модели оптимального проектирования системы управления телекоммуникационными сетями / Г. Ф. Колченко, О. Г. Варфоломеева // Праці 2-ї міжнародної конференції «Проблеми управління мережами та послугами телекомунікацій в умовах конкурентного ринку»: – Вісник УБЕНТЗ. –2003. – №2. – С. 15-18.

7. Дуглас Камер. TCP/IP Крупным планом: главы 20 – 21, 2006 / [Електронний ресурс] / Дуглас Камер. – Режим доступу : http://docs.iho.ru/tcp-ip/tcp20.htm

Автори статті

Варфоломеєва Оксана Григорівна – канд. техн. наук, доцент кафедри телекомунікаційних систем, Державний університет телекомунікацій, Київ. Тел.: +380 (99) 548 0376. E-mail: ogvar13@ gmail.com.

Мороз Олександр Олександрович – студент. Державний університет телекомунікацій, Київ. Тел.: +380 (93) 480 6285. E-mail: iggwakeup@ gmail.com.

Authors of the article

Varfolomeieva Oksana Hryhorivna – candidate of science (technic), associate professor of telecommunication system department. State University of Telecommunications, Kyiv. Tel. +380 (99) 548 0376. E-mail: ogvar13@. gmail.com

Moroz Oleksandr Oleksandrovych – student, State University of Telecommunications. Kyiv. Tel. +380 (93) 480 6285. E-mail: iggwakeup@. gmail.com

Дата надходження
в редакцію: 11.10.2016 р.

Рецензент:
доктор технічних наук, професор А. І. Семенко
Державний університет телекомунікацій