УДК 004.7

О.П. Ігнатенко

МОДЕЛІ КЕРУВАННЯ ПОТОКАМИ ДАНИХ МЕРЕЖІ ІНТЕРНЕТ ЗА УМОВ НЕСТАБІЛЬНОЇ ПОВЕДІНКИ

Дана робота присвячена моделям керування потоками даних у мережах за умов нестабільної поведінки. В роботі представлено огляд проблеми, підкреслена її актуальність, описані існуючі підходи до вирішення. Для мереж зі змінними протоками даних і наявності конфліктуючих користувачів пропонується новий підхід до аналізу роботи мережі на базі конфліктно-керованих потокових моделей. Побудована модель одного класу мереж та доведене існування оптимального у сенсі швидкодії розв'язку задачі завантаження файлу. Розглянуто також ситуацію зловмисного впливу на роботу мережі (атаки на відмову).

Вступ

Система керування потоками даних у мережах Інтернет є складною, адаптивною, взаємопов'язаною і розподіленою. З огляду на постійне зростання ролі Інтернет у сучасному світі, яке відбувається одночасно з його розбудовою, ускладненням і інтелектуалізацією, надзвичайно важливо розуміти й адекватно моделювати процеси керування, що відбуваються всередині мереж.

Однак, враховуючи масштаб систем і евристичний характер багатьох алгоритмів, застосування методів теорії керування до аналізу функціонування мережі Інтернет на сьогодні залишається обмеженим. Теоретичні дослідження в даній області в основному були спрямовані на аналіз вузького місця системи або моделювання проходження одного потоку пакетів. З іншого боку, зусилля практиків зосереджувались на імітаційному моделювання невеликих мереж та дослідження їх роботи за різних умов. Проблема потребувала вирішення, і протягом останнього часу розроблено нові підходи до моделювання процесів керування мережею Інтернет [1 – 3]. Завдяки роботам Келі, Паганіні, Кумара, Сріканта та багатьох інших створено і апробовано комплекс аналітичних моделей, спрямованих на дослідження точок рівноваги у роботі мережі, справедливості розподілу ресурсів, динаміки і стійкості роботи протоколів тощо. Відкритою залишається проблема побудови динамічної керованої моделі, яка б включала поведінку різних конфліктуючих гравців, кожен з яких намагається досягнути власної мети.

1. Аналіз предметної області

Поведінка учасників такої гри та їх вплив на роботу мережі визначається структурою Transmission control protocol (TCP) протоколу, аналітичне моделювання якого завжди було складною задачею. Стандарт TCP описує загальну «рекомендовану» поведінку, починаючи з 1981 року він постійно вдосконалювався і змінювався у відповідності до проблем, що виникали з розвитком Інтернет. На сьогодні запропоновано більш ніж 50 версій сімейства TCP протоколів (далеко не всі з них стали стандартами і були реалізовані), повний огляд і хронологічна картина представлена в оглядовій роботі [4]. Хоча різні частини TCP зазнавали змін, існують базові принципи роботи, які залишаються незмінними і є характерними для всіх реалізацій TCP протоколів.

Одним з головних принципів є такий:

«бути консервативним у тому, що ти робиш; бути ліберальним у тому, що ти приймаєш від інших.»

Бути консервативним для протоколу означає працювати у точній відповідності до своєї специфікації. Якщо специфікація не забороняє прямо певні відхилення у поведінці, це не означає, що їх потрібно реалізовувати. Друга частина означеного принципу означає, що протокол має бути готовий до того, що інша сторона, з якою він взаємодіє, може мати не документовані особливості і він має ці відхилення у поведінці, за можливістю, ігнорувати і не допускати розриву з'єднання. Внаслідок цього принципу TCP протокол з легкістю масш-

табується і може поєднувати зовсім різні мережі в одну надмережу – Інтернет. Однак цей же принцип дозволяє зловмисникам реалізовувати спрямовані дії, націлені на погіршення роботи частини мережі – так звані «атаки на відмову» (Denial of Service Attack) [5].

Принципово така атака може бути здійснена двома способами. Перший спосіб використовує слабкості й помилки програмного забезпечення, служб або протоколів які використовуються об'єктом атаки для надання сервісів користувачам. Використання спеціально сформованих шкідливих пакетів дозволяє ускладнити або заблокувати роботу системи. Другий спосіб полягає у використанні великих обсягів беззмістовного трафіка для завантаження ресурсів, необхідних для обробки запитів звичайних користувачів. Якщо у першому випадку можна захиститися знешкоджуючи слабкості шляхом оновлення програм, то попередити атаку другого типу не так просто. При цьому потужність і складність атак постійно зростає.

Так загальним явищем стала ситуація, коли трафік атаки надсилається з багатьох джерел (такі атаки отримали назву розподілені атаки на відмову). Останні тенденції показують на появу нових типів атак – прихованих атак. У цьому випадку підконтрольні атакуючому комп'ютери отримують доступ до цільового сервісу на цілком законних підставах (наприклад, відвідують веб-сайт компанії) і завантажують канал ресурсоємними операціями (атака погіршення якості) або в певний момент «вибухають» беззмістовним трафіком, що ставить перед системами захисту нові нетривіальні задачі виявлення і протидії.

У даній роботі розглядається підхід до побудови конфліктної керованої моделі взаємодії користувачів за правилами спрощеної версії TCP проколу. Розглянута задача завантаження файлу за найкоротший час. Доведено існування оптимального керування. Розглядається поведінка системи за умов гри двох користувачів, описується ефект «колапсу перевантаження», який мав місце в мережі APARNET. Аналізується зміна поведінки системи за умов атаки на відмову, знайдені умови поглинання вузького місця мережі.

Витоки Інтернету. Початковою причиною створення мережі були наукові цілі. Вузли існували в одиничному екземплярі у дослідницьких центрах, тому природною ідеєю було їх об'єднання на основі максимально широкої концепції. Ключовою стала нова технологія відкритої архітектури мережі, яка дозволяла об'єднувати різнорідні реалізації мереж (провідні, безпровідні, супутникові й інші) в одну мета мережу.

Ця ідея вперше була озвучена Каном в 1972 році. Його проект, що називався «Internetting» був спрямований на розробку надійного протоколу пакетної передачі даних, який би забезпечував зв'язок за умов перешкод і втрат пакетів. При створенні нового протоколу Кан виділив чотири принципові моменти (які є фундаментальними характеристиками сучасного Інтернету)

1. Кожна окрема мережа не повинна змінювати свою внутрішню структуру для під'єднання до Інтернету.
2. Зв'язок має бути «найкращий з можливого». Якщо пакет втрачається, система має надіслати його повторно.
3. Для з'єднання елементів мережі слід використовувати «чорні ящики» – пристрої, що мають максимально швидко передавати потоки пакетів не аналізуючи їх.
4. На глобальному рівні не має бути централізованого керування.

У кінці 70-х років роботу по створенню протоколу передачі пакетів з вузла на вузол (IP) було в принципі завершено. Подальше зростання кількості вузлів і розвиток мережі призвели до необхідності створення керування потоками даних користувачів.

Механізми керування мережами. Як уже зазначалось, Інтернет представляє мережу, яка функціонує за принципом надання найкращого можливого обслуговування. Це означає, що у користувача немає гарантій того, що його пакети дійдуть за призначенням; мережа гарантує лише те, що вона намагатиметься зробити це найбільш ефективним способом.

Причин, з яких обслуговування в мережі Інтернет може бути незадовільним декілька:

- маршрут передачі перервався і пакети намагаються знайти новий;
- якість передачі даних змінюється від зовнішніх факторів (наприклад, безпровідні мережі залежать від погодних умов);
- частину мережі перевантажено.

Перевантаження виникає коли потреби користувачів перевищують наявну кількість ресурсів. Ресурсами можуть бути пропускна здатність, пам'ять, процесорний час та інше. Якщо окрема ланка мережі зіткнулась з несподіваним стрибком трафіка, який перевищує її можливості, то у неї є дві можливі дії – відкинути пакети або помістити їх у чергу.

Зрозуміло, що жодна з цих дій не є задовільним рішенням. Відкидання пакетів призводить до втрати даних користувачів, тому стандартні маршрутизатори, як правило, поміщають надлишкові пакети у чергу, яка функціонує за принципом FIFO (обслуговує пакета у порядку надходження) і відкидає пакети тільки у разі переповнення черги.

Такий підхід дозволяє «перечекати» короткотермінові стрибки трафіка і доставити пакети за потрібною адресою пізніше. Однак це рішення має два суттєвих недоліки. По-перше, навіть короткотермінове збереження пакетів у черзі значно збільшує затримки у їх доставці. По-друге, трафік у мережі Інтернет не відповідає певному фіксованому випадковому процесу, а утворюється в результаті незалежних дій багатьох розподілених агентів. Тому припущення, що за стрибком трафіка буде спад, який дозволить очистити черги у реальних системах може не виконуватися. Доводиться констатувати, що наразі проблема керування процесами перевантаження ланок мережі (тобто передбачення, запобігання і розвантаження) є актуальною і складною. Складність цієї проблеми спричинена тим, що мережа Інтернет розподілена і децентралізована, надсилаючи пакет за певною адресою ми не можемо сказати нічого певного про стан завантаженості мережі до повернення підтвердження про його отримання або втрату.

Для кращого розуміння проблеми слід відзначити основні зовнішні фактори, що впливають на розвиток мереж:

- мережа Інтернет – це сукупність великої кількості мереж, з'єднаних з метою надання доступу до ресурсів кінцевим користувачам. Тут не існує центральної влади або управлінської ієрархії, кожна з мереж управляється локально. Результатом такого підходу була надзвичайна швидкість розвитку Інтернет. Однак, з огляду на відсутність центрального управління, надзвичайно важко швидко запровадити нові алгоритми керування глобально. Більш того, з причин секретності та інших комерційних таємниць провайдери мережевих сервісів уникають надання інформації про схеми трафіка в їхніх мережах;
- комерційний успіх Інтернету призвів до значного здешевлення основних компонентів мереж. У результаті надлишкові потужності коштують набагато менше ніж можливі фінансові втрати від незадоволених якістю користувачів. Крім того, мережа з надлишковими потужностями менш ймовірно буде перевантажена порівняно з мережею з мінімально достатніми обсягами ресурсів.

Наявність високопродуктивних каналів доступу дозволяє зосередити в окремій точці великі обсяги трафіка, створюючи перевантаження у локальних кінцевих мережах. Це може трапитися випадково, під час подій, цікавих великій кількості користувачів (олімпійські ігри, голосування), так і зі зловмисною метою – під час виконання атаки на відмову.

2. Опис проблеми

Перший відомий випадок перевантаження мережі Інтернет стався у 1980-х роках. У внутрішній документації мережі [6] описана ситуація «колапсу перевантаження», коли спостерігалась стійка тенденція погіршення якості роботи мережі внаслідок збільшення швидкості надсилання пакетів користувачами. Проблема полягала у зростанні кількості користувачів та їх потреб у ресурсах мережі але також у особливостях тогочасного протоколу роботи мережі. Користувачі без-

контрольно збільшували свої потоки даних і перевантажували ланки мережі.

Таким чином, виникла необхідність у розробці механізму забезпечення контролю над перевантаженнями мережі, яка призвела до появи TCP протоколу. На сьогодні 80 відсотків усього Інтернет трафіка складає TCP трафік, тому важливо зрозуміти причини і фактори, які спричинили появу й успіх цього протоколу.

Починаючи від стандарту RFC 793 [7], у якому вперше було реалізовано обмеження швидкості надсилання пакетів джерелом з метою недопущення перевантаження вхідного буфера адресата, і до сьогоднішнього дня продовжується розвиток алгоритмів керування потоками даних. Основною причиною цього стрімкого розвитку була поява нових і нових проблем, пов'язаних з розвитком мережі Інтернет. До таких проблем можна віднести: перевантаження, втрати у безпровідних мережах, зміни у порядку надсилання пакетів, затримки у високопродуктивних мережах. Частина з них була зумовлена збільшенням кількості користувачів, частина – технічним прогресом, але кожна спричинила сплеск ідей, експериментів і досліджень зі створення нових, більш ефективних протоколів взаємодії користувачів у мережах.

Робота по створенню ефективного, надійного і безпечного протоколу ще далека від завершення. І якщо перші специфікації TCP представляли собою досить прості евристичні схеми, що використовували мінімум інформації, то сучасні реалізації дозволяють аналізувати втрату пакетів, стан завантаженості мережі, зміни у маршруті доставки; вимірювати відсоток втрат пакетів, зміну часу очікування підтвердження доставки, розміри вузьких місць і перевантаження мережі.

У фундаментальній роботі [8] описуються основні компоненти розподіленої самоорганізованої системи керування потоками даних (рис. 1).

1. Джерела генерують трафік. Це те місце де мають прийматися рішення щодо обмеження потоків даних (коли і скільки пакетів надсилати). Кожне джерело має визначити обмеження свого потоку пакетів. При цьому у нього немає інформації ні про кількість ні про потреби інших джерел, ні про загальний стан мережі. Якщо сумарний потік від джерел буде більше за пропускну здатність мережі, відбудеться перевантаження.

2. Проміжні вузли приймають та ретранслюють пакети за призначенням. У кожного вузла є черга в якій можуть зберігатися пакети за умов перевантаження. Якщо черга переповнюється, то нові пакети втрачаються. При цьому вузол не знає чи це короткотерміновий сплеск чи значне перевищення запитів користувачів межі пропускної здатності.

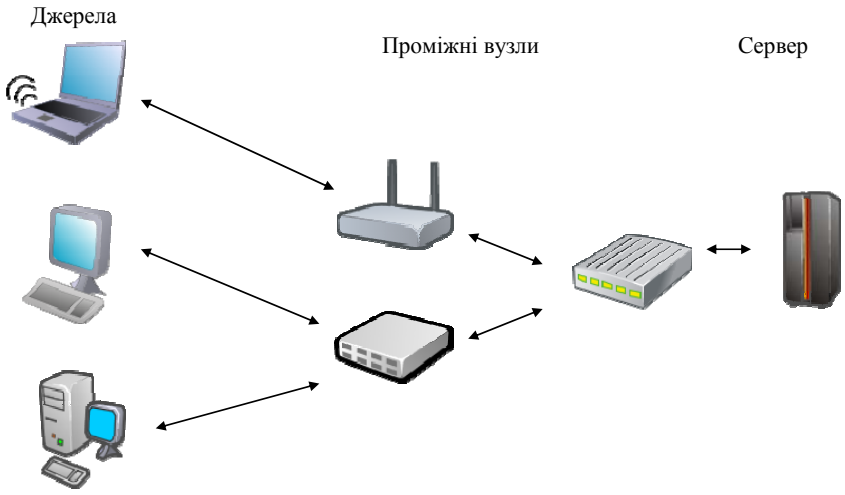

Рис. 1. Загальна схема мережі

3. Сервер надає обслуговування – опрацьовує надіслані джерелами пакети. Сервер має повну інформацію щодо завантаженості мережі та потоків пакетів користувачів і може обчислити їх сумарні потоки та визначити обмеження. Однак ця інформація просто не дійде до джерел оскільки мережа в цей момент вже перевантажена. Крім цього одним з базових принципів побудови мережі Інтернет полягає у тому, щоб тримати внутрішню структуру мережі максимально простою і виносити всю складність функціонування на граничні хости. Це робиться задля того, щоб проміжні маршрутизатори, особливо кореневі, як найшвидше пересилали пакети, до того ж всі майбутні зміни у функціонуванні Інтернет протоколів виносяться, таким чином, на граничні комп'ютери, що дозволяє швидко впроваджувати нові протоколи або нові програми обробки.

Таким чином, механізми керування мають реалізовуватися на стороні користувача і головною задачею такої системи керування потоками є адаптації швидкості обміну пакетами з метою зменшення загрози перевантаження. Зрозуміло, що реалізація суттєво залежить від наявних обмежень, пов'язаних з фізичною архітектурою мережі. Зокрема, з надісланим у мережу пакетом користувача можуть відбутися такі й тільки такі дії:

- пакет успішно дійшов до точки призначення із затримкою. Причини затримки можуть бути різними: значна відстань, затримки у чергах, обробка на проміжних вузлах;
- пакет був втрачений. Причини можуть бути наступні: переповнення черги, відмова адресата, несправність лінії та інше;
- пакет дійшов, але був змінений. Під змінами тут розуміються зміни у його заголовці (іноді використовується для передачі інформації між вузлами мережі) або у його даних (шум, збурення у лінії зв'язку, відмова приладів).

Фактично результати описаних подій вичерпують всю доступну користувачу інформацію про поведінку мережі, тому механізми керування намагаються якомога повно її використовувати.

AQM алгоритми. Можливості керування завантаженістю на рівні джерела обмежені. Система керування перевантаженнями, реалізована тільки на стороні клієнта не може бути ефективною, оскільки:

- протокол роботи у мережі Інтернет не є обов'язковим стандартом. Навіть у рамках стандарту TCP різні реалізації ведуть себе по різному при виникненні втрат. Інші типи протоколів можуть взагалі не зменшувати свою швидкість при перевантаженні (як, наприклад, CBR мережі);
- деякі протоколи більш «агресивні» і захоплюють більші потужності каналів зв'язку, якщо їх вчасно не обмежити. Інформація для таких обмежень наявні тільки на маршрутизаторі, який опрацьовує всі потоки даних;
- маршрутизатор може стати ціллю атаки на відмову, або бути перевантажений великими потоками трафіка, тому йому потрібно мати власну систему захисту від перевантажень.

Найпростішим механізмом AQM, який був реалізований на маршрутизаторах з самого початку мережі Інтернет, являється DropTail. Згідно з ним пакети, що надійшли поміщуються у загальну чергу і опрацьовуються згідно з порядком надходження. Якщо черга переповнюється, пакети відкидаються. Така система виглядає природно, але при роботі стандартного TCP у комбінації з DropTail було виявлено [9] два суттєвих недоліки:

залипання. Один або декілька потоків даних займають чергу і не дозволяють іншим з'єднанням отримати доступ;

переповнення. Якщо черги майже повністю заповнені і відбувся стрибок трафіка, то відбудеться втрата багатьох пакетів. У результаті відразу багато з'єднань обмежать свою швидкість і канал буде недовантажено. Оскільки класичний TCP підвищує свою швидкість надсилання аж до виникнення втрат, то черги знову переповняться і ситуація переповнення буде повторюватися періодично.

З цього можна зробити наступний висновок – невеликі черги сприяють покращенню загальної пропускної здатності і

зменшують ризик втрат. Оскільки стандартний TCP реагує тільки на втрату пакетів, то необхідною умовою розв'язання задачі є надання маршрутизатору права превентивно відкидати пакети ще до виникнення ситуації переповнення. Прикладом такого алгоритму є RED – алгоритм, який починає відкидати пакети з певною ймовірністю, що залежить від завантаженості черги. В даній роботі будемо вважати, що маршрутизатори працюють на базі DropTail.

Математичне моделювання мережі Інтернет. Процеси, що відбуваються у мережі ставлять перед дослідниками нові задачі керування, що виходять за рамки існуючої теорії. На рівні потоків даних такими задачами є:

- ефективне використання наявних ресурсів. Завантаженість (використання) ланок мережі має бути максимальною;
- справедливий розподіл ресурсів між користувачами. Доступ користувачів до ресурсів має бути прозорим і прогнозованим;
- низькі затримки пакетів у чергах. Для деяких прикладних програм низькі затримки є необхідною умовою роботи;
- низькі втрати пакетів. Повторне надсилання пакетів не є ефективним використанням мережі;
- адаптивність і стійкість роботи. Стабільна і прогнозована поведінка мережі є необхідною умовою ефективного застосування алгоритмів керування.

Основним традиційним інструментом теоретичних досліджень в області комунікаційних технологій є моделі теорії черг, які розглядають поведінку мережі на рівні окремих пакетів, що пересилаються і опрацьовуються з певними затримками, щільність ймовірнісного розподілу яких відома. Основні результати в цій області суттєво залежать від припущень щодо властивостей вхідного потоку пакетів. Зокрема, як правило, він вважається розподіленим за відомим законом, цей розподіл не залежить від часу або завантаження мережі. Типовим спрощенням також є припущенням про те, що вхідний процес є пуасонівським.

Механізм керування завантаженням є складним алгоритмом, що використовує зворотній зв'язок, змінюється в часі і не може вважатись випадковим, тому застосування тут результатів з теорії черг не є ефективним. Новий підхід був започаткований у роботі Келі [1], де запропоновано моделі керування на рівні потоків даних. Такий підхід дозволив використати потужний апарат теорії керування і представити роботи мережі «в цілому», відкидаючи несуттєві деталі. В рамках цього підходу було отримано багато вагомих результатів в області аналізу стійкості поведінки мереж, які описані, наприклад, в оглядових роботах [10, 11] та в монографії [3]. Стабільна і прогнозована поведінка мережі є необхідною умовою ефективного застосування алгоритмів керування і тому на ній у першу чергу зосереджена увага дослідників.

Можна виділити два основних фундаментальних напрямки в яких просуваються роботи в цій області. Перший з них стосується пошуку умов існування станів рівноваги, в яких може перебувати мережа за роботи даного протоколу (рівновага тут може визначатись у контексті досягнення вищезазначених цілей – максимально ефективного використання ланок, рівномірності розподілу ресурсів тощо). Визначаючи функцію корисності протоколу в явному вигляді та застосовуючи до цієї задачі методи оптимізації, можна знайти умови існування екстремуму та їх зв'язок з параметрами роботи мережі.

Інший напрямок пов'язаний з динамікою функціонування протоколу. Головною проблемою тут є дослідження стійкості знайдених точок рівноваги за різних умов (наявності затримок, перебоїв у роботі мережі, розмірів буферів і каналів). З огляду на кількість версій TCP протоколу та різних можливих комбінацій з AQM алгоритмами (які суттєво впливають на роботу мережі) в цій області зосереджені зусилля більшості дослідників. Як правило поведінка мережі аналізується за стаціонарних «нормальних» умов. При цьому вхідний потік пакетів вважається стаціонарним і не враховує можливості зміни поведінки користувача, яка, в реальності, залежить від поставлених перед ним задач. Наприклад, задача завантаження файлу та задача

встановлення аудіо-зв'язку генерують принципово різний трафік.

Окремим випадком є ситуація коли користувач має задачу *погіршити* роботу мережі. В даній роботі пропонується підхід до моделювання, який би включав такого роду ситуації і дозволяв застосувати до них теоретичні інструменти теорії конфліктно-керованих систем.

3. Моделі керування потоками даних у мережі Інтернет

Потокова модель мережі. Розглянемо математичну модель мереж, що складаються з вузлів, з'єднаних ланками. Кожен вузол може бути користувачем, маршрутизатором або сервером (рис. 2). Користувачу потрібно завантажити певний обсяг інформації на сервер і він може надсилати у мережу пакети з даними. Потік пакетів можна описати швидкістю передачі – функцією $\lambda(t)$. Черги на вузлах мережі описуються змінними $q_i(t)$, $i = 1,...,n$. Маршрутизатор представляє собою вузол мережі, який складається з двох черг та спільного ресурсу (пам'яті, процесорного часу). Пакети користувача потрапляють у буфер $q_1(t)$ і опрацьовуються або зберігаються у черзі в залежності від наявності вільних ресурсів. Маршрутизатор пересилає пакети користувача зі швидкістю $u_1(t)$ на сервер і вони потрапляють у буфер $q_2(t)$. Сервер опрацьовує дані (перевіряє, копіює, зберігає) з швидкістю $u_2(t)$, формує пакети підтвердження (АСК пакети) і надсилає їх на маршрутизатор. АСК пакети набагато менші за обсягом пакетів з даними (фактично це заголовки пакетів), до того ж вони несуть важливу інформацію про отримання даних, тому маршрутизатори опрацьовують їх у першу чергу. Після отримання підтвердження користувач отримує інформацію про доставку вказаної частини даних. Якщо в якийсь момент часу черга виявляється переповненою, то нові пакети відкидаються і втрачаються. Цей процес описується функціями $l_i(t)$, які будуть визначені пізніше. У разі виникнення втрати пакетів користувач має надіслати їх повторно. Розглянемо задачу оптимального (у сенсі швидкодії) завантаження файлу на сервер для мережі, показаної на рис. 2.

Потокова модель процесу завантаження описується диференціальним рівнянням:

$$\dot{q} = \lambda + Bu(t), \qquad (1)$$

де B – матриця, що описує структуру мережі. В роботах [12, 13] та інших досліджуються умови, за яких такі моделі є стійкими, тобто коли для заданих $q(0)$ та λ існує керування $u(t)$ і час $T \geq 0$, такі, що

$$q(T) = q(0) + T\lambda + \int_0^T Bu(\tau)d\tau = 0. \qquad (2)$$

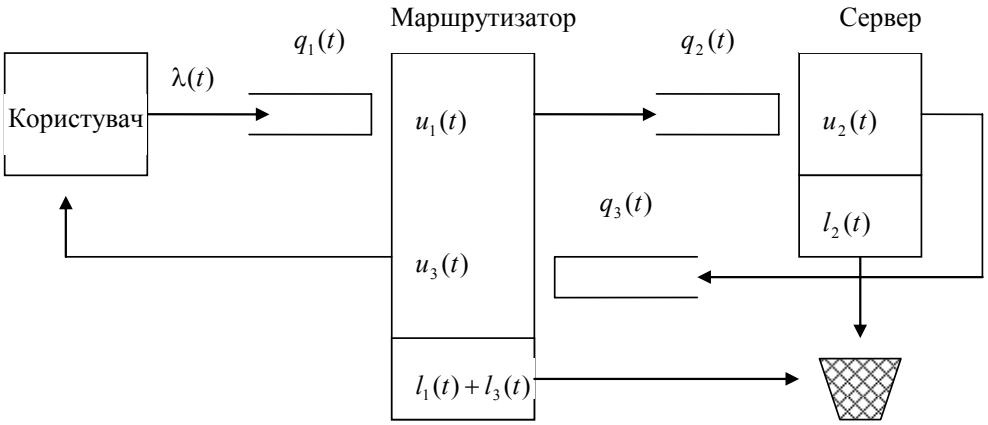

Рис. 2. Модель мережі з втратами і підтвердженням доставки

На керування $u(t)$ накладені обмеження - $u(t) \in U$, де U - компакт з R^n. Для багатьох моделей мереж він допускає спрощений запис

$$Cu \le \overline{1},$$

де C - *структурна матриця* [13], а нерівність розуміється порядково.

При цьому мінімальний $T(q(0))$, для якого виконується (2), називається часом вичерпання черги. Пошук мінімального часу для системи (2) – це задача оптимального керування, яка має розв'язок для λ та початкового вектора $q(0)$ якщо виконується включення

$$q(0) - \lambda T \in T \cdot BU. \qquad (3)$$

Включення (3) дає можливість визначити момент часу $T(q(0))$ в загальному випадку. Якщо структура задачі (1) описується матрицями B і C, то можлива побудова ефективних умов стійкості мережі. Окремі ідеї і результати допускають узагальнення і для мереж з довільними потоками. Опишемо головну ідею цього підходу, яка полягає у введенні поняття *завантаженості* вузлів мережі.

1. Визначимо *матрицю завантаженості* за допомогою наступного рівняння:

$$\Xi = -CB^{-1} = M^{-1}[I - R^T]^{-1},$$

де M – невироджена діагональна матриця, а обернена матриця розуміється у сенсі $[I - R^T]^{-1} = \sum_{k=0}^{\infty} R^k = \sum_{k=0}^{n} R^k$. Позначимо ξ^s, $s \in S$ – вектори-рядки, що утворюють матрицю Ξ.

2. *Вектором завантаження* назвемо вектор $\rho = \Xi\alpha$. Введемо позначення $\rho_\bullet = \max_{s \in S} \rho_s$, тоді вузьким місцем мережі називаються такі вузли s, що $\rho_s = \rho_\bullet$.

Відомі твердження [13].

1. Модель (1) може бути стабілізована тоді і тільки тоді, коли $\rho_\bullet < 1$. При цьому мінімальний час вичерпання черги дорівнює

$$T(q) = \max_{s \in S} \frac{(\xi^s, q)}{1 - \rho_s}.$$

Одним з можливих керувань, що забезпечують стабілізацію системи є вектор

$$u^* = -B^{-1}\left(\alpha + \frac{q(0)}{T(q(0))}\right).$$

2. Необхідною і достатньою умовою оптимальності стратегії є максимальне завантаження вузьких місць.

Цей підхід до моделювання дає можливість оцінити поведінку системи «в цілому», визначити вузькі місця і проаналізувати стійкість для фіксованого потоку вхідних пакетів.

Проблема, однак, полягає у тому, що процеси, які відбуваються в мережах, характеризуються значною кількістю різнотипних користувачів, кожен з яких впливає на роботу системи відповідно до своїх задач і можливостей. Поведінка кожного користувача, взагалі кажучи, змінюється в часі і може залежати від роботи мережі.

Недоліком описаного підходу також є вимога стійкості, яка суттєво обмежує область застосування моделей до аналізу комп'ютерних систем – це, фактично, означає, що пропускні здатності мережі завжди більші за сумарні потреби всіх користувачів. Зрозуміло, що в реальних системах ця умова порушується (принаймі тимчасово).

Таким чином, виникає необхідність у розробці нового підходу до побудови потокових керованих моделей комп'ютерних мереж зі змінними потоками і втратами.

Модель зі змінними потоками і втратами. Розглянемо задачу завантаження файлу. Нехай задана загальна кількість пакетів λ_{int}, які потрібно завантажити на сервер за найкоротший час. Завантаження вважається закінченим, якщо кількість підтверджених пакетів досягає заданої. Введемо додаткову змінну $q_4(t)$, яка описує кількість *підтверджених* пакетів.

Динаміка системи описується системою диференціальних рівнянь:

$$\begin{aligned} \dot{q}_1(t) &= \lambda(t) - u_1(t) - l_1(t), \\ \dot{q}_2(t) &= u_1(t) - u_2(t) - l_2(t), \\ \dot{q}_3(t) &= u_2(t) - u_3(t) - l_3(t), \\ \dot{q}_4(t) &= u_3(t) \end{aligned} \qquad (4)$$

при цьому на фазові змінні і функції накладені наступні обмеження:

$$\begin{gathered} 0 \le \lambda(t) \le \lambda^{\max}, \\ u_i(t) \ge 0,\ \frac{u_1(t)}{\mu_1} + \frac{u_3(t)}{\mu_3} \le 1,\ u_2(t) \le \mu_2, \\ 0 \le q_i(t) \le q_i^{\max},\ q_i(t_0) = 0,\ a(t_0) = 0, \\ i = 1,2,3. \end{gathered}$$

Функції $l_i(t)$ описують процес переповнення черги і працюють у відповідності до алгоритму DropTail:

$$l_1(t) = \begin{cases} 0, & q_1(t) < q_1^{\max} \\ \lambda_1(t) - u_1(t), & q_1(t) = q_1^{\max} \end{cases},$$

$$l_2(t) = \begin{cases} 0, & q_2(t) < q_2^{\max} \\ u_1(t) - u_2(t), & q_2(t) = q_2^{\max} \end{cases},$$

$$l_3(t) = \begin{cases} 0, & q_3(t) < q_3^{\max} \\ u_2(t) - u_3(t), & q_3(t) = q_3^{\max} \end{cases}. \qquad (5)$$

Функції обробки пакетів на маршрутизаторах $u_i(t)$ в даному прикладі працюють за принципом *FIFO*. Крім цього на першому вузлі більший пріоритет надається пакетам підтвердження. Загальноприйняті [13] визначення функцій керування призводять до виникнення ковзних режимів, що ускладнює їх аналіз. Ми пропонуємо новий підхід до визначення $u_i(t)$, який усуває ковзні режими.

$$u_1(t) = \begin{cases} \min\left(\lambda(t), \mu_1\left(1 - \dfrac{u_3(t)}{\mu_3}\right)\right), & q_1(t) = 0 \\ \mu_1\left(1 - \dfrac{u_3(t)}{\mu_3}\right), & q_1(t) > 0 \end{cases};$$

$$u_2(t) = \begin{cases} \min(\mu_2, u_1(t)), & q_2(t) = 0 \\ \mu_2, & q_2(t) > 0 \end{cases};$$

$$u_3(t) = \begin{cases} \min(\mu_3, u_2(t)), & q_3(t) = 0 \\ \mu_3, & q_3(t) > 0 \end{cases}. \qquad (6)$$

Запишемо (4) у скороченій формі $\dot{q} = f(\lambda(t), q(t)), \dot{a} = u_3(t)$. Ця система диференціальних рівнянь визначена в області $Q \times T$, де $Q = \left\{ q \in R^3 : 0 \le q_i \le q_i^{\max} \right\}$, $T = \left\{ t : t \ge 0 \right\}$. Функцію $\lambda(t)$ будемо вважати кусково неперервною. Будемо говорити, що функція задовольняє умовам Каратеордорі, якщо в області $Q \times T$:

1) $f(t,q)$ визначена для майже всіх t і неперервна за q;

2) $f(t,q)$ вимірна за t для кожного q;

3) $|f(t,q)| \le m(t)$, де $m(t)$ інтегрована за Лебегом на кожному скінченому відрізку.

Твердження 1. Для системи (4) з функціями (5), (6) існує розв'язок для будь-якої кусково неперервної $\lambda(t)$.

Доведення. Якщо при $t_0 \le t \le t_0 + a$, $|q - q_0| \le b$ функція $f(t,q)$ задовольняє умовам Каратеодорі, то на відрізку $[t_0, t_0 + d]$ існує розв'язок задачі $\dot{q} = f(t,q)$, $q(t_0) = q_0$ (теорема Філіпова [14]). Як легко бачити, для системи (1) умови Каратеодорі не виконуються. В області $M = \{q \in R^3 : \exists i \in \{1,2,3\} : q_i = q_i^{\max}\}$ функції $l_i(t,q)$ можуть мати розрив по q. Однак, оскільки функції $\lambda(t)$, $u_i(t)$ є кусково неперервними за t, то траєкторія, що потрапила на множину M не може відразу її покинути. Отже ковзного режиму не виникає і згідно з підходом, описаним у [14] розв'язком буде абсолютно неперервна функція $q(t)$, така, що

$$\begin{gathered} \dot{q}(t) = f(t, q(t)), \text{ якщо } q(t) \in Q \times T \setminus M, \\ \dot{q}_i(t) = 0, \text{ якщо } q_i(t) \in M \text{ і} \\ f_i(t, q(t)) \ge 0. \end{gathered}$$

Якщо у певний момент часу виконується $f_i(t,q(t))<0$, то траєкторія залишає M.

Моментом закінчення завантаження назвемо перший момент часу t для якого $a(t)\ge\lambda_{\text{int}}$ і оптимальність будемо розуміти тільки у сенсі швидкодії.

Твердження 2. Для системи (1) з початковою умовою $q(t_0)=0$ існує функція $\lambda(t)$, така, що момент закінчення завантаження $T<\infty$.

Якщо керування $\lambda^*(t)$ оптимальне, то вузьке місце максимально завантажене протягом усього часу завантаження файлу.

Якщо за керування $\lambda^*(t)$ вузьке місце максимально завантажене протягом усього часу завантаження файлу, то відповідний час оптимальний.

Доведення. Покладемо $\lambda(t)=\lambda_{\min}$, де $\lambda_{\min}$ знаходиться з умови

$$\max\left(\frac{\lambda_{\min}}{\mu_2},\frac{\lambda_{\min}}{\mu_1}+\frac{\lambda_{\min}}{\mu_3}\right)=1.$$

Для $\lambda_{\min}$ керування приймають значення $u_1(t)=u_2(t)=u_3(t)=\lambda_{\min}$. Оскільки $q(t_0)=0$ і $\dot q(t)=0$ для заданого $\lambda(t)=\lambda_{\min}$, то $q(t)\equiv 0$. Отже, в момент часу $T=\dfrac{\lambda_{\text{int}}}{\lambda_{\min}}$ виконується $a(t)\ge\lambda_{\text{int}}$.

Нехай керування $\lambda^*(t)$ оптимальне, тобто T^* – мінімально можливий час закінчення роботи. Нехай $\dfrac{1}{\mu_2}\ge\dfrac{1}{\mu_1}+\dfrac{1}{\mu_3}$, тоді вузьким місцем являється вузол 2. Якщо він працює на повній завантаженості, то обробити всі пакети можливо за час не менший за $\widetilde T=\dfrac{\lambda_{\text{int}}}{\mu_2}$. Отже, $T^*\ge\widetilde T$. З іншого боку, існує $\lambda(t)=\lambda_{\min}$, для якої у момент часу $\hat T=\dfrac{\lambda_{\text{int}}}{\lambda_{\min}}$ виконується $a(t)\ge\lambda_{\text{int}}$. Оскільки $\dfrac{1}{\mu_2}\ge\dfrac{1}{\mu_1}+\dfrac{1}{\mu_3}$, то $\lambda_{\min}=\mu_2$. Отже $T^*=\widetilde T$.

Доведення для випадку $\dfrac{1}{\mu_2}\le\dfrac{1}{\mu_1}+\dfrac{1}{\mu_3}$ аналогічне.

Нехай за керування $\lambda^*(t)$ вузьке місце максимально завантажене протягом усього часу завантаження файлу. Нехай $\dfrac{1}{\mu_2}\ge\dfrac{1}{\mu_1}+\dfrac{1}{\mu_3}$, тоді максимальне завантаження означає, що $u_2(t)=\mu_2$, $t\in\left[0,T^*\right]$. Тоді $\widetilde T=\dfrac{\lambda_{\text{int}}}{\mu_2}=T^*$ (оскільки $\dfrac{\lambda_{\text{int}}}{\mu_2}\ge\dfrac{\lambda_{\text{int}}}{\mu_1}+\dfrac{\lambda_{\text{int}}}{\mu_3}$). Припустимо, що існує оптимальне керування $\hat\lambda(t)$, таке що $\hat T<T^*$, це означає, що завантаження відбулось раніше ніж всі пакети пройшли через вузьке місце, що неможливо.

Міркування для випадку $\dfrac{1}{\mu_2}\le\dfrac{1}{\mu_1}+\dfrac{1}{\mu_3}$ аналогічні.

Наслідок. Керування $\lambda(t)=\lambda^{\max}$ завжди є оптимальним.

Модель завантаження з двома користувачами. Розглянемо систему з двома користувачами (рис. 3) що завантажують файли на один сервер. При цьому вони використовують спільну мережу. Для моделювання процесу їх взаємодії в рамках підходу потокових моделей і теорії ігор, введемо *віртуальні з'єднання* $q_i(t)$, $k=1,...,6$. Віртуальне з'єднання означає, що ресурси одного вузла розподіляються (за допомогою функції $u_k(t)$) між обслуговуванням різних потоків пакетів користувачів. Користувач i впливає на систему за допомогою функції $\lambda_i(t)$, намагаючись мінімізувати свій час завантаження.

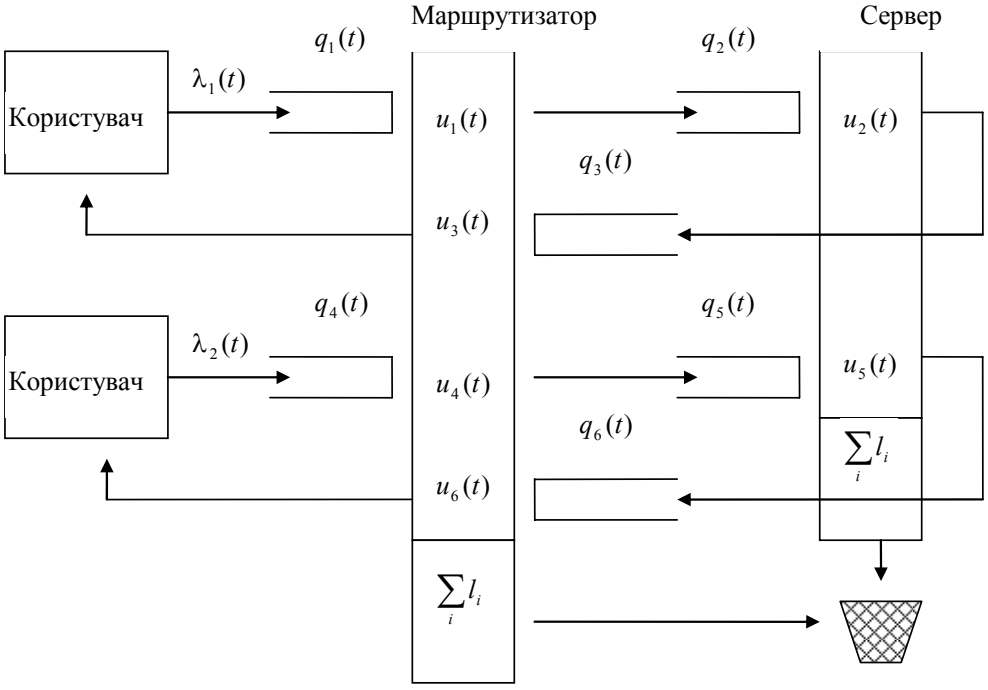

Рис. 3. Модель системи з двома користувачами

Введемо вектор вхідного потоку $\overline{\lambda}(t) = (\lambda_1(t),0,0,\lambda_2(t),0,0)^T$, та вектор черг $\overline{q}(t) = (q_1(t),...,q_6(t))^T$. Керування і втрати описуються векторами:
$\overline{u}(t) = (u_1(t),...,u_6(t))^T$, $\overline{l}(t) = (l_1(t),...,l_6(t))^T$.

Фізичні обмеження буферів позначимо $p^{\max} = (p_1^{\max}, p_2^{\max}, p_3^{\max})^T$.

Структура мережі задається матрицями:

$$C = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix},$$

$$D = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix},$$

$$B = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix},$$

$$M = \begin{bmatrix} \frac{1}{\mu_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{1}{\mu_6} \end{bmatrix}.$$

Тоді динаміка роботи мережі записується у вигляді рівняння:

$$\dot{\overline{q}}(t) = \overline{\lambda}(t) + B\overline{u}(t) - \overline{l}(t),$$

$$Dq(t) \le p^{\max},\ C \cdot M \cdot Du(t) \le \overline{1}. \quad (7)$$

Нерівності розуміються порядково.

Визначимо функції керування $u_k(t)$ та втрат $l_k(t)$, $k = 1,...,6$.

$$u_1(t) = \begin{cases} \min(\lambda_1(t), \mu_1(1 - \frac{[Du(t)]_3}{\mu_3})) \frac{\lambda_1(t)}{[D\lambda(t)]_1}, \\ \qquad q_1(t) = 0 \\ \left(\frac{\mu_3 - u_3(t) + u_6(t)}{\mu_3}\right) \frac{\mu_1 q_1(t)}{p_1(t)}, \\ \qquad q_1(t) > 0 \end{cases}$$

$$u_4(t)=\begin{cases}\min\left(\lambda_2(t),\mu_1\left(1-\dfrac{[Du(t)]_3}{\mu_3}\right)\right)\dfrac{\lambda_1(t)}{[D\lambda(t)]_1}, & q_4(t)=0\\ \left(1-\dfrac{u_3(t)+u_6(t)}{\mu_3}\right)\dfrac{\mu_1 q_1(t)}{p_1(t)}, & q_4(t)>0\end{cases}$$

$$u_2(t)=\begin{cases}\min(\mu_2,u_1(t))\dfrac{u_1(t)}{[Du(t)]_2} & q_2(t)=0\\ \mu_2\dfrac{q_2(t)}{p_2(t)}, & q_2(t)>0\end{cases},$$

$$u_5(t)=\begin{cases}\min(\mu_2,u_4(t))\dfrac{u_4(t)}{[Du(t)]_5} & q_5(t)=0\\ \mu_2\dfrac{q_4(t)}{p_2(t)}, & q_4(t)>0\end{cases}$$

$$u_3(t)=\begin{cases}\min(\mu_3,u_2(t))\dfrac{u_2(t)}{[Du(t)]_3} & q_3(t)=0\\ \mu_2\dfrac{q_3(t)}{p_3(t)}, & q_3(t)>0\end{cases},$$

$$u_6(t)=\begin{cases}\min(\mu_3,u_5(t))\dfrac{u_5(t)}{[Du(t)]_6} & q_6(t)=0\\ \mu_2\dfrac{q_6(t)}{p_3(t)}, & q_6(t)>0\end{cases}$$

$$l_i(t)=\begin{cases}0, & p_j(t)<p_j^{\max}\\ [\overline{\lambda}(t)+B\overline{u}(t)]_i, & p_j(t)=p_j^{\max}\end{cases}. \quad (9)$$

Твердження 3. Нехай задані функції $\lambda_1^*(t)$, $\lambda_2^*(t)$ і $\lambda_1^*(t)\neq\lambda_1^{\max}$. Позначимо час закінчення завантаження T_1^*, T_1^* відповідно. Тоді перший користувач може за допомогою керування $\hat{\lambda}_1(t)\equiv\lambda_1^{\max}$ зменшити свій час завантаження.

Доведення. Зафіксуємо $\lambda_1^*(t)$, $\lambda_2^*(t)$. Розглянемо $p_1(t)=q_1(t)+q_2(t)$, де

$$\dot{q}_1(t)=\lambda_1^*(t)+u_1^*(t)-l_1^*(t),$$
$$\dot{q}_2(t)=\lambda_2^*(t)+u_2^*(t)-l_2^*(t).$$

Нехай $\dfrac{1}{\mu_2}\geq\dfrac{1}{\mu_1}+\dfrac{1}{\mu_3}$, тоді другий вузол являється вузьким місцем і час закінчення залежить від μ_2 і співвідношення між $\lambda_1(t)$ і $\lambda_2(t)$. Чим більше $\lambda_1(t)$, тим більше пакетів потрапляють у чергу і тому швидше опрацьовуються. Мінімальний час закінчення відповідає максимальному завантаженню $\hat{\lambda}_1(t)\equiv\lambda_1^{\max}$. Для $\hat{\lambda}_1(t)\equiv\lambda_1^{\max}$, $\hat{\lambda}_2(t)\equiv\lambda_2^{\max}$ моменти завершення дорівнюють:

$$\hat{T}_1=\frac{\lambda_1^{\text{int}}(\lambda_1^{\max}+\lambda_2^{\max})}{\min(\mu_2,\lambda_1^{\max})\lambda_1^{\max}},$$

$$\hat{T}_2=\frac{\lambda_2^{\text{int}}(\lambda_1^{\max}+\lambda_2^{\max})}{\min(\mu_2,\lambda_2^{\max})\lambda_2^{\max}}.$$

Нехай $\dfrac{1}{\mu_2}<\dfrac{1}{\mu_1}+\dfrac{1}{\mu_3}$. У цьому випадку мінімальний час завантаження визначається розподілом ресурсів на першому вузлі. Оскільки пакети, що потрапляють на лінію 3 (пакети підтвердження) більш пріоритетні, то час закінчення буде залежати від μ_1, μ_3. Для максимальних керувань перший час закінчення завантаження дорівнює

$$\hat{T}_1=\frac{\lambda_1^{\text{int}}(\lambda_1^{\max}+\lambda_2^{\max})}{\min(\mu_1,\lambda_1^{\max})\lambda_1^{\max}}+\frac{\lambda_1^{\text{int}}(\lambda_1^{\max}+\lambda_2^{\max})}{\min(\mu_3,\lambda_1^{\max})\lambda_1^{\max}}.$$

Якщо керування $\hat{\lambda}_2(t)$ на певному проміжку часу менше за максимальне, то протягом цього періоду часу $\hat{u}_4(t)<u_4^*(t)$, отже за той самий період часу буде оброблено більше пакетів першого користувача. Застосувавши ці міркування до решти вузлів, та врахувавши обмеження $u_6(t)+u_3(t)\leq\mu_3$, отримуємо $\hat{u}_3(t)>u_3^*(t)$.

Наслідок. Керування $\hat{\lambda}_1(t)\equiv\lambda_1^{\max}$, $\hat{\lambda}_2(t)\equiv\lambda_2^{\max}$ являються рівновагою за Нешем для системи (7). Таким чином, наявність втрат та обробка, пропорційна до кількості пакетів користувачів у системі (7) призводить до ситуації, коли кожному користувачу вигідно надсилати свої дані максимально швидко (навіть перевантажуючи систему). Твердження 3 природнім чином узагальнюється на випадок N користувачів.

Модель роботи системи за атаки на відмову. Розглянемо роботу системи (7) за наявності атаки. В даній роботі ми розглядаємо спрощену модель поглинаючої атаки [5]. Трафік атаки потрапляє на сервер, вимагає ресурсів на обробку та, після обробки, відкидається як помилковий. Виявляється, що атака різних ланок мережі суттєво відрізняється за наслідками для роботи мережі.

Будемо розглядати атаку мережі з одним користувачем. Динаміку процесу атаки мережі запишемо у вигляді рівняння (позначення матриць відповідають рівнянню (4)):

$$\dot{\bar{q}}(t) = \bar{\lambda}(t) + \bar{\alpha} + B\bar{u}(t) - \bar{l}(t), \qquad (10)$$

де $\bar{\alpha} = (\alpha_1, ..., \alpha_3)$ – трафік атаки.

Будемо аналізувати атаки вигляду $(0, \alpha_2, \alpha_3)$ системи (10).

Твердження 4. Атака вигляду $(0, \alpha_2, 0)$ призведе до нового мінімального часу закінчення завантаження

$$T(\alpha) = \max\{T_1, T_2 + T(\mu_2, \alpha_2)\},$$

де

$$T(\mu, \alpha) = \frac{\alpha}{\min(\mu, \lambda^{\max})\lambda^{\max}}.$$

Доведення. Оскільки ресурси діляться пропорційно до потоків користувачів, то з твердження 3 випливає, що для потоку α_2 ресурси другого вузла розподіляться таким чином, що час завантаження буде мати вигляд:

$$T_2 = \frac{\lambda^{\text{int}}(\lambda^{\max} + \alpha_2)}{\min(\mu_2, \lambda^{\max})\lambda^{\max}}.$$

Оскільки мінімальний час визначається як максимум з T_1 та T_2, то твердження доведене.

Таким чином атака сервера збільшує загальний час не більш як лінійно. Більш складним і небезпечним випадком являється АСК атака [5].

Твердження 5. Якщо $\alpha_3 < \mu_3$, то атака вигляду $(0, 0, \alpha_3)$ призведе до нового мінімального часу закінчення завантаження $T(\alpha) = \max\{T_2, T(\mu_1, \mu_3, \alpha_3)\}$, де

$$T(\mu_1, \mu_3, \alpha_3) = \lambda^{\text{int}}\left(\min\left(\mu_1 - \frac{\mu_1\alpha_3}{\mu_3}, \lambda^{\max}\right)\right)^{-1} + \lambda^{\text{int}}\left(\min(\mu_3 - \alpha_3, \lambda^{\max})\right)^{-1}.$$

Якщо $\alpha_3 \geq \mu_3$, то $T(\alpha) = +\infty$.

Доведення. Черга $q_3(t)$ має пріоритет, тому трафік атаки α_3 займає ресурси усього вузла. Якщо $\alpha_3 < \mu_3$, то це означає, що для використання вільна тільки частина ресурсів – $\mu_3 - \alpha_3$. Ресурси, доступні для використання на першому вузлі задовольняють нерівності

$$\frac{x}{\mu_1} + \frac{\alpha_3}{\mu_3} \leq 1.$$

Максимально можливий обсяг дорівнює $\mu_1 - \mu_1\frac{\alpha_3}{\mu_3}$. Тоді час, необхідний для проходження всіх пакетів λ^{int} через перший вузол складається з часу необхідного для проходження всіх пакетів через першу чергу

$$\lambda^{\text{int}}\left(\min(\mu_1 - \frac{\mu_1\alpha_3}{\mu_3}, \lambda^{\max})\right)^{-1}$$

плюс час, необхідний для проходження через другу чергу

$$\lambda^{\text{int}}\left(\min(\mu_3 - \alpha_3, \lambda^{\max})\right)^{-1}.$$

Якщо $\alpha_3 \geq \mu_3$, то всі ресурси першого вузла витрачаються на обробку трафіка атаки, тобто $T(\alpha) = +\infty$.

Висновки

У даній роботі розглянуто підхід до моделювання керування потоками даних у мережах. Здійснено огляд основних робіт з даного напрямку, розглянуто розвиток алгоритмів керування завантаженням у мережі Інтернет. Описуються основні особливості сучасних мереж, що не враховуються в існуючих моделях – наявність змінних протоків даних і наявність конфліктуючих користувачів. Пропонується новий підхід до розробки потокових керованих моделей мереж. Побудована модель одного класу мереж та доведене існування оптимального у сенсі швидкодії розв'язку

задачі завантаження файла. Розглянуто також ситуацію зловмисного впливу на роботу мережі (атаки на відмову). Для моделі за умов поглинаючої атаки знайдено залежність між обсягом атаки та часом закінчення завантаження.

1. *Kelly F.P.* Charging and rate control for elastic traffic // European Trans. on Telecommunications. – 1997. – **8**. – P. 33 – 37.
2. *Paganini F., Doyle J.C., Low S.H.* Scalable laws for stable network congestion control // Proc. of IEEE Conference on Decision and Control. – 2001. – 1. – P. 185 – 190.
3. *Srikant R.* The Mathematics of Internet Congestion Control. – Springer Verlag, 2004. – 137 p.
4. *Afanasyev A., Tilley N., Reiher P., Kleinrock L.* Host-to-Host Congestion Control for TCP // IEEE Communications surveys & tutorials. – 2010. – 12 (3). – P. 304 – 342.
5. *Андон П.І., Ігнатенко О.П.* Атаки на відмову в мережі Інтернет: опис проблеми та підходів щодо її вирішення / Ін-т програмних систем. – Препр. – К.; 2008. – 50 с.
6. http://tools.ietf.org/pdf/rfc896.pdf
7. http://rfc2.ru/793.rfc
8. *Welzl M.* Network Congestion Control: Managing Internet Traffic. Wiley. – 2005. – 263 p.
9. http://tools.ietf.org/html/rfc2309
10. *Low S. Paganini F., Doyle J.* Internet congestion control // IEEE Control Syst. Mag. – 2002. – **22** (1). – P. 28–43.
11. *Low S.H., Srikant R.* A Mathematical Framework for Designing a Low-Loss, Low-Delay Internet // Network and Spatial Economics. – 2004. – **4** (1). – P. 75 – 102.
12. *Nazarathy Y., Weiss G.* A Fluid Approach to Large Volume Job Shop Scheduling // J. of Scheduling. – 2010. – **13** (5). – P. 509 – 529.
13. *Meyn S.* Control Techniques for Complex Networks. – Cambridge University Press. – 2007. – 582 p.
14. *Филиппов А.Ф.* Дифференциальные уравнения с разрывной правой частью. – М.: Наука, 1985. – 224 с.

Отримано 10.05.2011

Про автора:

Ігнатенко Олексій Петрович,
кандидат фізико-математичних наук,
старший науковий співробітник,
докторант.

Місце роботи автора:

Інститут програмних систем
НАН України,
03187, Київ-187,
проспект Академіка Глушкова, 40.
Тел.: 526 6025.
e-mail: o.ignatenko@gmail.com