УДК 004.056.55:004.75

А.К. Гіранова, аспірантка, ІПМЕ ім. Г.Є. Пухова НАНУ, м. Київ

ДЕЯКІ ПИТАННЯ ЗАСТОСУВАННЯ КЛАСТЕРНИХ ОБЧИСЛЮВАЛЬНИХ СИСТЕМ ДЛЯ ВИРІШЕННЯ ЗАДАЧ ЗАКРИТТЯ ІНФОРМАЦІЇ

Розвиток обчислювальної техніки, комп'ютерних і мережевих технологій в даний час призвів до значного збільшення обсягів даних, що обробляються, передаються та зберігаються в електронному вигляді. При цьому спостерігається тенденція відчуження (фізичного і логічного віддалення) користувача від використовуваних інформаційних ресурсів. Як наслідок, все більшої актуальності набувають питання захисту інформації, їх значущість неухильно зростає.

При передачі в розподіленій мережі великих обсягів даних, що вимагають ресурсномісткої операції шифрування, виникає задача розпаралелювання обчислювального процесу.

В даній роботі було виконано дослідження особливостей, які виникають при виконанні ресурсномістких операцій шифрування великих обсягів даних, що передаються в розподіленій мережі кластерних багатопроцесорних систем. Також розглянуті деякі питання створення паралельних програм із застосуванням відповідного інструментального забезпечення.

Аналіз останніх досягнень і публікацій свідчить про наявність великої кількості досліджень і розробок, направлених на рішення складних ресурсномістких задач на обчислювальних кластерах. Розпаралелювання процесу криптографічного перетворення не потрапляє до кола таких задач в силу природного паралелізму за даними. Тим не менш, при шифруванні інтенсивних потоків інформації, зокрема, в грід-середовищах виникають деякі особливості, які вимагають окремого дослідження.

Метою даної статті є дослідження можливостей прискореної реалізації задач закриття інформації на кластерних багатопроцесорних обчислювальних системах.

1. Застосування стандарту паралельного програмування MPI

В даний час при створенні паралельного програмного забезпечення, орієнтованого на роботу в грід-середовищі, стандартом де-факто є технологія MPI (message passing interface – інтерфейс передачі даних), що забезпечує розподіл обчислювального навантаження й організацію інформаційної взаємодії між процесорами кластера [1-3].

При практичному використанні MPI розробникам необхідно звернути увагу в першу чергу на наступні особливості:

– MPI – це бібліотека, а не мова. Вона визначає імена, виклики процедур і результати їх роботи. Програми, написані на мовах FORTRAN, C, і C++,

компілюються звичайними компіляторами і зв'язуються з MPI-бібліотекою в процесі збирання (лінковки).

– MPI – це опис, а не реалізація. Коректно написана MPI-програма має бути сумісна з будь-якими програмними інструментальними засобами, які підтримують MPI, без додаткових змін вихідних текстів.

Інтерфейс MPI, теоретично, підтримує створення паралельних програм в стилі MPMD (Multiple Program Multiple Data), що має на увазі об'єднання процесів з різними вихідними текстами. Проте, розробляти і налагоджувати такі програми дуже складно. Тому, на практиці програмісти, як правило, використовують паралельне програмування в стилі SPMD (Single Program Multiple Data). В цьому випадку вихідна задача розбивається на декілька підзадач, кожна з яких може бути вирішена єдиним методом стосовно різних фрагментів оброблюваних даних. Тоді паралельна програма складається з однакових фрагментів і, фактично, на всіх процесорах один і той же код реалізує однакові підзадачі. Така схема і називається SPMD-моделлю.

2. Умови проведення експериментів

Експерименти проводилися на обчислювальному кластері ІПМЕ ім. Г.Є.Пухова НАН України в середовищі UNIX-подібної операційної системи CentOS ver.5.2.

Всі програми написані на мові С. Як інструментальний засіб використовувався компілятор Intel C++ Compiler ver.11.0, що входить в універсальний пакет Intel Cluster Toolkit Compiler Edition for Linux ver.3.2.

Як засіб паралельної обробки даних використовувалася бібліотека Intel MPI Library ver.3.2, що також є компонентом вищезазначеного пакету. До складу даної бібліотеки входять декілька різних командних файлів, які використовуються для збору MPI додатків, які є, фактично, надбудовами над наявними компіляторами, і дозволяють управляти вибором необхідного інструментального засобу залежно від мови програмування і розрядності додатку.

Використання компілятора Intel при виконанні даної роботи виявило ряд переваг даного інструментального засобу в порівнянні із застосуванням відкритого програмного забезпечення. Крім отримання ефективнішого коду, цей компілятор має розширені сервісні можливості, що полегшують користувачам розробку програм. Так, наприклад, при виникненні проблеми з компіляцією вбудованого в текст програми асемблерного коду, компілятор Intel видав повідомлення про необхідність додати опцію -fasm-blocks. У тій же ситуації вільно поширюваний за ліцензією GNU компілятор gcc ver.4.1, окрім повідомлення про помилку не видав ніякої додаткової інформації.

Принципи організації обчислювального процесу у разі використання паралельної роботи трохи відрізняються від традиційного порядку запуску послідовних однопроцесорних програм. Зокрема, крім підключення спеціальних паралельних бібліотек на етапі компіляції і запуску відповідних компонентів часу виконання (run-time) цих же бібліотек, на кластері необхідно забезпечити функціонування певних системних модулів, що забезпечують взає-

модію обчислювальних ядер один з одним шляхом низькорівневої реалізації механізму обміну повідомленнями. У разі використання компілятора Intel в якості таких модулів використовуються так звані демони MPD (multiple purpose daemon).

Запуск демонів Intel здійснюється командою:

mpdboot -n <#nodes> -f ~/mpd.hosts -r ssh,

де #nodes - кількість вузлів, на яких виконуватиметься MPI-додаток. За допомогою команд mpdtrace і mpdallexit можна відповідно проглянути конфігурацію MPD і зупинити демони, що виконуються. Для нормальної роботи механізму MPD в домашньому каталозі користувача необхідно створити файл mpd.hosts, що містить список наявних обчислювальних вузлів.

Власне запуск експериментальних програм, відповідно до вимог до будь-якого додатку, виконуваного на кластері, як на ресурсі колективного користування, що розділяється, здійснювався із застосуванням спеціальної системи управління пакетною обробкою (СУПО). На кластері ІПМЕ ім. Г.Є.Пухова НАН України в якості такого засобу використовується вільно поширюваний менеджер черг TORQUE ver.2.3.3.

3. Результати обчислювальних експериментів

В ході проведення експериментів в якості процедури шифрування використовувався блочний алгоритм шифрування ГОСТ 28147-89 в режимі простої заміни. Блок-схема розпаралелювання оброблювальної процедури приведена на рис. 1.

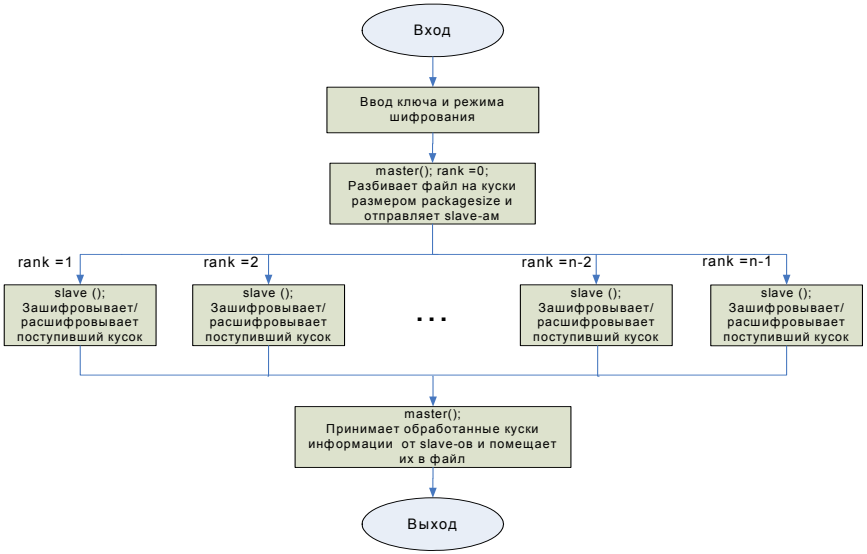

Рис. 1. Залежність продуктивності програми від кількості задіяних процесорних ядер

Екземпляр паралельної MPI-програми, що має ранг 0, виступає в ролі процесу, що управляє (master). Він виконує допоміжні операції і здійснює обмін з іншими процесами (slave). Зокрема, майстер-процес розбиває вхідні файли, що імітують потік тих даних, що поступають на вхід системи, на блоки, які можуть бути оброблені обчислювальними процесами незалежно один від одного. Нульовий процес також відповідає за прийом оброблених сегментів даних і видачу їх в потрібній послідовності у вихідний потік

На рисунку 2 приведені результати обчислювальних експериментів, що відображають залежність продуктивності обчислювального процесу в цілому від кількості задіяних апаратних ресурсів. Тут по осі абсцис відкладено число процесорних ядер, а по осі ординат – досягнута продуктивність в байт/сек. Різні криві відповідають різним розмірам вхідних файлів. Експерименти проводилися з масивами даних завдовжки 512 Мб, 1 Гб і 2 Гб.

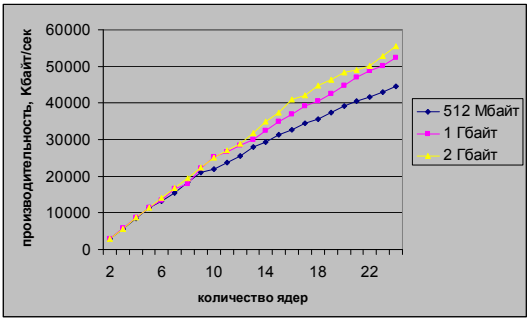

Рис. 2. Залежність продуктивності програми від кількості задіяних процесорних ядер

Завдяки властивому вирішуваній задачі природному паралелізму досягнута практично лінійна залежність швидкості обробки даних від числа процесорів.

Цікавіші результати отримані в ході проведення другої групи експериментів. На рисунках 3 – 6 приведені залежності часу виконання обчислювального алгоритму від розміру блоку обміну між процесорами для широкого діапазону значень вхідних файлів.

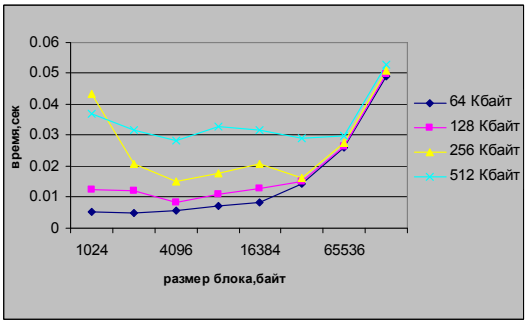

Рис. 3. Залежність часу обробки від розміру блоку обміну даними для файлів завдовжки від 64 Кб до 512 Кб

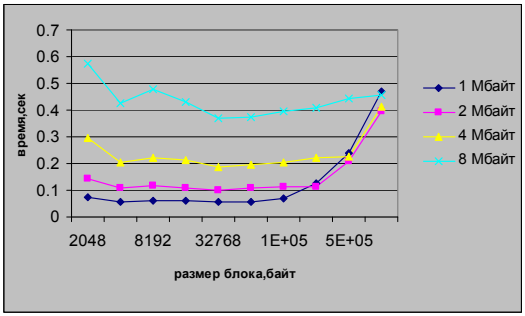

Рис. 4. Залежність часу обробки від розміру блоку обміну даними для файлів завдовжки від 1 Мб до 8 Мб

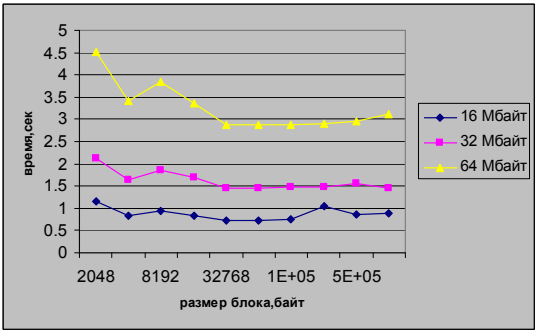

Рис. 5. Залежність часу обробки від розміру блоку обміну даними для файлів завдовжки від 16 Мб до 64 Мб

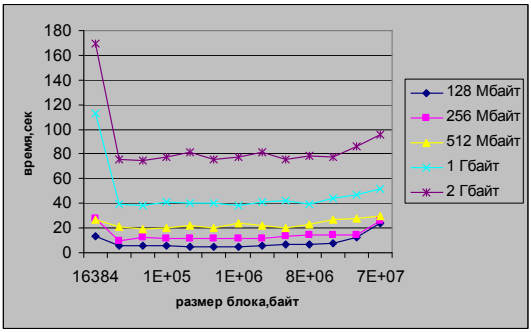

Рис. 6. Залежність часу обробки від розміру блоку обміну даними для файлів завдовжки від 128 Мб до 2 Гб

Як видно з графіків, значення даного параметра істотно впливає на швидкість обчислень. Величина блоку, на який розбиваються вхідні дані, має деякий діапазон значень, при виході за межі якого ефективність обчислень різко знижується. Причому, цей діапазон тим ширше, чим інтенсивніше потік

оброблюваних даних. Наслідком даного ефекту є той факт, що для певного діапазону розмірів вхідних файлів, починаючи з деякого значення, може бути заданий один і той же параметр, що є оптимальним в сенсі скорочення часу обчислень.

Висновки

В результаті проведеного дослідження може бути зроблено висновок про те, що процедура шифрування, що реалізовується на кластерах грід-мережі із застосуванням технології MPI, добре розпаралелюється і дозволяє досягти високих показників ефективності для широкого діапазону значень інтенсивності потоку даних без зміни основних параметрів обчислювального процесу.

Як результат даної статті можна також розглядати описаний досвід створення паралельних програм із застосуванням відповідного інструментального забезпечення, який може виявитися корисним для дослідників у різних наукових галузях.

1. *Антонов А.С.* Параллельное программирование с использованием технологии MPI: Учебное пособие. – М.: Изд-во МГУ, 2004. – 71 с.
2. *Шпаковский Г.И., Серикова Н.В.* Программирование для многопроцессорных систем в стандарте MPI. - Минск: Изд-во БГУ, 2002. – 323 с.
3. *Немнюгин С.А., Стесик О.Л.* Параллельное программирование для многопроцессорных вычислительных систем. – СПб.: БХВ-Петербург, 2002. – 400 с.

Поступила 22.02.2010р.

УДК 621.3

С.О.Нікулін

МЕТОДИ ФОРМУВАННЯ ОСНОВНИХ ІНФОРМАЦІЙНИХ ЗАСОБІВ СИСТЕМИ ЗАХИСТУ

Важливою компонентою довільної системи захисту являються дані про загрози. Останні представляють собою певні властивості об’єкту, що охороняється, який в даному випадку є системою управління спеціалізованим об’єктом. Виходячи з загально прийнятої інтерпретації поняття загрози [1] остання є негативною характеристикою системи. Тому при проектуванні системи передбачається не допускати в рамках останньої існування загроз. Практично, загрозами в інформаційній системі являються такі способи реалізації програмних засобів, які дозволяють або надають можливість використовувати відповідні фрагменти для несанкціонованого втручання в

