Высокопроизводительные вычисления как облачный сервис: ключевые проблемы*

А.О. Кудрявцев, В.К. Кошелев, А.О. Избышев, А.И. Аветисян

Институт системного программирования Российской академии наук (ИСП РАН)

В настоящее время облачные технологии получают все большее распространение. При этом возможности применения концепции облачных сервисов в области высокопроизводительных вычислений существенно ограничены. Ключевая проблема — накладные расходы, возникающие при использовании технологий виртуализации. В данной работе рассматриваются ключевые источники накладных расходов при виртуализации высокопроизводительного кластера. Приводятся результаты исследований накладных расходов и методы повышения производительности параллельных приложений в виртуальных машинах. Подробно рассматривается проблема виртуализации высокопроизводительного ввода-вывода на примере сети Infiniband.

1. Введение

В настоящее время облачные технологии получают все большее распространение. Облачные системы широко применяются на индустриальном уровне, в науке и образовании. При этом возможности применения концепции облачных сервисов в области высокопроизводительных вычислений существенно ограничены. Ключевая проблема — накладные расходы, возникающие при использовании технологий виртуализации. Технологии виртуализации играют ключевую роль в реализации облачных систем, обеспечивая возможность предоставления вычислительных ресурсов по запросу.

Возможности технологий виртуализации платформы x86_64 в последние годы растут высокими темпами. Технологии аппаратной и контейнерной виртуализации уже позволяют обеспечить производительность отдельных классов приложений в виртуальных средах, практически не отличимую от производительности на реальном оборудовании. В результате исследователи по всему миру начали изучать возможности и ограничения виртуализации высокопроизводительных вычислений (HPC, High Performance Computing).

Достоинства применения виртуализации в области высокопроизводительных вычислений широко обсуждаются [1–3]. Среди них устойчивость к сбоям, совместимость и гибкость при работе с виртуальными машинами (ВМ). Особенно интересна идея применения концепции облачных вычислений для создания высокопроизводительных облачных систем, предоставляющих масштабируемость, эффективность использования ресурсов и удобство доступа. Последние исследования показывают, что виртуализация HPC имеет смысл по крайней мере для отдельных классов приложений [1, 4].

Основной целью проводимых работ является оценка накладных расходов при виртуализации высокопроизводительных вычислений, а также выявление причин и минимизация этих расходов. Исследования проводятся с использованием системы виртуализации KVM (Kernel-based Virtual Machine) [5].

Для достижения наилучшей производительности (относительно случая запуска на реальном оборудовании) виртуальным машинам выделяется максимально возможное количество ресурсов, включая все процессорные ядра. Кроме того, виртуальным машинам предоставляется реальное коммуникационное устройство, с использованием технологий виртуализации ввода-вывода Intel VT-d [6]. При анализе производительности были обнаружены накладные расходы, связанные с использованием технологии Intel VT-d. В данной работе подробно рассматриваются причины этих накладных расходов и методы повышения про-

*Работа выполнена при финансовой поддержке РФФИ, грант № 12-07-00726-а.

изводительности.

В качестве тестового пакета используется пакет SPEC MPI2007 [7]. Данный пакет достаточно широко распространен в области высокопроизводительных вычислений. Исследование производительности выполнялось на современном кластере, с использованием до 16 узлов, построенных на базе процессоров Intel Xeon (в сумме до 192 процессорных ядер). Узлы кластера связаны высокоскоростной сетью Infiniband 40 Гбит/сек. Стоит отметить, что многие исследования в области виртуализации HPC, представленные ранее, проводились на системах из одного узла, что не позволяло полноценно оценить накладные расходы, вызываемые виртуализацией.

В ходе выполнения данной работы достигнуты следующие результаты:

- Выявлены основные источники накладных расходов, возникающих при виртуализации.
- Проведена оценка накладных расходов виртуализации высокопроизводительной системы ввода-вывода для современного HPC кластера, построенного на базе высокоскоростной сети Infiniband. Узлы кластера — двухпроцессорные 12-ядерные серверы, используется 16 узлов.
- Проанализированы полученные экспериментальные данные и накладные расходы, вызванные применением технологий виртуализации ввода-вывода Intel VT-d. В рамках ядра ОС Linux и системы виртуализации KVM/QEMU разработан и реализован механизм, позволяющий предоставлять реальное коммуникационное устройство доверенной виртуальной машине без применения технологии Intel VT-d. Данный механизм позволил существенно снизить накладные расходы на отдельных тестах не повлияв на другие.
- В целом, на рассматриваемом стенде достигнуто снижение накладных расходов на виртуализацию до уровня 1-10% на тестах пакета SPEC MPI2007.

2. Ключевые источники накладных расходов при виртуализации

Можно выделить несколько ключевых источников накладных расходов, возникающих при виртуализации. Часть проблем связана с использованием технологий аппаратной виртуализации процессора, памяти, устройств. В настоящее время считается, что вопросы виртуализации процессора тщательно изучены, и различные подходы к реализации гипервизоров позволяют достичь производительности, близкой к производительности реального процессора при запуске приложения, требовательного к вычислительной мощности.

При виртуализации памяти стоит цель разделения и изоляции памяти виртуальных машин и основной системы (системы, в рамках которой выполняется гипервизор). При этом решается задача отображения гостевого виртуального адреса в реальный физический адрес. На платформе x86 применяется два подхода: теневая страничная трансляция (shadow paging), основанная на традиционном механизме пейджинга x86, и вложенная страничная трансляция (nested paging), основанная на аппаратных расширениях виртуализации памяти (более подробно см. [3]). Метод вложенной трансляции дает лучшие результаты для гостевых систем с частым переключением таблиц страниц, например, для ОС Linux. В нашей работе мы также использовали вложенную трансляцию, поскольку в качестве гостевой ОС используется Linux.

В рамках виртуализации HPC-вычислений, необходимо предоставлять виртуальной машине все доступные на узле ресурсы. С этой точки зрения актуальным становится соответствие архитектуры основной системы и гостевой системы. Многие современные серверы имеют архитектуру NUMA (Non-Uniform Memory Access, архитектура с неравномерным

доступом к памяти), при наличии двух и более процессоров. Проблемы виртуализации таких серверов рассматривались нами в работе [4]. В системе виртуализации KVM/QEMU не предусмотрена возможность полносистемной эмуляции архитектуры NUMA в соответствии с реальной топологией узла, такая возможность была реализована нами ранее и используется в данной работе. Корректная эмуляция архитектуры NUMA позволила снизить накладные расходы с 50-60% до 1-10% на многих тестах.

Отдельной проблемой, особенно актуальной при виртуализации HPC, является виртуализация высокопроизводительного ввода-вывода. Данная проблема подробно рассматривается в разделе 3.

При увеличении масштаба вычислений (числа задействованных узлов в кластере), начинают влиять незначительные факторы отклонения производительности, вызванные работой основной ОС, гипервизора и гостевой ОС. Такие факторы принято называть шумом. Среди таких факторов — работа программ-демонов, обработчиков прерываний, других программ в системе. Влияние шума на производительность приложений широко изучается [8]. Для ряда приложений даже минимальный уровень шума может привести к значительному ухудшению производительности и масштабируемости. Таким образом, при виртуализации высокопроизводительной системы, особенно с большим числом узлов, необходимо учитывать дополнительный шум гипервизора и основной ОС.

3. Проблема виртуализации ввода-вывода

Существенной проблемой при виртуализации HPC-задач является виртуализация высокопроизводительных систем ввода-вывода, в частности, коммуникационных сред. В настоящее время для эффективной работы устройства необходимо предоставлять контроль над этим устройством гостевой системе (при серверной виртуализации все устройства в виртуальной машине, как правило, реализованы программно в гипервизоре).

На платформе x86 для проброса устройств возможно применение устройства аппаратной виртуализации ввода-вывода (IOMMU, I/O Memory Management Unit). Основной задачей IOMMU является отображение адресного пространства пробрасываемого устройства в адресное пространство физической памяти гостевой системы. При отсутствии такого устройства, гостевая система должна быть осведомлена о наличии гипервизора и положении гостевой физической памяти в реальной физической памяти, что позволит выполнять запросы на прямой доступ к памяти (DMA, Direct Memory Access) корректно.

Отдельной проблемой виртуализации устройств являются накладные расходы на обработку прерываний устройства, вызванные ограничениями технологий аппаратной виртуализации процессора. При каждом прерывании процессор должен передать управление от виртуальной машины гипервизору. Далее, гипервизор передает управление основной ОС, которая, в свою очередь, определяет, что прерывание принадлежит гипервизору. После этого, гипервизор производит «вброс» прерывания в гостевую систему. Эта цепочка вызовов может замедлить доставку прерываний и серьезно увеличить задержку коммуникаций.

В работе [9] описанное выше предположение подтверждается. Авторы предложили систему ELI (ExitLess Interrupts), которая позволяет обрабатывать часть прерываний в гостевой системе без необходимости выходов в гипервизор. Результаты тестов показали, что система ELI может уменьшить накладные расходы KVM/QEMU с 40% до 1-2% при работе ВМ на одном вычислительном ядре. В данном случае ухудшение производительности было связано с большим числом прерываний, генерируемых проброшенной внутрь ВМ сетевой картой Ethernet 10 Гбит/сек.

В рамках данной работы применялось коммуникационное устройство на базе сети Infiniband, при этом были выявлены накладные расходы, связанные с применением IOMMU. Описание проведенного исследования приведено в следующих подразделах.

3.1. Экспериментальная среда

В качестве стенда используется кластер фирмы HP, узлы HP ProLiant SL390s G7, используется до 16 узлов. Каждый узел имеет архитектуру NUMA, содержит по 2 процессора Intel Xeon X5650 (6 ядер на процессор) и 24 Гб ОЗУ. Взаимодействие узлов осуществляется с использованием 1 Гбит/сек сервисной сети Ethernet и 40 Гбит/сек коммуникационной сети Infiniband. Используются адаптеры Mellanox ConnectX PCIe 2.0.

В качестве основной ОС и гостевой ОС применяется дистрибутив CentOS 6.3 (версия ядра `2.6.32-279.5.2.el6.x86_64`). В качестве библиотеки MPI применяется MVAPICH версии `1.2.0`. Для сборки тестов применяется компилятор GCC версии `4.4.6`.

Гипервизор KVM встроен в ядро ОС Linux, его версия соответствует версии ядра ОС. Версия эмулятора QEMU — `1.0.1` с изменениями, необходимыми для корректной эмуляции архитектуры NUMA. Гостевая система запускается с 16 ГБ ОЗУ, все ядра процессоров назначены ВМ с учетом топологии NUMA. Адаптер Infiniband проброшен в ВМ с использованием технологии Intel VT-d. Память ВМ выделяется с использованием вложенной страничной адресации, страницами размером 2 МБ.

Для оценки накладных расходов на виртуализацию применяется тестовый пакет SPEC MPI2007 версии 2.0. Подробное описание тестов пакета приведено в документации по адресу `http://www.spec.org/mpi/Docs/`. Тесты запускаются по 10 раз.

3.2. Оценка накладных расходов

Результаты проведенных экспериментов представлены на Рисунке 1. Столбец «Native» соответствует конфигурации при запуске без гипервизора, «KVM» — базовой рассматриваемой конфигурации, столбец «KVM, no IOMMU» рассматривается в следующем разделе. Тонкие черные столбцы обозначают среднеквадратичное отклонение для соответствующих средних значений. Данные изображены относительно случая Native, которому соответствует среднее значение 0 в каждой группе столбцов. Каждая группа столбцов соответствует конкретному тесту пакета, название теста приведено под группой.

В целом, на большинстве тестов пакета падение производительности не превышает 4%. Существенные накладные расходы возникают в случае тестов leslie3d (8%) и pop2 (23%).

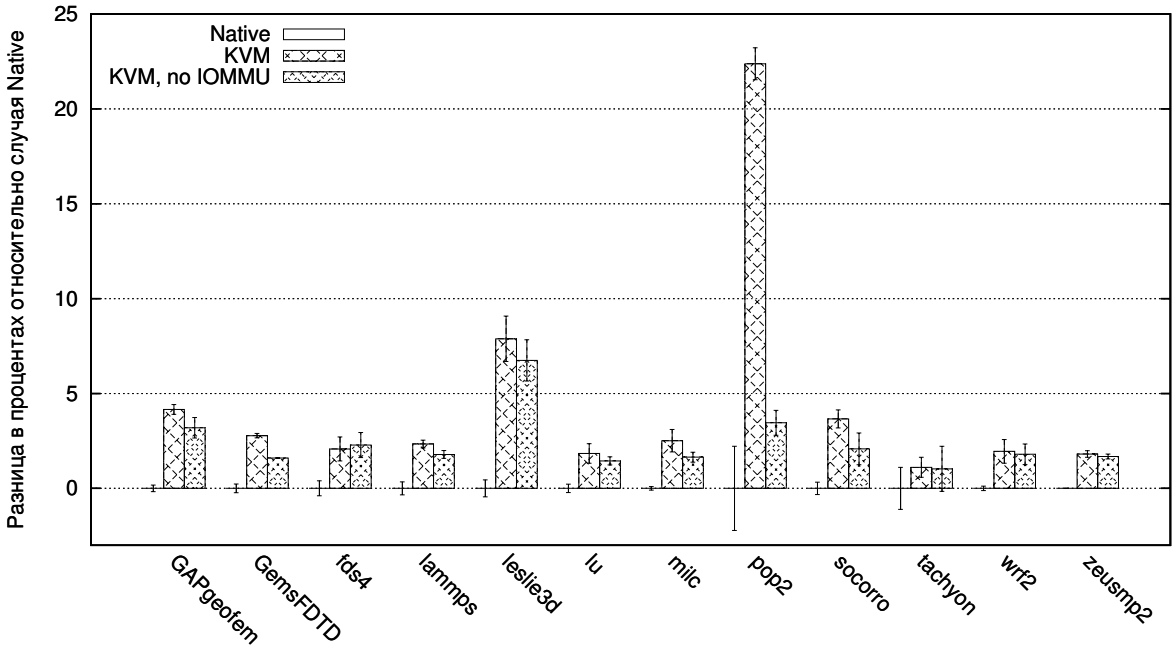

Рис. 1. Результаты экспериментов SPEC MPI2007 с использованием 192 ядер.

3.3. Метод обхода аппаратного IOMMU

В ходе анализа производительности теста pop2 было выдвинуто предположение, что накладные расходы вызваны работой устройства IOMMU. Основной операцией MPI, выполняемой в тесте pop2, является функция Allreduce с небольшим объемом посылаемых данных (8 байт). В ходе тестирования производительности синтетического теста, реализующего такой характер коммуникаций, было получено снижение производительности в ВМ в 2 раза.

Основная задача, которую выполняет IOMMU — трансляция адреса устройства в физический адрес ВМ при выполнении DMA-транзакций. При трансляции используется таблица страниц, предоставленная основной ОС по запросу гипервизора. Так же, как и в случае классической страничной трансляции, используется буфер трансляции, в котором сохраняются последние оттранслированные адреса. Данный буфер имеет ограниченный размер. В соответствии со спецификацией Intel VT-d, в таблице страниц могут поддерживаться различные размеры страниц (включая 4 КБ, 2 МБ, 1 ГБ). Размер страниц непосредственно влияет на объем участка памяти, трансляции адресов для которого могут одновременно находиться в буфере трансляции.

В ходе проведенного исследования было установлено, что в ядре ОС Linux дистрибутива CentOS 6.3 драйвер Intel VT-d не поддерживает размеры страниц, отличные от 4 КБ. При этом в основной ветке ядра Linux (см. `http://kernel.org`) такая поддержка есть начиная с версии 3.0. Несмотря на это, применение обновленного ядра показало отсутствие на тестовом стенде поддержки страниц размером более 4 КБ.

Для проверки предположения о влиянии IOMMU на производительность был разработан и реализован метод обхода аппаратного IOMMU.

Проброс устройства в ВМ без использования IOMMU осуществляется с использованием паравиртуального интерфейса между гостевой ОС и гипервизором. Взаимодействие организуется между драйвером устройства, установленным в гостевой ОС, и гипервизором. Архитектура системы приведена на Рисунке 2. В драйвер пробрасываемого устройства включается специальный модуль, обеспечивающий трансляцию адресов при выполнении DMA-запросов. При этом, как правило, не требуется значительных модификаций драйвера, за счет наличия стандартного API для работы с DMA в ядре Linux. При инициализации драйвера модуль устанавливает связь с гипервизором с использованием паравиртуального интерфейса, реализуемого компонентом обхода IOMMU, входящим в состав гипервизора. Модуль получает таблицу отображений гостевых физических адресов в реальные физические адреса, после чего с использованием этой таблицы осуществляется трансляция адресов DMA-запросов перед передачей команд устройству.

Реализация модуля обхода IOMMU как части драйвера позволяет избежать изменения кода гостевой ОС. Недостатком данного подхода является необходимость незначительной модификации драйвера. Описанный метод также имеет недостатки с точки зрения безопасности (возможно получение доступа к памяти основной системы путем выполнения DMA-транзакций проброшенного устройства), однако в случае HPC-вычислений зачастую гостевая система может считаться доверенной.

Описанный метод обхода IOMMU был реализован для ОС Linux и системы виртуализации KVM/QEMU. Результаты тестирования производительности приведены на Рисунке 1, столбец «KVM, no IOMMU». Применение описанного метода позволило снизить накладные расходы теста pop2 с 23% до 4%. При этом, для теста leslie3d уменьшение накладных расходов незначительно. Необходимо также отметить, что число прерываний, генерируемых адаптером Infiniband, невелико (около 3000 прерываний в секунду), в связи с чем применение подхода ExitLess Interrupts в данном случае не целесообразно.

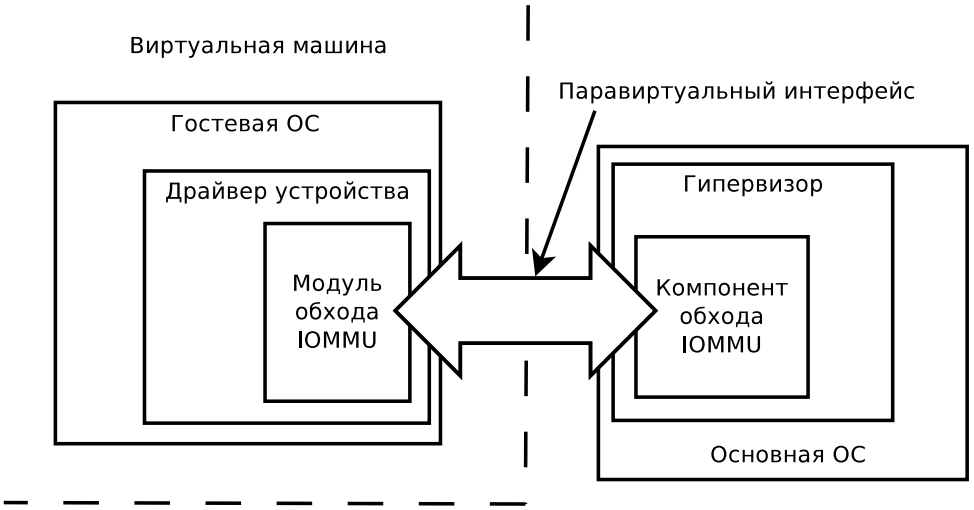

Рис. 2. Архитектура системы обхода IOMMU.

4. Заключение

Основным вкладом данной работы является демонстрация возможности эффективной виртуализации достаточно широкого класса HPC-приложений на Linux-кластерах небольшого масштаба. Для большинства тестов пакета SPEC MPI2007 накладные расходы на виртуализацию составили не более 5% при использовании 192 процессорных ядер (16 узлов). При этом были использованы наработки в области улучшения производительности гипервизора KVM/QEMU, в частности эмуляция архитектуры NUMA в соответствии с архитектурой реального сервера.

Продемонстрировано влияние аппаратной виртуализации ввода-вывода на производительность на примере адаптера Infiniband. В рамках ядра ОС Linux и системы виртуализации KVM/QEMU разработан и реализован механизм, позволяющий предоставлять реальное коммуникационное устройство доверенной виртуальной машине без применения технологии Intel VT-d. Данный механизм позволил существенно снизить накладные расходы на тесте pop2 (с 23% до 4%).

В рамках дальнейших работ планируется увеличение масштаба экспериментов до 1000 ядер и последующее изучение накладных расходов. При увеличении масштаба расчетов на производительность HPC-приложений начинает оказывать влияние шум как гостевой ОС, так и основной.

Литература

1. Younge A.J., Henschel R., Brown J., G. von Laszewski, Qiu J., Fox G.C. Analysis of Virtualization Technologies for High Performance Computing Environments // 4th International Conference on Cloud Computing (IEEE CLOUD 2011), July 4-9, 2011, Washington DC, USA.

2. A. Gavrilovska, S. Kumar, H. Raj, K. Schwan, V. Gupta, R. Nathuji, R. Niranjan, A. Ranadive, and P. Saraiya. Abstract High-Performance Hypervisor Architectures: Virtualization in HPC Systems // 1st Workshop on System-level Virtualization for High Performance Computing, (HPCVirt), in conjunction with EuroSys 2007, March 20, Lisbon, Portugal.

3. J. R. Lange, K. Pedretti, P. Dinda, P. G. Bridges, C. Bae, P. Soltero, and A. Merritt. Minimal-overhead virtualization of a large scale supercomputer // 7th ACM

SIGPLAN/SIGOPS international conference on Virtual execution environments, VEE '11, March 09-11, 2011, Newport Beach, CA, USA, Proceedings. ACM New York. 2011. P. 169–180.

4. А. О. Кудрявцев, В. К. Кошелев, А. И. Аветисян. Перспективы виртуализации высокопроизводительных систем архитектуры x64 // Труды Института системного программирования РАН, том 22, 2012. Стр. 189-209.

5. A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. KVM: the Linux virtual machine monitor // OLS '07: The 2007 Ottawa Linux Symposium, July, 2007, P. 225–230.

6. D. Abramson, J. Jackson, S. Muthrasanallur, G. Neiger, G. Regnier, R. Sankaran, I. Schoinas, R. Uhlig, B. Vembu, J. Wiegert. Intel® Virtualization Technology for Directed I/O // Intel Technology Journal.
URL: http://www.intel.com/technology/itj/2006/v10i3/2-io/1-abstract.htm (дата обращения: 22.11.2012).

7. M. S. Muller, M. van Waveren, R. Lieberman, B. Whitney, H. Saito, K. Kumaran, J. Baron, W. C. Brantley, C. Parrott, T. Elken, H. Feng, C. Ponder. SPEC MPI2007—an application benchmark suite for parallel systems using MPI // Concurrency and Computation: Practice and Experience, February 2010, Vol. 22, N. 2, P. 191–205.

8. K. Ferreira, P. Bridges, R. Brightwell. Characterizing application sensitivity to OS interference using kernel-level noise injection // International Conference for High Performance Computing, Networking, Storage and Analysis, November 2008, P. 1-12.

9. A. Gordon, N. Amit, N. Har'El, M. Ben-Yehuda, A. Landau, A. Schuster, D. Tsafrir. ELI: Bare-Metal Performance for I/O Virtualization // Seventeenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2012), March 03-07, 2012, London, UK, Proceedings. ACM New York. 2012. P. 411-422.