Динамическая оптимизация нагрузки на вычислительных узлах частных, публичных и гибридных облаков

А.С. Чадин <a.chadin@servionica.ru>
Сервионика, Россия, г. Москва, ул. Кедрова, 15

Аннотация. Данная система предназначена для автоматизированного распределения нагрузки в кластере путем анализа загруженности вычислительных узлов и последующей миграции виртуальных машин с загруженных узлов на менее загруженные. Помимо стабилизации нагрузки рассмотрена возможность снижения энергопотребления путем разгрузки слабонагруженных узлов и перевода их в ждущий режим. Стабилизация нагрузки в кластере ведет к повышению стабильности и сокращению времени выполнения запросов.

Ключевые слова: Балансировка; Кластер; Облачные вычисления

DOI: 10.15514/ISPRAS-2015-27(6)-19

Для цитирования: Чадин А.С. Динамическая оптимизация нагрузки на вычислительных узлах частных, публичных и гибридных облаков. Труды ИСП РАН, том 27, вып. 6, 2015 г., стр. 307-314. DOI: 10.15514/ISPRAS-2015-27(6)-19.

1. Введение

Система OpenStack [1], представляющая собой комплекс проектов свободного обеспечения для создания облачной инфраструктуры, имеет положительную репутацию в среде поставщиков и разработчиков облачных технологий. OpenStack не имеет на данный момент автоматическую систему балансировки виртуальных машин, что отрицательно сказывается на стабильности вычислительного кластера. Ввиду того, что в проекте OpenStack Nova уже созданы все необходимые инструменты для проведения ручной балансировки, предлагается автоматизировать процесс балансировки внутри вычислительного кластера.

Целью данной работы является разработка автоматизированной системы балансировки нагрузки в контексте системы OpenStack.

В рамках данной работы было проведено исследование и предоставлено одно из возможных решений построения сервиса автоматической балансировки для OpenStack. Решение является модульным и позволяет использовать различные подходы для реализации механизма балансировки. Предложенный в работе подход выполнен на основе среднеквадратичного отклонения.

2. Структура сервиса балансировки загруженности узлов

Разработанный сервис OpenStack Nova LoadBalancer позволяет автоматизировать балансировку нагрузки между узлами. Он собирает данные, анализирует их и принимает решение о миграции виртуальных машин с одного узла на другой. Сервис LoadBalancer является частью системы OpenStack Nova, что позволяет вызывать внутренние функции без использования внешнего API.

Рассмотрим структуру Nova LoadBalancer:

- Модуль сбора статистических данных
- Модуль анализа собранных данных и принятия решений.
- Модуль балансировки виртуальных машин (overload-алгоритм).
- Модуль определения недостаточности загруженности вычислительных узлов (underload-алгоритм).
- Вспомогательные службы (API, правила балансировщика).

Данные модули имеют базовые классы, что позволяет использовать собственные реализации классов.

Рассмотрим работу балансировщика поподробнее.

2.1 Сбор статистических данных

Для анализа требуется получать с каждого узла данные о загруженности вычислительных ресурсов. Показатель загруженности CPU узла получается с использованием Python-библиотеки psutil. Показатель свободной оперативной памяти узла вычисляется как сумма свободной, кэшируемой и буферной памяти. Сведения о памяти можно найти в файле meminfo директории /proc/.

Библиотека управления виртуализацией libvirt позволяет получать актуальные данные об использовании ресурсов виртуальной машиной. Сервис nova-compute был модифицирован таким образом, чтобы он с определенным интервалом отправлял данные по загруженности узла (процентная загруженность CPU, загруженность RAM в мегабайтах) и загруженности каждой виртуальной машины (процессорное время, загруженность RAM в мегабайтах), расположенной на нем, в сервис Nova Conductor, обеспечивающий взаимодействие с базой данных. Полученная информация записывается в таблицы «compute_node_stats» и «instances_stat». Данные о

загруженности виртуальных машин и узлов заполняются за счет использования атомарного метода UPSERT.

Схема сбора статистических данных и используемые компоненты OpenStack Nova представлена на рис. 1. Необходимо отметить, что расположение сервисов OpenStack Nova может различаться в зависимости от конфигурации.

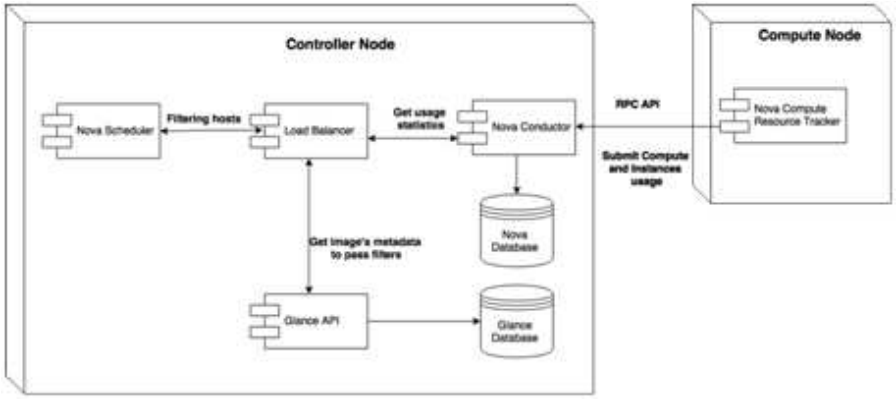

Рис.1. Диаграмма развёртывания сервиса OpenStack Nova LoadBalancer.

2.2 Анализ собранных данных и принятие решений

По показателям CPU и RAM в конфигурационном файле Nova устанавливаются пороговые значения среднеквадратичного отклонения. Данный модуль запускается с определенной в конфигурационном файле периодичностью.

Собранные данные необходимо преобразовать в процентные показатели. Следует нормализовать показатели в границах [0,1]. Для этого процессорное время переводится в загруженность CPU по следующей формуле:

$$CPU = \frac{(cpu_{time} - oldcpu_{time})}{(time_{current} - time_{old}) * vcpus} * 10^7,$$

где:

- cpu_time – процессорное время, затраченное виртуальной машиной для обработки процессов (задач). Выражено в наносекундах.
- oldcpu_time – процессорное время, затраченное виртуальной машиной в предыдущий момент времени (во время time_old).
- time_current – время, в которое был снят показатель cpu_time. Выражено в секундах.
- time_old – время, в которое был снят показатель oldcpu_time. Выражено в секундах.
- vcpus – количество ядер, выделенное данной виртуальной машине.

Загруженность RAM вычисляется как отношение занятой памяти к общему количеству памяти узла.

После обработки загруженности по всем узлам, вычисляется среднее арифметическое по CPU и RAM и среднеквадратичное отклонение. Полученные значения сравниваются с пороговыми значениями, указанными в конфигурационном файле, на основании чего делается вывод о необходимости стабилизации кластера. В случае, если полученные значения ниже пороговых, балансировщик проверит кластер underload-алгоритмом на энергоэффективность. В противном случае, запускается модуль балансировки виртуальных машин.

2.3 Балансировка виртуальных машин

Данный модуль позволяет создавать собственные реализации балансировки, либо воспользоваться реализацией на основе среднеквадратичного отклонения. Рассмотрим ее алгоритм поведения.

Балансировщик получает из Nova Conductor текущие значения загруженности всех узлов и их виртуальных машин. На основе полученных данных проводится симуляция перемещения каждой виртуальной машины на каждый узел (кроме исходного для виртуальной машины узла), в ходе которой меняются показатели загруженности для определения нового вероятного среднеквадратичного отклонения. Результаты симуляции записывается в список в виде словаря {узел, виртуальная машина, отклонение}. Данный список, по завершении всех симуляций, сортируется по возрастанию с ключом среднеквадратичного отклонения. Полученный список характеризует все возможные случаи перемещений виртуальных машин, из которых нас интересуют те, что сводят отклонение к минимуму. Взятая пара {узел, виртуальная машина} должна пройти фильтрацию сервисом Nova Scheduler. Фильтры сервиса Nova Scheduler позволяют определить, возможно ли запустить виртуальную машину на заданном узле. В случае, если один из фильтров не завершился успешно, пара {узел, виртуальная машина} пропускает дальнейшую фильтрацию и отбрасывается.

В случае, если пара прошла фильтрацию, вызывается метод *live-migration*, перемещающий виртуальную машину с исходного узла на целевой. В ходе миграции создается новый домен в libvirt, в который копируется содержимое оперативной памяти виртуальной машины. Когда копирование близко к завершению, гипервизор приостанавливает работу виртуальной машины, копирует оставшиеся данные и активирует домен виртуальной машины, возобновляя ее работу. По окончанию миграции, модуль балансировки прекращает свою работу до следующего вызова в случае перегрузки.

Если пара сервисом Nova Scheduler не прошла фильтрацию, из списка берется следующая пара.

Подобная реализация позволяет поэтапно стабилизировать нагрузку в кластере до тех пор, пока среднеквадратичное отклонение по узлам будет

ниже порогового значения. Изменение порогового значения СКО позволяет регулировать “чувствительность” балансировщика к перегрузкам.

2.4 Повышение энергоэффективности кластера

Существуют ситуации, когда не все вычислительные узлы кластера достаточно загружены. Т.к. вопрос энергопотребления для вычислительного центра стоит особенно остро, необходимо выработать методики сокращения энергопотребления. Предлагается следующая методика: определять слабонагруженные узлы, мигрировать с них виртуальные машины и приостанавливать работу узлов, помещая их в состояние ACPI S3 (“Ждущий режим”). Рассмотрим этапы поподробнее.

Как было замечено ранее, в случае, если балансировщик не определил перегрузки в кластере, будет проверена достаточность загруженности каждого вычислительного узла в кластере. Для этого устанавливаются пороговые значения в конфигурационном файле. Если показатели загруженности узла не превышают порог, принимается решение о необходимости введения узла в ждущий режим.

Клиент, подписывающий соглашение об уровне предоставления услуг (SLA) с поставщиком облачных решений, рассчитывает на определенный уровень доступности его виртуальных машин. Таким образом, перед переводом узла в ждущий режим, требуется комплекс мер по обеспечению сохранности доступа к виртуальным машинам.

Перевод узла в ждущий режим включает в себя следующие действия:

- Установка значения “suspending” полю “suspend_state” таблицы “compute_nodes” для узла, вводимого в ждущий режим.
- Попытка миграции всех виртуальных машин с узла. На данном этапе действует фильтр “max_migrations”, ограничивающий количество одновременных миграций с узла. Если не удается мигрировать одну из виртуальных машин в результате действия других фильтров, в поле “suspend_state” записывается значение “active” и перевод узла в ждущий режим прекращается. Это значит, что есть минимум одна виртуальная машина, для которой подходящее окружение находится только на текущем узле.
- Если все виртуальные машины были перемещены с “suspending”-узла, балансировщик запрашивает у узла MAC-адрес интерфейса, по которому будет возможно “разбудить” узел; записывает адрес в поле “mac_to_wake” таблицы “compute_nodes” и ставит флаг g параметру wol сетевого интерфейса. После данной подготовки, узлу посылается команда “systemctl suspend” (для RHEL/CentOS), что переводит узел в состояние ACPI S3. Узел в базе данных Nova помечается как suspended.

Для корректной работы необходим также suspend-фильтр, не разрешающий балансировщику взаимодействовать с узлами, которые находятся в состояниях suspended или suspending.

2.5 Управление балансировкой

Ввиду того, что в вычислительный кластер зачастую входят узлы, конфигурации которых разнятся, запускать балансировку на определенных узлах может быть излишним. OpenStack позволяет объединять узлы в различные группы по функциональным и логическим признакам [2]. Группировка по функциональным признакам «Host Aggregates» позволяет разграничить узлы в соответствии с ресурсными различиями. Группировка по логическим признакам «Availability Zones» позволяет физически разграничить узлы по местоположению в вычислительных центрах. Предлагается ввести систему правил в сервис балансировки для определения набора хостов, на которых можно разрешить или запретить его работу.

Правило является словарем вида {тип, регулярное выражение, действие}, где:

- “Тип” – логическая единица, к которой применяется правило. Допустимые значения: «host», «az», «ha».
- “Регулярное выражение” – шаблон, по которому проверяется принадлежность логической единицы к правилу.
- “Действие” – разрешить или запретить балансирование данной логической единицы.

Правила “читаются” системой в порядке следования, формируя в конце список из разрешенных узлов. Узлы из полученного списка участвуют в балансировке.

3. Экспериментальные результаты

В результате лабораторного тестирования были взяты замеры среднеквадратичного отклонения загруженности трех вычислительных узлов, между которыми балансировались виртуальные машины. Граница отклонения установлена на отметке 7%. Результаты взятия до и после включения балансирования представлены на рис. 2.

Рис. 2. Среднеквадратичное отклонение загруженности узлов.

Резкое снижение отклонения с 13.23% до 5.92% свидетельствует о перемещении той виртуальной машины, которая сводит отклонение к минимуму. Последующие колебания являются следствием изменений нагрузки на узлах.

Девид Мейзнер [3] утверждает, что энергопотребление блейд-сервера при полной мощности составляет 450 Вт, при средней загруженности 270 Вт и в ждущем режиме 10.4 Вт. Перевод двух узлов в ждущий режим в кластере, состоящем из 10 средненагруженных вычислительных узлов, позволит достичь 19.23% снижения энергопотребления.

4. Заключение

Разработанный сервис OpenStack Nova LoadBalancer показал практический результат в балансировании загруженности кластера. Архитектура сервиса построена таким образом, чтобы для решения задач была возможность использовать различные средства. Полученный опыт и знания позволят разрабатывать новые теоретические и практические подходы в решении задачи балансировки. Страница [4] сервиса на Launchpad позволит следить за ходом разработки и стать участником проекта.

Список литературы

[1]. OpenStack Overview. http://www.openstack.org/software/
[2]. OpenStack Scailing. http://docs.openstack.org/openstack-ops/content/scaling.html
[3]. Meisner D, Gold B, Wenisch T. PowerNap: eliminating server idle power. ACM SIGPLAN Notices 2009; 44(3): 205–216.
[4]. OpenStack Compute Load Balancer. https://launchpad.net/nova-loadbalancer

Dynamic Optimization of Workload on Compute Nodes in Private, Public and Hybrid Clouds

A. Chadin, <a.chadin@servionica.ru>
Servionica, 15 Kedrova Str., Moscow, Russian Federation

Abstract. Cluster's overload is a complex problem and can be solved by using one of stabilization methods: by migrating VMs from one node to another in case of compute nodes. The main goal of this research is to create nova load-balancer service that manages cluster loads and makes decisions of stabilization based on statistics data. This system is designed for automatic workload distribution in a cluster by analyzing workload of compute nodes and migrating VMs from overloaded nodes to underloaded ones. It runs periodically and checks latest statistics to compare with threshold values. If one of the statistics values is greater than the threshold value, the nova load-balancer runs cluster stabilization process to reduce load on cluster by performing live-migration of VMs. Besides workload stabilization, the system also provides an opportunity to reduce power consumption by unloading and suspending underloaded nodes. If statistics values of node are less than underload-threshold values, then all VMs of current node will be migrated to other nodes. When there are no VMs on the node, LB marks this node as suspended and is going to suspended mode (ACPI S3). If nova load-balancer wants to activate suspended nodes to reduce overall cluster load, it uses WOL technology. Workload stabilization in a cluster increases stability and reduces system response time.

Keywords: Balancing; Cluster; Cloud Computing

DOI: 10.15514/ISPRAS-2015-27(6)-19

For citation: Chadin A. Dynamic Optimization of Workload on Compute Nodes in Private, Public and Hybrid Clouds. Trudy ISP RAN/Proc. ISP RAS, vol. 27, issue 6, 2015, pp. 307-314 (in Russian). DOI: 10.15514/ISPRAS-2015-27(6)-19

References

[1]. OpenStack Overview. http://www.openstack.org/software/
[2]. OpenStack Scailing. http://docs.openstack.org/openstack-ops/content/scaling.html
[3]. Meisner D, Gold B, Wenisch T. PowerNap: eliminating server idle power. ACM SIGPLAN Notices 2009; 44(3): 205–216.
[4]. OpenStack Compute Load Balancer. https://launchpad.net/nova-loadbalancer