КЛАСТЕРНЫЕ ВЫЧИСЛЕНИЯ КАК СЕРВИС НА ПРИМЕРЕ ЗАДАЧИ МОДЕЛИРОВАНИЯ ТЕПЛОВЫХ ПОЛЕЙ ОТ СКВАЖИН НА СЕВЕРНЫХ НЕФТЕГАЗОВЫХ МЕСТОРОЖДЕНИЯХ

А.Ю. Берсенев, Н.А. Ваганова ,П.А. Васев, А.С. Игумнов, М.Ю. Филимонов

Институт математики и механики УрО РАН

Введение

Первое десятилетие XXI века ознаменовалось развитием общедоступных высокоскоростных компьютерных сетей, а вместе с ним концепцией удаленного предоставления вычислительных услуг без доступа к оборудованию и программам, которые на нём выполняются. Существуют различные степени абстрагирования пользователя от среды, в которой предоставляется услуга: от виртуальных машин, на которых можно запускать произвольное программное обеспечение; через преконфигурированые программы, такие как веб-хостинги; к чистым сервисам, таким как электронная почта Gmail или программа восстановления 3D модели по набору фотографий Autodesk 123D Catch.

Суперкомпьютерные вычисления стоят несколько в стороне от этой концепции в связи со спецификой решаемых задач, особенностями оборудования и спецификой доступа к вычислительным ресурсам. Специфика решаемых задач предполагает, что у суперкомпьютера не может быть массовой аудитории потребителей (массовым задачам соответствуют облачные вычисления). Особенности оборудования приводят к необходимости предоставления пользователям терминального доступа к среде разработки/запуска/отладки. Специфика доступа заключается в строгой персонификации пользователей (уровень отдельных кластеров) и организаций (ГРИД-системы). Большая часть работ, посвященных веб-интерфейсам вычислительных кластеров, сосредотачивается на замене интерфейса командной строки конечного пользователя на более привычный графический интерфейс [1, 2], а также на интеграцию суперкомпьютерных компонентов в классические облачные сервисы [3].

В данной работе предлагается подход, позволяющий разработчику при небольших затратах организовать доступ к своей суперкомпьютерной программе как к сервису. Данный подход учитывает традиционную инфраструктуру доступа к вычислительным ресурсам и существующие организационно-коммерческие отношения между пользователем и суперкомпьютерным центром.

Предполагается, что задача достаточно велика, чтобы её выполнение заняло многие часы и дни, но не настолько, чтобы её запуск можно было бы отложить до того момента, когда её разработчик получит от заказчика входные данные и запустит её вручную.

Работа выполнена при поддержке программы Президиума РАН № 18, проекта УрО РАН 12-П-1-1034, программы УрО РАН "Арктика" 12-1-4-005.

Организационные принципы прохождения задач

Предлагается следующая организационная модель запуска задач (Рис.1):

Владелец вычислительного ресурса (Владелец) заключает договор с разработчиком программы (Разработчик). Данный договор определяет стоимость единицы вычислений (например, ядро процессора на час), а также может ограничивать класс задач для запуска, объёмы доступного дискового пространства и т.п. С точки зрения Владельца его единственным клиентом является Разработчик, имеющий доступ через типовой интерфейс (чаще всего ssh) и запускающий известную задачу через типовую систему запуска (Slurm, СУППЗ и т.п.).

Разработчик подготавливает и компилирует программу, не требующую диалога с пользователем, и, при необходимости, набор вспомогательных программ/скриптов для подготовки входных данных и запуска задачи в соответствии с правилами, действующими на кластере.

Разработчик просит Владельца ограничить ему терминальный доступ входом по ключу и специализированным ограниченным интерпретатором команд.

Разработчик на стороннем ресурсе устанавливает веб-интерфейс запуска своей программы, снабжённый собственной системой авторизации и биллинга. На этом же ресурсе размещается секретный ключ доступа на вычислительный ресурс по ssh.

Разработчик заключает договор с конечным пользователем (Пользователь) на использование своей программы как сервиса и регистрирует Пользователя в своей биллинговой системе.

Пользователь вводит входные данные через веб-интерфейс, предварительно пройдя этап аутентификации. Введённые данные проверяются на корректность и передаются от имени Разработчика на вычислительный сервер, где стандартными средствами формируется задача и ставится в очередь на выполнение.

После завершения счёта Пользователь оповещается об этом тем или иным путём и получает возможность получить результат через веб-интерфейс.

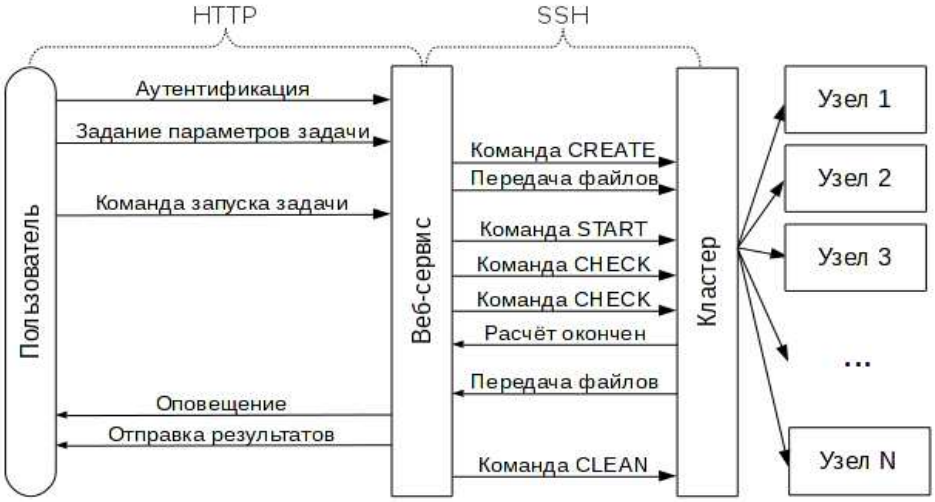

Рис. 1. Взаимодействие компонентов системы

Данная организация запуска задач без внесения существенных изменений может быть модифицирована в двух направлениях:

Один веб-интерфейс — много кластеров

При увеличении количества Пользователей Разработчик может получить несколько учетных записей на различных кластерах (или в облачных системах). В этом случае веб-интерфейс кроме систем авторизации и биллинга должен быть снабжён собственным планировщиком, способным распределить задачи Пользователей по нескольким вычислительным ресурсам.

Много веб-интерфейсов один кластер

Возможна ситуация, когда веб-интерфейс не является общедоступным, а, например, расположен на персональном компьютере Пользователя. В этом случае его можно рассматривать как удобную замену интерфейсу командной строки для запуска типовой задачи. В такой ситуации веб-интерфейс может не иметь подсистем биллинга и авторизации и копироваться на персональные компьютеры участников некоего проекта, совместно использующих одну учетную запись.

Функциональность системы

Веб-интерфейс с точки зрения Пользователя обеспечивает следующий функционал: авторизация Пользователя, ввод исходных данных и формирование задачи, просмотр состояния сформированных задач, получение результатов счета, просмотр истории ранее запущенных задач, просмотр состояния счета.

Интерфейс администратора системы позволяет: менять информацию об адресе сервера вычислений и способе связи с ним, регистрировать Пользователей, менять им пароли, вводить информацию об оплате и цены на услугу для биллинговой системы, просматривать статистику запуска задач как в целом, так и по каждому из Пользователей.

В случае изменения формата входных данных счетной программы интерфейс администратора позволяет менять форму для вводимых данных.

Риски взлома системы

Система состоит из двух основных частей: веб-интерфейса и кластера. Поскольку они могут принадлежать различным организациям, то система проектировалась так, чтобы в случае взлома веб-интерфейса минимизировать риск взлома кластера.

В случае взлома хоста с веб-интерфейсом атакующий получает данные для авторизации Разработчика на кластере и пары логин-пароль всех Пользователей веб-части системы.

С точки зрения Пользователей это означает, что атакующий может получить доступ к их исходным данным и к результатам расчётов. Это обычный риск при хранении любых данных на любых сервисах с доступом по сети. В ситуации, когда это все-таки критично, можно использовать шифрование данных перед отправкой их через веб-интерфейс и дешифрование на кластере перед запуском программы.

С точки зрения Разработчика взлом веб-интерфейса угрожает запуском неопределённого количества вычислительных сеансов, что может нанести ему коммерческий ущерб. Возможный вариант противодействия — сквозная нумерация задания заданий, подпись входных данных Пользователем до отправки через веб-интерфейс и проверка подписи и номера задания на кластере.

С точки зрения Владельца атакующий ограничен списком действий, явно прописанных в конфигурации ограниченного шелла, используемого при доступе через ssh. Для запуска произвольных программ (например, по добыче биткойнов, рассылке спама или эксплуатации уязвимостей ОС) атакующему требуется найти уязвимость в кластерной части системы, объем кода которой сравнительно невелик, что позволяет проводить регулярный его аудит.

В случае взлома головного узла кластера атакующий может читать и подменять данные задач. Для минимизации последствий на головном узле кластера не должны храниться данные для аутентификации в веб-интерфейсе. Разработчик веб-интерфейса не должен полагаться на корректность данных, возвращаемых кластерной частью.

Протокол запуска задач на узлах

Для взаимодействия веб-интерфейса и кластера используется протокол ssh. Этот протокол обеспечивает защищенное от прослушивания соединение между узлами и взаимную аутентификацию серверной и клиентской сторон. Клиентом выступает хост, на котором запущен веб-интерфейс системы, а сервером — головной узел кластера. При необходимости, ssh может быть заменён на любой другой защищённый канал связи, обеспечивающий удалённый запуск программ.

Взаимодействие веб-интерфейса с кластером всегда инициируется веб-интерфейсом. Веб- интерфейс отправляет строку команды, кластер после каждой команды возвращает строку ответа. При успешном выполнении команды эта строка начинается с "OK" и содержит результат команды, в противном случае строка начинается с "ERR" и содержит описание ошибки. В дальнейшем описании OK и ERR в начале строки опускаются.

Опишем подробнее команды работы с задачами.

create <taskid> — создаёт задачу.

taskid — идентификатор задачи, сгенерированный веб-интерфейсом. Любая строка без символов "." и "/"

При успешном завершении возвращает путь к служебному каталогу созданной задачи. Псевдослучайное имя каталога генерируется кластером. Этот путь можно использовать для копирования входных и выходных файлов задачи с помощью протокола scp.

start <taskid> <progname> [args] … — запускает программу на кластере.

taskid — идентификатор задачи, предварительно созданный с помощью команды create

progname — имя программы. Возможные имена программ содержатся в файле конфигурации шелла programs.conf

args — один или несколько аргументов программы

При успешном завершении возвращает пустую строку.

check <taskid> — проверяет статус запущенной программы.

taskid — идентификатор задачи

При успешном завершении возвращает статус задачи:

- NOTSTARTED — команда start ещё не была выполнена
- PENDING — задача находится в очереди на выполнение
- RUNNING — задача выполняется
- COMPLETED — задача завершилась
- FAILED — задача завершилась с ненулевым кодом возврата
- NODE_FAIL — при выполнении задачи произошла аппаратная ошибка на одном из узлов
- TIMEOUT — задача была снята из-за превышения максимального времени счёта
- CANCELLED — задача была снята администратором кластера
- CONFIGURING — узлы кластера подготавливаются к запуску задачи
- - COMPLETING — узлы кластера подготавливаются к завершению задачи

clean <taskid> — удаляет задачу и все файлы, относящиеся к ней.

taskid — идентификатор задачи

При успешном завершении возвращает пустую строку.

Интерпретатор команд на кластере

Для минимизации поверхности атаки на кластер возможности веб-интерфейса после аутентификации ограничены. Вместо полноценного интерпретатора командной строки, ему предоставляется оболочка, в которой разрешены только четыре команды для работы с задачами: создание, запуск, проверка состояния и удаление задачи. Реальные действия, выполняемые на кластере в результате интерапретации этих команд, описываются в отдельном файле, недоступном для редактирования.

Кроме выполнения четырёх базовых команд оболочка предоставляет урезанные возможности по передаче файлов с помощью протокола scp. Возможности этого протокола также были ограничены — запрещены работа с файлами за пределами заданного каталога и просмотр списка файлов в каталоге.

Код оболочки состоит из двух функциональных составляющих: поддержка протокола взаимодействия с веб-интерфейсом и поддержка взаимодействия с системой запуска задач. Первая компонента универсальна, вторая должна быть модифицирована при переносе на конкретный кластер.

Конфигурационный файл интерпретатора команд

Имя программы, передающееся при помощи команды start, должно быть описано в файле programs.conf.

Каждая строка этого файла имеет вид:

<progname> <launchargs> <proglocation> [progargs]...

progname — имя программы;

launchargs — параметры запуска программы, которые будут переданы системе запуска (например, здесь можно задать максимальное время счёта, число узлов, выходной файл);

proglocation — полный путь до исполняемого файла программы;

progargs — аргументы программы. К этим аргументам при запуске будут добавлены аргументы из команды start.

Детали реализации сервисов

При написании веб-интерфейса и интерпретатора команд была поставлена задача минимизировать зависимость от ОС и каких-либо экзотических средств разработки. В результате, для реализации веб-интерфейса был использован Ruby On Rails, для написания интерпретатора команд — язык Python, для организации удалённого взаимодействия — протокол ssh и соответствующие утилиты командной строки.

Вычислительное ядро

В пакете Wellfrost (свидетельство о государственной регистрации программы для ЭВМ № 2012660988 от 4 декабря 2012 г.) в качестве основной математической модели (рис. 2), в соответствии с работами [4-6], для расчета распространения тепла от скважины в условиях вечной мерзлоты используется уравнение контактной (диффузионной) теплопроводности в грунте с неоднородными коэффициентами, включающее локализованную теплоемкость фазового перехода – подход, позволяющий решать задачу типа Стефана, без явного выделения границы таяния вечномерзлых пород [7]. Эта методика апробирована на 8-ми северных нефтегазовых месторождениях и показала расчетную точность 5% при сравнении с экспериментальными данными. Расчет задачи в полной трехмерной постановке для длительных (несколько десятков лет) промежутков времени требует значительных вычислительных ресурсов. Основная цель моделирования – это определение положения границы таяния (радиуса растепления) многолетнемерзлых пород (ММП) вблизи скважины, нагреваемой добываемой нефтью, или газом. Эта информация используется для обустройства и проектирования кустовых площадок. В соответствии с российскими строительными стандартами считается, что две скважины не могут быть пробурены ближе, чем два радиуса растепления от функционирующей скважины на всем промежутке эксплуатации месторождения в течении 30 лет.

В ходе серийных отладочных расчетов был определен основной набор исходных параметров, входящих

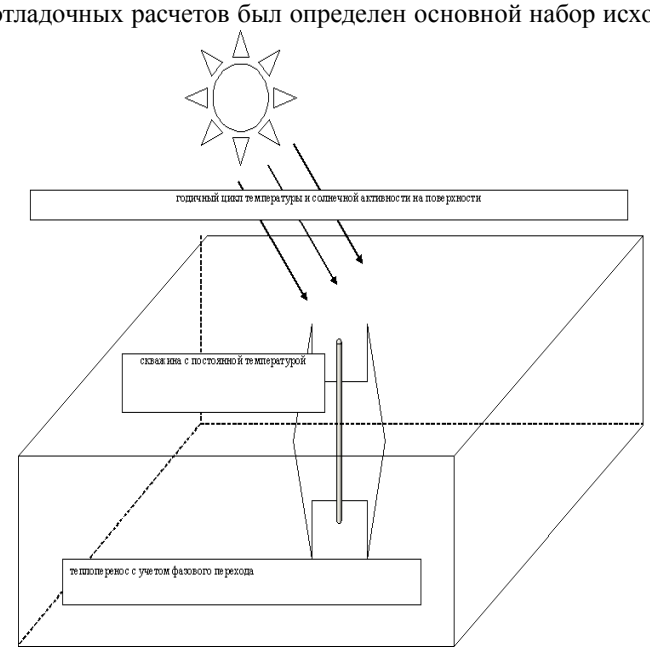

Рис. 2. Схема модельной расчетной области

в модель, и учитывающих географические особенности конкретного месторождения. Минимальный набор параметров – это температура грунта и флюида в скважине, вид грунта и географические координаты месторождения. Полный алгоритм расчета, включающий построение сетки, задание параметров расчета и управление итерационным процессом, остается для пользователя «за кадром». В качестве основного результата моделирования является нахождение значения радиуса растепления от скважины на заданной глубине. После завершения расчета пользователю также может быть представлена и графическая информация. Типичная картина тепловых полей представлена на рис. 3.

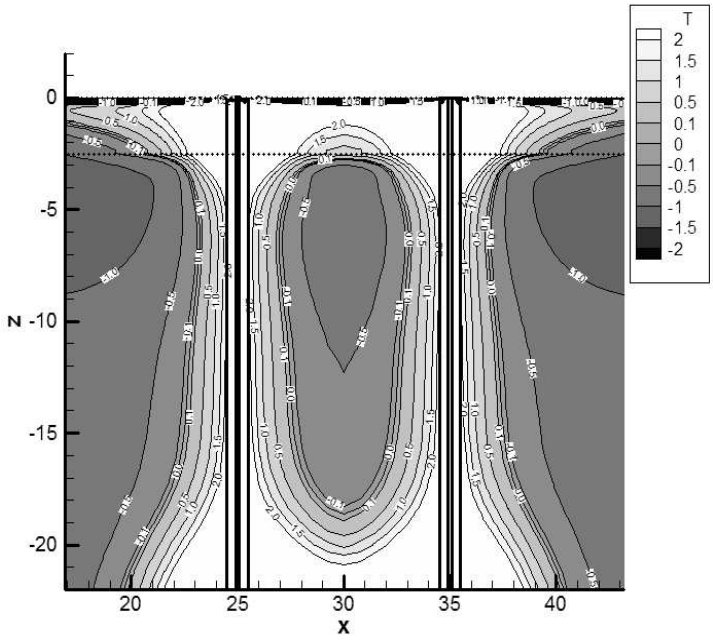

Рис. 3. Две изолированные скважины через 5 лет эксплуатации

Заключение

Описанная система была реализована в ИММ УрО РАН для поддержки удалённого доступа к программе расчёта растепления скважин в условиях вечной мерзлоты Wellfrost. Созданная система Wellfrost (вычислительное ядро и облачная среда) позволит удаленно получать рассчитанные на суперЭВМ долгосрочные прогнозы по динамике растепления ММП от различных инженерных объектов на кустовых площадках в зависимости от географического положения, технических параметров скважин и литологии грунта.

На настоящий момент реализованы все ключевые компоненты системы за исключением подсистемы биллинга. Программа Wellfrost адаптирована для запуска в многопользовательской среде.

Система запущена в опытную эксплуатацию для демонстрации потенциальным заказчикам.

ЛИТЕРАТУРА:

1. Hawick, K.A., James, H.A. Web services for remote management of scientific simulations //Proceedings Paper International Conference on Web Technologies, Applications, and Services JUL 04-06, 2005 Calgary, CANADA PP. 100-105
2. Kim, Huioon; Chun, Kyungwon; Jung, Kil Su; et al. 7th IEEE International conference on computer and information technology, proceedings 2007 PP. 218-222
3. В.С. Заборовский, А.А. Лукашин, А.С. Ильяшенко Защищенная платформа облачных вычислений для задач компьютерного инжиниринга//Научный сервис в сети Интернет: все грани параллелизма: Труды Международной суперкомпьютерной конференции (23-28 сентября 2013 г., г. Новороссийск).- М.: Изд-во МГУ, 2013. с. 111-116
4. M.Y. Filimonov, Vaganova, N.A. Simulation of thermal stabilization of soil around various technical systems operating in permafrost // Applied Mathematical Sciences. 2013. 7 (141-144). PP. 7151-7160. DOI: 10.12988/ams.2013.311669.
5. Н.А. Ваганова, Филимонов М.Ю. Прогнозирование изменений в вечной мерзлоте и оптимизация эксплуатации инженерных систем // Вестник НГУ. Сер. Математика, механика, информатика. 2013. Т. 13. № 4. С. 37-42.
6. M.Yu. Filimonov, Vaganova N.A. Simulation of thermal fields in the permafrost with seasonal cooling devices // Proceedings of the Biennial International Pipeline Conference IPC. 2012. 4. PP. 133 — 141. DOI: 10.1115/IPC2012-90287.
7. Самарский А.А., Вабищевич П.Н. Вычислительная теплопередача. М.: Едиториал УРСС, 2003. 784 с.