НЕПРЕРЫВНАЯ ИНТЕГРАЦИЯ И ДОСТАВКА С ИСПОЛЬЗОВАНИЕМ ANSIBLE

ВВЕДЕНИЕ

Ansible – это мощный язык автоматизации с открытым исходным кодом, уникальный тем, что отлично подходит не только для управления, но и для развертывания и оркестрации ИТ-систем. Ansible изначально создавался и для эффективного решения широкого круга задач автоматизации, и как простой универсальный базис для замены традиционных средств управления, а в итоге оказался весьма полезен во многих областях.

Например, при обеспечении нулевых простоев в ходе непрерывной интеграции и доставки приложений (CI/CD). Обычно эта задача решается за счет обширной доработки софта, применения различных программных пакетов и массы ухищрений, уникальных для каждой конкретной конфигурации. Ansible же изначально спроектирован в расчете именно на такие сценарии оркестрации и предлагает готовое решение «все в одном».

НЕПРЕРЫВНАЯ ИНТЕГРАЦИЯ И ДОСТАВКА ПРИЛОЖЕНИЙ (CI/CD)

Практика разработки программных систем за последние 10 лет показывает, что длинный жизненный цикл версий ПО (каскадная модель разработки) имеет гораздо более высокие накладные расходы по сравнению с коротким циклом (так называемая «итеративная» или agile-разработка). Все дело в аритмичности: когда программисты только начинают работать над новой версией, ИТ-специалистам, отвечающим за тестирование и развертывание, попросту нечего делать. Но чем ближе версия к выпуску, тем сильнее заняты ИТ-специалисты, и тем чаще программистам приходится переключать контекст, чередуя работу над ошибками и планирование следующей версии.

Кроме того, длинный цикл увеличивает интервал между выявлением и устранением программных ошибок и недочетов, что особенно критично для больших веб-систем с многомиллионной пользовательской аудиторией. Поэтому индустрия производства ПО стремительно осваивает методологии agile под лозунгом «выпускать быстрее и чаще», чтобы участники процесса разработки могли реже переключать контекст работы и гораздо быстрее создавать, отлаживать и внедрять доработки и новшества.

Автоматизация контроля качества, TDD-разработка через тестирование (Test Driven Development) и другие сопутствующие техники еще больше повышают эффективность новых методов работы. И если лозунг «выпускать быстрее и чаще» стал синонимом прогресса, то CI/CD может стать вашим путем к истинному блаженству. Agile символизирует движение вперед, и следование по этому пути сулит организации весомые преимущества, ключом к которым является автоматизация. Отсюда и повышенный интерес к технологиям, заставляющим шестеренки крутиться быстрее и сводящим участие человека к строго необходимому минимуму.
Ниже мы покажем, почему решения Ansible и Ansible Tower от Red Hat идеально подходят для оркестрации ИТ-систем в рамках современных процессов разработки ПО.

НУЛЕВЫЕ ПРОСТОИ

Любая автоматизация процессов должна обеспечивать обновление без приостановки операций, иначе говоря, с нулевыми простоями.

Простои – это упущенная выгода и недовольные заказчики. Поэтому в веб-системах массового обслуживания, пользователи которых распределены по всем часовым поясам, плановое отключение допускается только в действительно серьезных случаях, перечень которых явно не включает в себя обновление версий приложения. Схожим образом дела обстоят и в корпоративных средах, где недоступность интранета или учетной системы резко снижает продуктивность сотрудников. Таким образом, любая автоматизация процессов должна обеспечивать обновление без приостановки операционной деятельности - иначе говоря, с нулевыми простоями.
Добиться нулевых простоев вполне реально, но для этого нужны соответствующие инструменты - такие, что обеспечивают расширенную, многоуровневую и многоступенчатую оркестрацию, как, например, система Ansible.

СИСТЕМЫ СБОРКИ ПРИЛОЖЕНИЙ

Непрерывная доставка (CD) начинается с непрерывной интеграции (CI). Вкратце, CI – это система, которая мониторит репозитории исходных кодов на предмет изменений, самостоятельно прогоняет соответствующие тесты и автоматически выполняет сборку (а в идеале и тестирование) новой версии приложения при каждом обновлении кода. В качества примера такой системы можно привести проект Jenkins (jenkins.io).
Для передачи эстафеты CD-системе после успешной сборки новой версии приложения подсистема сборки CI-системы может вызывать Ansible, чтобы сразу же предоставить эту новую версию тем, кто выполняет модульное или интеграционное тестирование. В частности, Jenkins может задействовать Tower для развертывания сборок в различных средах, причем тестовая или промежуточная среда могут моделироваться на основе производственной среды, что сильно повышает предсказуемость на всем жизненном цикле ПО. Данные, возвращаемые Ansible по результатам выполнения сценариев автоматизации, можно напрямую задействовать в заданиях Build Systems системы Tower. На самом деле, как будет показано ниже, Tower даже позволяет тестировать сценарии развертывания в промежуточной среде, прежде чем запускать их на «боевых» серверах.

ПООЧЕРЕДНОЕ ОБНОВЛЕНИЕ МНОГОУРОВНЕВЫХ ПРИЛОЖЕНИЙ

CD-система в обязательном порядке должна уметь оркестрировать процессы поочередного обновления (rolling update) многоуровневых приложений. Благодаря push-архитектуре и мощным возможностям многоуровневой многоступенчатой оркестрации, Ansible отлично справляется с этой задачей, позволяя легко обновить любое приложение уровень за уровнем, обмениваясь при этом данными между ними.

Для реализации поочередного обновления в Ansible используются сценарии Play, которые позволяют точно задать группу целевых хостов и назначить задачи (Role), которые должны быть на них выполнены. Задачи – это обычно объявления о том, что конкретный ИТ-ресурс должен находиться в заданном состоянии, например, для одной версии ПО должен быть установлен определенный пакет, а для другой требуется свериться с репозиторием кода. Топологии веб-приложений, как правило, требуют обновлять их в строгой последовательности. Например, перед тем, как обновлять сервер приложений необходимо модифицировать схему БД и сбросить содержимое кэш-серверов.

Кроме того, ни в коем случае нельзя обновлять приложения и системные конфигурации на всех машинах одновременно.

При перезапуске сервиса он какое-то время остается недоступен, замена версии приложения тоже не происходит мгновенно. Поэтому, прежде чем обновлять систему, ее нужно вывести из пула балансировки. Как следствие, нужна возможность автоматизировать операции подключения и отключения машин от пула.

«Последовательно» – вот ключевое слово, объясняющее, почему Ansible позволяет очень точно контролировать размер окна поочередного обновления. Кроме того, отработка таких обновлений ведется очень аккуратно, и если на каком-то этапе возникает сбой, то обновление приостанавливается, чтобы не вывести из строя оставшуюся часть ИТ-инфраструктуры.

Благодаря push-архитектуре и мощным возможностям многоуровневой многоступенчатой оркестрации, Ansible отлично справляется с этой задачей.

НЕПРЕРЫВНОЕ РАЗВЕРТЫВАНИЕ ДЛЯ СЦЕНАРИЕВ АВТОМАТИЗАЦИИ

Помимо CD-функционала для сервисов, действующих в режиме промышленной эксплуатации, можно также организовать непрерывное развертывание самих сценариев автоматизации (наборов инструкций Ansible Playbook). Это позволяет системным администраторам и разработчикам управлять сценариями с помощью репозитория исходных кодов, тестировать эти сценарии в промежуточной среде и автоматически переносить их в производственную среду в случае успешной обкатки. Иначе говоря, при работе со сценариями вы получаете все методологические и другие преимущества центрального репозитория кода, к которым привыкли при разработке софта.

Внесение изменений в ПО и системные конфигурации является одной из главных причин внеплановых отключений. Поэтому, помимо автоматизированного тестирования, не помешает и человеческий контроль. Его можно организовать путем интеграции с системой инспекции кода, наподобие Gerrit (gerritcodereview.com), и применять изменения только после того, как их утвердят ответственные сотрудники.

ПООЧЕРЕДНЫЕ ОБНОВЛЕНИЯ И СИСТЕМЫ БАЛАНСИРОВКИ НАГРУЗКИ

Ansible очень грамотно работает с системами балансировки нагрузки при выполнении поочередных обновлений. Поэтому вы можете просто написать в сценарии Playbook, в любом цикле для группы хостов, что-то вроде «выполнить это действие в системе X от имени хоста Y», и Ansible позаботится об остальном.
Ansible хорошо взаимодействует с балансировщиками нагрузки всех видов и умеет устанавливать флаг временного отключения хоста, чтобы деактивировать для него мониторинг доступности на период обновления. Простая схема «отключаем мониторинг – убираем из пула – обновляем нужный уровень ПО – возвращаем в пул – включаем мониторинг» легко реализует поочередное обновление с нулевыми простоями и без ложных срабатываний тревоги. И все это в полностью автоматизированном режиме, без участия оператора.

ИНТЕГРИРОВАННОЕ ПРОМЕЖУТОЧНОЕ ТЕСТИРОВАНИЕ

Tower может работать с различными файлами инвентаризации ресурсов (Inventory), что позволяет легко тестировать сценарии поочередного обновления в промежуточной среде, прежде чем запускать их на «боевых» серверах. Для этого достаточно смоделировать производственную среду в тестовой, запустить Ansible с параметром «-i» и указать, какой файл инвентаризации должен использоваться при выполнении сценария – для тестовой среды или для производственной. Сам сценарий при этом модифицировать не нужно.

РАЗВЕРТЫВАНИЕ НА ОСНОВЕ СИСТЕМЫ КОНТРОЛЯ ВЕРСИЙ

Некоторые любят выполнять упаковку приложений вместе с пакетами ОС (RPM, debs, и т. п.), однако зачастую, особенно для веб-приложений, такая упаковка не нужна. Поэтому в состав Ansible входят сразу несколько модулей для развертывания приложений непосредственно из систем управления версиями. В сценарии Playbook можно прописать сверку с репозиторием кода по заданному тегу или номеру версии, после чего Ansible проверит выполнение этого условия на всех целевых серверах и активирует последующие действия, только если версия нуждается в замене, устранив, таким образом, ненужные перезапуски служб.

ИНТЕГРАЦИЯ СО СРЕДСТВАМИ МОНИТОРИНГА

Как полноценная система оркестрации, Ansible поддерживает интеграцию с APM-системами управления производительностью приложений на уровне мониторинга. Допустим, на этапе развертывания или интеграционного тестирования вместе с приложением необходимо установить или обновить программный агент APM. В Ansible для этого есть специальная роль, и после установки и активирования агента, Ansible может настроить его в стеке APM-мониторинга (если он еще не настроен), чтобы специалисты по управлению приложениями могли сразу же проконтролировать, что новая версия установилась и работает без проблем.
Если после обновления приложения в производственной среде что-то пошло не так, средства мониторинга могут вызвать Ansible, чтобы выполнить откат к предыдущей версии. Разумеется, только если такой откат разрешен.

УВЕДОМЛЕНИЕ О СОБЫТИЯХ

В парадигме CI/CD зачастую важно получать уведомления о событиях как можно быстрее. Ansible предлагает для решения этой задачи как встроенные функции, включая модуль электронной почты, так и интеграцию с внешними инструментами оповещения, наподобие мессенджеров, социальных сетей или систем регистрации событий.

РАЗВЕРТЫВАНИЕ С ИСПОЛЬЗОВАНИЕМ МОДЕЛИ СОСТОЯНИЯ РЕСУРСОВ

Объединение процессов конфигурации систем и развертывания приложений в рамках одной инструментальной цепочки гораздо эффективнее схемы с несколькими специализированными инструментами, и повышает согласованность политик ОС и приложений.

Одна из ключевых особенностей Ansible, делающая его очень полезным инструментом для развертывания приложений – это штатное использование в процессах обновления ПО модели состояния ресурсов, завоевавшей популярность при управлении системными конфигурациями. В отличие от традиционных средств управления с открытым исходным кодом, Ansible не нужно дооснащать каким-либо дополнительным софтом или особыми сценариями, чтобы организовать доставку приложений.

В Ansible можно очень точно прописать и проконтролировать порядок событий на разных уровнях архитектуры, что позволяет делегировать действия другим системам, а также комбинировать директивы ресурсной модели (по типу «пакет X должен быть в состоянии Y») и традиционные сценарные команды (наподобие «run script.sh») в рамках одного процесса.

Ansible также позволяет легко запускать команды проверки различных условий и принимать решения по результатам их выполнения. Объединение процессов конфигурации систем и развертывания приложений в рамках одной инструментальной цепочки гораздо эффективнее схемы с несколькими специализированными инструментами, и, кроме того, повышает согласованность политик ОС и приложений.

ТЕСТИРОВАНИЕ В ХОДЕ РАЗВЕРТЫВАНИЯ

Чем больше возможностей, тем выше ответственность. Автоматизация процессов непрерывной доставки резко повышает опасность развертывания сбойной конфигурации на всех узлах системы. Чтобы снизить риски, Ansible предлагает вставлять в сценарии контрольные тесты, которые прервут поочередное обновление, если что-то пойдет не так. Для проверки различных условий, включая статус функционирования служб, можно развертывать произвольные тесты с использованием модулей Command или Script, и даже создавать такие тесты в виде отдельных модулей Ansible.

Модуль Fail может прервать выполнение сценария на хосте в любой момент, что позволяет отловить сбои на ранней стадии поочередного обновления. Например, из-за отличия промежуточной среды от производственной в последней возникает конфигурационная ошибка, которая выводит из строя «боевые» сервера. На этот случай в сценарии Playbooks можно прописать аварийный выход на первом же этапе поочередного обновления. И если у вас 100 серверов, а размер окна поочередного обновления равен 10, то такая аварийная остановка даст время спокойно во всем разобраться, исправить сценарий и продолжить обновление.

В случае сбоя Ansible не продолжает работу, оставляя системы в полунастроенном состоянии, а генерирует ошибку, чтобы привлечь внимание оператора и сообщить ему, на каких хостах цикл обновления прошел с ошибками, и сколько изменений было выполнено на каждой платформе.

В Ansible есть режим имитационного прогона, когда система формирует отчет о том, какие изменения были бы проведены при выполнении сценария без его реального выполнения.

В Ansible есть режим имитационного прогона, когда система формирует отчет о том, какие изменения были бы проведены при выполнении сценария без его реального выполнения.

ПРОВЕРКА СООТВЕТСТВИЯ

Есть среды, где конфигурации меняют только тогда, когда без этого уже никак. Любые изменения в таких средах предварительно анализируются и утверждаются соответствующими специалистами. Однако место для систем непрерывной доставки найдется и здесь, хотя и с оговорками.

В Ansible есть режим имитационного прогона (активируется флагом «--check»), когда система формирует отчет о том, какие изменения были бы проведены при выполнении сценария. Поскольку реального выполнения сценария при этом не происходит, имитационный прогон хоть и не позволяет отловить ошибки, но помогает лучше понять и проанализировать детали и результаты предлагаемых изменений.

С другой стороны, даже при непрерывном развертывании новых сборок, Ansible позволяет гораздо чаще запускать проверки соответствия, чтобы поймать момент, когда какие-то вещи в производственной среде меняются в результате человеческой вмешательства и должны быть исправлены путем запуска соответствующего сценария Ansible, например, чтобы сменить версию ПО, подкорректировать разрешения и т. п.

РАЗВЕРТЫВАНИЕ НА АВТОПИЛОТЕ

Благодаря обширному набору функций и инструментов, Ansible идеально подходит для решения задач CI/CD, в том числе для многоуровневой многоступенчатой оркестрации процессов поочередного обновления ПО с нулевыми простоями. Эта особенность Ansible плюс его уникальная архитектура и отсутствие программных агентов на целевых хостах (повышающее безопасность и избавляющее от необходимости управлять самой системой управления) позволяют доступно описать и легко автоматизировать сложные процессы развертывания. Раньше такие процессы выполнялись исключительно операторами (как вручную, так и с частичной автоматизацией) и требовали согласованных действий всех участников процесса. Ansible же реализует здесь режим полного автопилота.

Развертывание на автопилоте открывает дорогу к agile-разработке, позволяя вносить изменения в ПО быстрее, с меньшим числом ошибок и значительно реже переключая рабочий контекст. Никаких плановых простоев и, что важнее, человеческих ошибок – это именно то, что так нужно критически важным корпоративным приложениям и веб-системам массового обслуживания.

Архитектурные особенности Ansible и возможность интеграции с CI-системами, наподобие Jenkins, обеспечивают автоматизацию не только процессов управления конфигурациями, но и гораздо более широкого круга ИТ-задач. Именно поэтому мы называем Ansible комплексной системой оркестрации, а не просто инструментом развертывания ПО и управления конфигурациями.

ДОПОЛНИТЕЛЬНЫЕ СВЕДЕНИЯ

Примеры сценариев автоматизации Ansible можно найти на сайте github.com/ansible/ansible-examples

Узнайте больше о решении Ansible Tower
ansible.com/tower

redhat.

ЧТО ТАКОЕ ANSIBLE

Ansible представляет собой систему ИТ-автоматизации с открытым исходным кодом, которая развивается силами сообщества разработчиков при спонсорской поддержке Red Hat. Ansible – это единственный язык автоматизации, которым могут пользоваться все участники ИТ-процесса: сетевые и системные администраторы, разработчики, управленцы. Решение Ansible от Red Hat – это система автоматизации корпоративного класса, охватывающая все этапы жизненного цикла приложений, включая серверы, облачные инфраструктуры, контейнеры и все, что связывает их между собой. Ansible Tower от Red Hat – это коммерческое предложение, которое расширяет возможности открытой технологии Ansible для эффективного развертывания многоуровневых систем в больших и сложных средах.

О КОМПАНИИ RED HAT

Компания Red Hat – ведущий поставщик надежных и высокопроизводительных технологий облачных вычислений, виртуализации, хранения данных, связующего и операционных систем Linux , в основе которых лежат решения с открытым кодом, развиваемые силами сообщества разработчиков. Компания также предлагает неоднократно отмеченные наградами услуги технической поддержки, обучения и консалтинга. Выступая в качестве центрального узла всемирной сети корпоративных заказчиков, партнеров и сообщества разработчиков открытого ПО, Red Hat способствует созданию инновационных технологий, раскрепощающих ресурсы для роста и помогающих заказчику во всеоружии встретить будущее.

ЕВРОПА, БЛИЖНИЙ ВОСТОК И АФРИКА
00800 7334 2835
europe@redhat.com

РОССИЯ И СНГ
+7 495 662 88 37
russia@redhat.com

ТЕХНИЧЕСКАЯ ПОДДЕРЖКА НА РУССКОМ ЯЗЫКЕ
+7 499 951 13 24
8 800 555 27 88

redhat.