ли им этот контракт право на запуск программы на нескольких виртуальных машинах, запущенных на одной и той же физической машине? Многие поставщики программного обеспечения не знают, что делать в подобных случаях.

Проблема усугубляется в тех компаниях, которые имеют лицензию, разрешающую одновременно запускать программу на нескольких машинах, особенно в тех случаях, когда виртуальные машины создаются и удаляются по мере необходимости.

В некоторых случаях поставщики программного обеспечения включают в лицензии особый пункт, запрещающий лицензиату запуск программы на виртуальной машине или на неавторизированной виртуальной машине. Для компаний, запускающих все свое программное обеспечение исключительно на виртуальных машинах, такое положение дел может вылиться в серьезную проблему. Будут ли подобные ограничения оспариваться в суде и как на них отреагируют пользователи, нам еще предстоит увидеть.

7.11. Облака

В резком взлете облачных вычислений технология виртуализации играет решающую роль. Существует множество облаков. Некоторые из них относятся к публичным и доступны любому, кто согласен платить за использование ресурсов, другие же являются закрытыми облаками организаций. Также разные облака предлагают разные услуги. Некоторые из них дают своим пользователям доступ к физическому оборудованию, но большинство виртуализируют свою среду. Некоторые предлагают просто машины, как виртуальные, так и физические, и больше ничего, а некоторые — готовое к использованию и способное к объединению весьма интересными способами программное обеспечение или платформы, облегчающие своим пользователям разработку новых служб. Поставщики облачных услуг обычно предлагают различные категории ресурсов, таких как «большие машины», «малые машины» и т. д.

При всех разговорах об облаках, похоже, мало кто с уверенностью может сказать, что они на самом деле собой представляют. Национальный институт стандартов и технологий (США), являясь источником, к которому всегда можно прибегнуть, перечислил пять основных характеристик:

1. **Самообслуживание по требованию** (On-demand self-service). Пользователи должны иметь возможность получать ресурсы автоматически, без человеческого участия.
2. **Широкий доступ по сети** (Broad network access). Все эти ресурсы должны быть доступны по сети посредством стандартных механизмов, чтобы ими могли воспользоваться гетерогенные устройства.
3. **Объединение ресурсов в пул** (Resource pooling). Компьютерные ресурсы, принадлежащие поставщику, должны быть объединены в пул для обслуживания нескольких пользователей с возможностью динамического назначения и освобождения ресурсов. Пользователи обычно не знают точного местонахождения «своих» ресурсов и даже того, в какой стране они расположены.
4. **Быстродействующая эластичность** (Rapid elasticity). Должна быть предоставлена возможность эластичного получения и освобождения ресурсов, может быть, даже в автоматическом режиме, чтобы происходило незамедлительное масштабирование в соответствии с потребностями пользователей.

5. **Учтенные услуги** (Measured service). Поставщик должен вести учет потребленных ресурсов тем способом, который соответствует типу заранее оговоренных облачных услуг.

7.11.1. Облака в качестве услуги

В данном разделе мы рассмотрим облака, сконцентрировавшись на виртуализации и операционных системах. Особенно подробно будут рассмотрены те облака, которые предлагают непосредственный доступ к виртуальной машине, которой пользователь может воспользоваться любым подходящим для него способом. Следовательно, одно и то же облако может запускать разные операционные системы, возможно, на одном и том же оборудовании. В понятиях облака это называется инфраструктурой в качестве услуги (Infrastructure As A Service (**IAAS**)) в противоположность платформе в качестве услуги (Platform As A Service (**PAAS**)), когда предоставляется среда, включающая такие компоненты, как конкретная операционная система, база данных, веб-сервер и т. д., программного обеспечения в качестве услуги (Software As A Service (**SAAS**)), когда предоставляется доступ к конкретному программному продукту, например Microsoft Office 365 или Google Apps, и многим другим типам облаков в качестве услуг. Одним из примеров IAAS-облака является Amazon EC2, основанное на гипервизоре Xen и насчитывающее сотни тысяч физических машин. При условии наличия средств можно получить сколько угодно вычислительных мощностей.

Облака могут изменять способ производимых компаниями вычислений. В целом, объединение компьютерных ресурсов в небольшом количестве мест (с удобным расположением возле электростанций и там, где дешевле можно будет охлаждать оборудование) позволяет сэкономить средства при масштабировании. Привлечение внешних ресурсов к обработке информации означает, что вам не нужно особо волноваться об управлении вашей IT-инфраструктурой, резервном копировании, обслуживании, амортизации, масштабировании, надежности, производительности и, возможно, безопасности. Все это делается в одном месте, и, если предположить высокую компетентность поставщика облачных услуг, делается качественно. В связи с этим можно подумать, что теперь IT-менеджеры стали намного счастливее, чем 10 лет назад. Но наряду с исчезновением этих тревог появились новые. Можно ли действительно доверять своему поставщику облачных услуг безопасное хранение ваших конфиденциальных данных? Будет ли конкурент при работе на той же инфраструктуре иметь возможность выводить информацию, которую желательно сохранять в закрытом виде? Какой закон (или законы) применим к вашим данным (например, если поставщик облачных услуг находится в США, то распространяется ли на ваши данные «Патриотический акт», даже если ваша компания находится в Европе)? Если все ваши данные хранятся в облаке X, сможете ли вы извлечь их оттуда или же вы будете привязаны к этому облаку и его поставщику навсегда (это называется **привязкой к поставщику** (vendor lock-in)?

7.11.2. Миграция виртуальных машин

Технология виртуализации позволяет не только создавать IAAS-облака для одновременного запуска на одном и том же оборудовании нескольких разных операционных систем, но и искусно управлять ими. О возможности выделения большего количества ресурсов, чем их фактически имеется, особенно в сочетании с дедупликацией, мы уже говорили. Теперь давайте рассмотрим другие проблемы управления: что, если

машине понадобится обслуживание (или даже замена), а на ней запущено множество важных машин? Наверное, клиенты не обрадуются, если их системы утратят работоспособность по причине того, что поставщик облачных услуг хочет заменить дисковый привод.

Гипервизоры разобщают виртуальную машину и физическое оборудование. Иными словами, для виртуальной машины на самом деле неважно, на какой машине она работает, той или этой. Следовательно, можно просто остановить все виртуальные машины и снова запустить их на совершенно новой машине. Но в результате этого получится весьма существенный простой. Задача состоит в том, чтобы переместить виртуальную машину с оборудования, нуждающегося в обслуживании, на новую машину вообще без остановки.

Немного более приемлемым подходом может быть не остановка виртуальной машины, а ее приостановка. Во время паузы мы как можно быстрее копируем страницы памяти, используемые виртуальной машиной, на новое оборудование, проводим корректное конфигурирование в новом гипервизоре, а затем возобновляем работу виртуальной машины. Кроме памяти требуется перенести подключения к устройству хранения данных и к сети, но если машины рядом, это можно сделать относительно быстро. Для начала можно сделать файловую систему на основе сети (подобно сетевой файловой системе — NFS), чтобы было все равно, на каком оборудовании работает ваша виртуальная машина, на серверной стойке 1 или 3. Также на новое место может быть просто переключен IP-адрес. И все-таки придется приостановить работу машины на вполне заметный период времени. Возможно, на это уйдет меньше времени, чем вы ожидали, но все же его невозможно будет не заметить.

Вместо этого современные решения виртуализации предлагают так называемую **живую миграцию** (live migration). Иными словами, перемещение виртуальной машины происходит без прекращения ее работы. Например, в этих решениях используется технология, подобная **миграции памяти с предварительным копированием** (pre-copy memory migration). Это означает, что страницы памяти копируются в то время, когда машина все еще обрабатывает запросы. Большинство страниц памяти не подвергается интенсивной записи, следовательно, их копирование безопасно. Следует помнить, что виртуальная машина все еще работает, поэтому страница может быть изменена после того, как уже скопирована. При изменении страниц памяти мы должны гарантировать копирование в место назначения самой последней их версии, поэтому помечаем такие страницы как измененные. Позже они будут скопированы заново. Когда скопировано большинство страниц памяти, мы остаемся с небольшим количеством измененных страниц. Теперь делается очень короткая пауза на копирование оставшихся страниц, и работа виртуальной машины возобновляется на новом месте. Хотя без паузы здесь все же не обходится, она настолько коротка, что это обычно не оказывает никакого влияния на приложения. Ситуация, когда простой не заметен, называется **незаметной живой миграцией** (seamless live migration).

7.11.3. Установка контрольных точек

Разобщение виртуальной машины и физического оборудования имеет дополнительные преимущества. В частности, уже говорилось, что машину можно приостановить. Это полезно само по себе. Если состояние приостановленной машины (например, состояние центрального процессора, страниц памяти и хранилища данных) сохраняется на

диске, у нас получается стоп-кадр работающей машины. Если программа устраивает полный кавардак на все еще работающей виртуальной машине, можно просто сделать откат к стоп-кадру и продолжить работу как ни в чем не бывало.

Наиболее простой способ создать стоп-кадр — это скопировать все, включая всю файловую систему. Но копирование диска в несколько терабайт может занять уйму времени, даже если это быстрый диск. К тому же приостанавливать работу машины на длительное время, пока проделывается все необходимое, нежелательно. Решение заключается в использовании технологий **копирования при записи** (copy on write), чтобы данные копировались только в случае крайней необходимости.

Создание стоп-кадров работает неплохо, но все же вызывает ряд вопросов. Что делать, если машина взаимодействует с удаленным компьютером? Мы можем сделать стоп-кадр системы и вернуть ее в прежнее состояние на более поздней стадии, но та часть, которая относится к обмену данными, уйдет в прошлое. Понятно, что эту проблему решить невозможно.

7.12. Изучение конкретных примеров: VMWARE

С 1999 года VMware, Inc. стала ведущим коммерческим поставщиком решений по виртуализации, предлагая продукты для настольных компьютеров, серверов, облаков, а теперь даже и сотовых телефонов. Компания поставляет не только гипервизоры, но и программы, управляющие виртуальными машинами в больших масштабах.

Изучение этого конкретного примера мы начнем с краткой истории становления компании. Затем будет дано описание VMware Workstation, гипервизора второго типа и первого продукта компании, всех сложностей в его конструкции и ключевых элементов этого решения. Затем будет дано описание происходившего в течение нескольких лет развития VMware Workstation. А в завершение будет дано описание ESX Server, гипервизора первого типа компании VMware.

7.12.1. Ранняя история VMware

Хотя идея использования виртуальных машин в 1960–1970-х годах была популярна как в компьютерной промышленности, так и в академических исследованиях, после 1980-х годов интерес к виртуализации был полностью утрачен, и на подъеме было производство персональных компьютеров. Только подразделение универсальных машин компании IBM все еще занималось виртуализацией. Действительно, компьютерные архитектуры, разработанные в то время, в частности архитектура x86 компании Intel, не предоставляла архитектурную поддержку виртуализации (например, они не отвечали критериям, выработанным Попеком и Голдбергом). Это весьма печальный факт, ведь центральный процессор 386, являвшийся полной переработкой процессора 286, был изготовлен спустя 10 лет после выхода статьи Попека и Голдберга и разработчики должны были лучше разбираться в поднятых в ней вопросах.

В 1997 году в Стэнфорде три будущих основателя компании VMware создали прототип гипервизора под названием Disco (Bugnion et al., 1997) с целью запуска товарных операционных систем (в частности, UNIX) на сверхбольшом микропроцессоре, разработанном в Стэнфорде, — на FLASH-машине. В ходе разработки этого проекта авторы поняли, что использование виртуальных машин может простым и элегантным