РАЗРАБОТКА ОБРАЗА ОПЕРАЦИОННОЙ СИСТЕМЫ FREEBSD для установки виртуального маршрутизатора и программного средства переключения путей маршрутизации

Е.Ряполова, к.п.н., доцент ОФ ПГУТИ,
М.Студянникова, к.п.н., доцент ОФ ПГУТИ / studyannikovam@mail.ru,
А.Преснов, к.п.н., доцент ОФ ПГУТИ,
К.Цветкова, к.п.н., доцент, ОФ ПГУТИ

УДК 004.4; DOI: 10.22184/2070-8963.2019.81.4.60.63

Насущная проблема отделов ИТ многих современных предприятий – нехватка ресурсов сервера для обработки необходимой информации. В статье предлагается виртуализировать сервер с целью увеличения его функциональных возможностей. Научная новизна состоит в анализе существующих решений виртуализации, выбора виртуальной платформы и разработки программного обеспечения для эффективного перехода на нее всей системы предприятия без потерь времени и данных. Практическая новизна состоит в разработке программного средства автоматического переключения путей маршрутизации, написанной на языке Perl.

Введение

Эксперты в области ИТ едины во мнении, что технологии виртуализации стали неотъемлемой частью современной ИТ-инфраструктуры, позволив существенно ускорить внедрение новых систем, а также оптимизировать затраты на поддержку существующих. Так, 37% организаций, принявших участие в недавнем опросе компании VMware, достигли серьезного сокращения затрат на оборудование и техническое обслуживание в результате внедрения виртуализации, 34% планируют сократить затраты на оборудование, а 35% – издержки на обслуживание. Также при опросе 25% респондентов заявили, что виртуализация облегчила развертывание новых приложений и услуг [1]. Таким образом, сформировалась цель разработки: повышение эффективности работы сервера сектора информационных технологий за счет увеличения количества решаемых задач на основе виртуализации. В статье рассмотрена одна из поставленных для достижения этой цели задач – разработать образ ОС FreeBSD для установки виртуального маршрутизатора

и программного средства переключения путей маршрутизации.

К истории вопроса: с начала 90-х годов наблюдалось лавинообразное увеличение парка серверов, что привело к возникновению множества проблем. Во-первых, это трудоемкость поддержки множества серверов и необходимость контроля работоспособности каждого из них. При отказе физического сервера необходимо переносить приложение на другой сервер, что может занять время, в течение которого соответствующий сервис будет недоступен. Во-вторых – нехватка физических серверов: при развертывании новой задачи необходимо закупать новые серверы для нее, что занимает время на поставку оборудования, а также на установку программного обеспечения. В-третьих, это ограничения в масштабируемости существующих физических серверов, особенно заметные при возрастании нагрузки приложений. При превышении потребностями задачи возможностей сервера, на котором она запущена, необходима модернизация сервера (однако наступает момент, когда дальнейшая модернизация невозможна – и тогда закупается новый, более мощный сервер, на который переносится приложение, что также требует затрат времени и ресурсов). Кроме того, существует проблема перераспределения нагрузки между всеми физическими серверами, существующими в организации. Как правило, осуществить такое перераспределение рациональным образом крайне трудно, и, как следствие, часть серверов оказывается перегруженной – в то время как существует большое количество малозагруженных серверов, мощности которых используются на 5–10%. Согласно статистике, средний уровень загрузки процессорных мощностей у серверов на Windows не превышает 10%, а у Unix-систем данный показатель составляет не более 30% [2].

Также существует множество серверных операционных систем, а установка только одной из них ограничивает возможности по установке различных доступных программ и приложений. Таким образом, увеличение возможностей сервера сводится к задаче установки виртуальной машины с несколькими серверными операционными системами, на каждой из которых устанавливается дополнительное программное обеспечение.

Разработка образа

Предположим, в рассматриваемой организации ядро информационной сети – базовая сеть передачи данных (СПД) – построена на основе оборудования фирмы Cisco и магистральных оптических кабелей общей протяженностью более 30 км. Протяженность кабельных трасс "витой пары" составляет более 90 км. Сеть охватывает все корпуса предприятия, а также удаленные филиалы, которые подключены по ВОЛС. Компьютерная сеть этого предприятия постоянно развивается, и задача повышения эффективности работы сервера становится все более актуальной. На сегодняшний день существует ряд проблем, которые можно решить с помощью технологии виртуализации. Это увеличение пропускной способности канала, распределение серверных ресурсов, установка на один физический сервер нескольких логических серверов при условии, что физический сервер не будет перегружен. Рассмотрим эти возможности.

При оценке эффективности использования разработанного программного обеспечения необходимо рассматривать обширный набор аспектов-критериев. Существуют различные подходы к оценке эффективности, основанные на методиках различных организаций (как коммерческих, так и независимых научных), оптимизированных для использования в разных областях хозяйственной деятельности. Оценка эффективности основывается на определении, выборе критериев для рассмотрения и оценки разработанного программного обеспечения по этим качествам.

Под эффективностью разработанного программного обеспечения понимается свойство ПО выполнять поставленную цель в заданных условиях использования и с определенным качеством. Показатели эффективности характеризуют степень приспособленности ПО к выполнению поставленных задач и являются обобщающими показателями оптимальности функционирования программного обеспечения [3].

В качестве целевой функции принимаем загруженность сервера от решаемых задач при поддержании определенного уровня надежности

$$E = \min Z(T,R),\ T \geq T_{мин.},\ R \geq R_{зад.}, \qquad (1)$$

где E – целевая функция минимизации загруженности;

Z – загруженность сервера;

T – выполняемые задачи с помощью сервера;

$T_{мин.}$ – минимальное количество задач, выполняемых сервером;

$R_{зад}$ – заданная надежность;

R – надежность.

В данном случае надежность определяется временем безотказной работы [5]. Функция нацелена на оптимизацию загруженности сервера при увеличении количества решаемых задач с использованием серверных ресурсов. Поскольку возможны проблемы с задержкой и перезагрузкой сервера, необходимо учитывать ряд требований к аппаратно-программной части [4].

Оптимальные требования к аппаратной части: процессор Intel или AMD, количество ядер 4–5; частота процессора 2 ГГц; оперативная память порядка 4 Гб; жесткий диск объемом 4–10 Тб, две сетевые карты.

Так как разрабатываемый программный комплекс предназначен для виртуализации серверов, он не должен сильно влиять на загрузку процессора и оперативной памяти.

Определим требования к ресурсам для установки виртуальной машины: минимальный объем занимаемого места на диске – 50 Мб; минимальный объем занимаемой оперативной памяти – 150 Мб; минимальная системная конфигурация – серверный набор микросхем материнской платы (для Intel или AMD), две сетевые карты, 4–8 Гб оперативной памяти с ECC.

Для обеспечения функции распределения нагрузки по трафику между одним из виртуальных серверов и маршрутизатором необходимо разработать дополнительное ПО, которое должно обеспечивать автоматическое переключение каналов по приоритету. Функциональные требования к разрабатываемому программному средству: полная совместимость с оборудованием Cisco; автоматическое переключение маршрутизации; возможность установки приоритета пути; мониторинг состояния маршрута [5].

Для разработанного программного средства необходимо следующее программное и аппаратное обеспечение: BSD система, Perl, Webmin, FreeVRRPd. Во всем программном обеспечении должны использоваться версии под выбранную операционную систему. Оптимальные требования к аппаратному обеспечению (для установки виртуальной машины): процессор от 2 ГГц; ОЗУ 1024 Мб; 20 Гб свободного места на диске; не менее двух сетевых интерфейсных карт.

Сборка ядра ОС

Опишем процесс сборки ядра операционной системы. Сборка ядра проводится в каталоге /usr/src/sys, который содержит множество подкаталогов, представляющих собой различные части ядра, но наиболее важным является каталог arch/conf, в котором редактируется конфигурационный файл ядра и в котором находится каталог compile, где собирается ядро. Все, что находится внутри каталога определенной архитектуры, относится только к этой архитектуре; остальной код является машинно независимым и общим для всех платформ, на которые FreeBSD может быть потенциально портирована.

Сначала конфигурируем ядро. Устанавливаем в ядро NAT, IP-FW, протокол NETGRAPH. NAT (от англ. Network Address Translation – преобразование сетевых адресов) – это механизм в сетях TCP/IP, позволяющий преобразовывать IP-адреса транзитных пакетов. IP-FW – это системное устройство, которое позволяет выполнять фильтрацию, переадресацию и другие операции с IP-пакетами, проходящими через системный интерфейс. Выбор пакетов производится путем применения упорядоченного списка правил образцов к каждому пакету до тех пор, пока не будет найден подходящий, после чего выполняются соответствующие действия по передаче этого пакета. Netgraph – модульная сетевая подсистема ядра FreeBSD, основанная на принципе графов. В Netgraph строится граф из узлов различных типов, узел каждого типа имеет некоторое количество входов/выходов. Узел Netgraph позволяет производить определенные действия над пакетом, проходящим через него.

Для сборки ядра LENA-KERNEL необходимо наличие всех исходных файлов FreeBSD. Алгоритм сборки ядра:

1. Перейти в каталог /usr/src – # cd /usr/src;
2. Сборка ядра – # make buildkernel KERNCONF = LENA-KERNEL;
3. Установка нового ядра – # make installkernel KERNCONF = LENA-KERNEL.

По умолчанию, при построении ядра все модули ядра также будут пересобраны. Если необходимо обновить ядро быстрее или построить только определенные модули, то нужно отредактировать файл /etc/make.conf перед началом процесса сборки ядра. MODULES_OVERRIDE = linux acpi sound/sound sound/driver/ds1 ntfs – эта переменная устанавливает список модулей, которые нужно построить вместо

построения всех модулей. WITHOUT_MODULES = linux acpi sound ntfs – в этой переменной перечисляются основные модули, которые необходимо исключить из процесса сборки.

Новое ядро будет скопировано в каталог /boot/kernel как /boot/kernel/kernel, а старое перемещено в /boot/kernel.old/kernel. После этого перезагружаем систему для того, чтобы использовать новое ядро.

Устанавливаем дополнительные пакеты, необходимые для работы:

- Webmin – программный комплекс, позволяющий администрировать операционную систему через веб-интерфейс, в большинстве случаев позволяя обойтись без использования командной строки и запоминания системных команд и их параметров. Используя любой браузер, администратор сервера может создавать новые учетные записи пользователей, почтовые ящики, изменять настройки служб и сервисов (например, веб-сервера Apache, DNS);
- Midnight Commander (MC) – один из файловых менеджеров с текстовым интерфейсом типа Norton Commander для Unix-подобных ОС;
- GNU Zebra – пакет программного обеспечения для Unix-подобных операционных систем, созданный Проектом GNU и позволяющий превратить обычный компьютер в высокопроизводительный TCP/IP-маршрутизатор.

Устанавливаем FreeVRRPd для балансировки нагрузки. VRRP (Virtual Router Redundancy Protocol) – сетевой протокол, предназначенный для увеличения доступности маршрутизаторов, выполняющих роль шлюза по умолчанию. Это достигается путем объединения группы маршрутизаторов в один виртуальный маршрутизатор и назначения им общего IP-адреса, который и будет использоваться как шлюз по умолчанию для компьютеров в сети.

Устанавливаем язык программирования Perl для написания программы переключения маршрутизаторов.

В заключение создаем образ сборки: создаем папку для образа (например, # mkdir /home/backup/); выполняем создание образа (# dump -0 -L -f – / | gzip -8 > /home/backup/rootimg.img.gz); копируем данный образ на любой носитель (USB, записать на DVD).

Полная установка сборки состоит из двух частей: установки чистой системы FreeBSD и установки самой сборки.

Вывод

Разработанное программное обеспечение предназначено для повышения эффективности функционирования севера отдела информационных технологий организации, увеличения решаемых задач, повышения производительности сервера за счет внедрения технологии виртуализации и рационального управления ресурсами, а также разработки программного средства автоматического переключения путей маршрутизации. Пользователям предоставляется для удобства работы расширенная версия виртуальной машины вне зависимости от специфики работы и используемого программного обеспечения.

Использование технологии виртуализации существенным образом позволяет снизить расходы на приобретение дополнительного оборудования, решать задачи улучшения и совершенствования ИТ-инфраструктуры. Существующие на сегодняшний день решения по виртуализации ИТ-инфраструктуры дают возможность существенно улучшить эффективность использования оборудования.

ЛИТЕРАТУРА:

1. Электронный ресурс https://www.vmware.com
2. **Сафонов В.О.** Основы современных операционных систем. – М.: БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий, 2011. 358 с.
3. **Бройдо В.Л.** Вычислительные системы, сети и телекоммуникации. – СПб: Питер, 2003. 688 с.
4. **Ряполова Е.И.** Аппаратно-программные требования для реализации VPN концентратора высокой емкости // Norwegian Journal of development of the International Science. 2017. Vol. 1. № 3. С. 90–91.
5. **Ряполова Е.И.** Выбор платформы виртуализации для создания VPN концентратора // Znanstvena misel journal. 2017. № 3. С. 107–109.
6. **Ряполова Е.И., Студянникова М.** А., **Цветкова К.Е.** Виртуализация как увеличение функциональности сервера // Вестник связи. 2016. № 1. С. 8–11.
7. **Гультяев А.** Виртуальные машины – несколько компьютеров в одном. – СПб: Питер, 2006. 224 с.
8. Kieren Diment, Matt Trout. The Definitive Guide to Catalyst: Writing Extensible. – O'Reilly, 2009. 780 p.