

Тема 1.3. Виды серверного программного обеспечения (теория – 18ч; практика – 6ч)

Виды серверов и их функции

В зависимости от функций, сервера классифицируются на следующие виды:

1. **Сервер рабочей группы** – это однопроцессорная система начального уровня, которая устанавливается и располагается в офисах небольших фирм. Обычно функция такого сервера заключается в хранении общей базы данных.
2. **Сервер контроллер** домена необходим тем фирмам, в которых количество сотрудников превышает 20 человек. Функции этого сервера заключаются в централизованном управлении сетевыми и файловыми ресурсами фирмы.
3. **Прокси сервер** предоставляет общий доступ в Интернет, а также обеспечивает безопасную работу в глобальной сети всем сотрудникам фирмы.
4. **Сервер электронной почты** – это отдельный узел, который предназначен для обработки электронной почты фирмы с числом сотрудников 30-40 человек.
5. **Веб сервер** пользуется большим спросом среди крупных компаний, у которых есть свой сайт в Интернете. Этот отдельный узел для веб-приложений предоставит доступ не только сотрудникам фирмы, но и большому количеству посетителей.
Веб-сервер — это сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответы, как правило, вместе с HTML-страницей, изображением, файлом, медиа-поток или другими данными.
6. **Терминальный сервер** обеспечивает доступ в Интернет различным удаленным офисам, мобильным пользователям, сотрудникам, работающих в домашних условиях и прочим пользователям глобальной сети. Пользователь, введя свои учетные данные, соединяется через выделенный канал связи с терминальным сервером, попадая при этом на виртуальный свой рабочий стол либо документ. **Терминальный сервер, сервер терминалов** — сервер, предоставляющий клиентам вычислительные ресурсы (процессорное время, память, дисковое пространство) для решения задач. Технически **терминальный сервер** представляет собой очень мощный компьютер (либо кластер), соединенный по сети с терминальными клиентами — которые, как правило, представляют собой маломощные или устаревшие рабочие станции, либо специализированные решения для доступа к терминальному серверу.
7. **Сервер базы данных** предназначен для обработки данных, которые управляются такими средствами, как Oracle, Apache, MySQL и др. **Сервер БД** выполняет обслуживание и управление базой данных и отвечает за целостность и сохранность данных, а также обеспечивает операции ввода-вывода при доступе клиента к информации.
8. **Файл-сервер** — это выделенный сервер, предназначенный для выполнения файловых операций ввода-вывода и хранящий файлы любого типа. Как правило, обладает большим объемом дискового пространства, реализованном в форме RAID-массива для обеспечения бесперебойной работы и повышенной скорости записи и чтения данных. **Файловый сервер** используется для

организации и структурированного хранения личных данных пользователей, учитывая при этом политику безопасности и доступ.

9. **Сервер приложений** — это программная платформа, предназначенная для эффективного исполнения процедур (программ, механических операций, скриптов), которые поддерживают построение приложений. Сервер приложений действует как набор компонентов, доступных разработчику программного обеспечения через API (Интерфейс прикладного программирования), который определен самой платформой. **Серверы приложений** оснащены расширенными возможностями обработки информации. Одним из таких серверов является беспроводной узел, представленный в виде типичного Web-сервера или сервера приложений, который знает, как необходимо передавать документы, которые составлены на стандартном языке, характерной для беспроводных устройств. Обычно в роли языка выступает Wireless Markup Language
10. **Брандмауэры** – это защитный экран от вредного воздействия из всемирной сети. Таким образом, такой сервер обеспечивает прохождение исходящих данных, и анализ поступающей информации в сеть. Зачастую прокси-сервер выполняет функцию защиты, отвергая при этом, либо принимая определенные виды сетевых запросов, которые поступают из локальной сети и Интернета.
11. **Сервер DHCP** позволяет изменить конфигурацию локальной сети в случае ее расширения, добавления либо удаления машин, к примеру, портативных персональных компьютеров.
12. **Серверы FTP** отвечают за перемещение файлов в Интернете. Этот сервер является обычным компьютером с выходом в Интернет, где и содержатся общедоступные файлы. Для копирования файла на ПК понадобится специальный протокол передачи данных, то есть FTP программа.(FTP-клиент)
13. **Принт-серверы** предоставляют возможность всем подключенным к сети ПК распечатать необходимые документы на одном конкретном, либо нескольких общих принтерах.
14. **Факс-сервер** – это факс-аппарат, позволяющий любому подключенному к данному серверу компьютеру, отправить факс-сообщение, не распечатывая его при этом на бумаге.
15. **Домашний сервер** – это резервное хранилище данных. Кроме этого, такой узел предоставляет пользователям иных компьютеров доступ к данным, размещенным на жестком диске. Для этой цели используется обычный ПК.

Управляющий сервер

Функции управляющего сервера могут быть реализованы либо через одну, либо через группу рабочих станций.

- Управляющий сервер обеспечивает единую точку доступа к ресурсам кластера. Доступ возможен либо через его консоль, либо удаленно через средства безопасного, удаленного доступа к управляющему серверу.
- На управляющем сервере функционируют программные средства, обеспечивающие мониторинг и управление ресурсами кластера, как с консоли, так и удаленно.
- Если это необходимо управляющий сервер может осуществлять дополнительные функции файлового сервера, быть станцией резервного хранения данных и т.д.

Дисковый сервер

Средства работы с файлами произошли от средств для работы с дисками, предоставляемыми дисковым сервером. Дисковый сервер — центральное хранилище файлов и данных, подключенное к сети таким же образом, как и любой узел. Поскольку он фактически является общедоступным жестким диском совместного использования, то каждый клиент сети может задавать у себя соответствующее этому диску имя так, как если бы этот диск был локальным. При чтении или записи на диск операционная система обращается (рис. 12.1) к таблице FAT, которая локализует данные на диске по номерам соответствующих им кластеров.



Рис. 12.1. Клиент открывает файл, хранящийся на дисковом сервере

Вероятно, вы уже заметили потенциальные проблемы такого подхода. В конечном итоге задача сервера состоит в предоставлении всем клиентам сети доступа к жесткому диску, не правда ли? Но поскольку файлы на жестком диске создаются и удаляются, его таблица, поддерживаемая FAT, будет изменяться. Удаление файлов приводит к появлению на диске свободных кластеров, в которых могут быть сохранены вновь создаваемые файлы. В файловой системе FAT запись файла всегда начинается с первого свободного кластера. Таким образом, после достаточно длительной работы нескольких клиентов таблица жесткого диска может совершенно измениться. Таблица, содержимое которой было корректно по состоянию на 8 часов утра, может стать совершенно неточной к 4 часам вечера, что может замедлить и даже полностью приостановить поиск данных.

Для решения этой проблемы диск сервера обычно разделяют на несколько томов (по одному для каждого пользователя), либо в него устанавливают съемные диски, каждый из которых доступен только одному пользователю. Серверы могут применяться также и для других целей. Например, можно использовать дисковый сервер для предоставления доступа к гибкому диску пользователям бездисковой рабочей станции. Однако хотя дисковые серверы и не исчезли совершенно из употребления, они сегодня не используются так широко, как файловые.

Файловые серверы

Вместо организации совместного использования физических дисков, файловый сервер предоставляет клиентам сети совместный доступ к области хранения данных. Как указывалось выше, системы для работы с файлами функционируют так же, как и системы для работы с дисками.

- Файловый сервер предоставляет совместный доступ пользователям всей сети к определенному тому.

- Клиент использует имя диска (или путь Unicode) для доступа к совместно используемому тому.
- Клиент может читать и писать в совместно используемый том.

С точки зрения клиента обращение к совместно используемому тому подобно обращению к локальному жесткому диску, за исключением одного: если сервер, на котором хранится совместно используемый том, будет отключен от сети или приостановит совместное использование данного тома, то клиент не сможет к нему обратиться.

Как работает файловый сервер

Хотя файловые и дисковые серверы могут показаться клиенту совершенно одинаковыми, это вовсе не так. Разница заключается, в основном, в распределении обязанностей. Отличие между дисковыми и файловыми серверами можно выразить следующим образом. Когда сетевой клиент первый раз после перезагрузки запрашивает у дискового сервера файл, тот просматривает свое содержимое и находит "карту" склада, хранящего данные. После этого дисковый сервер говорит клиенту: "Вперед, малыш, забирай свой файл, но получи себе еще и карту. Я предпочитаю заниматься более интересными вещами, поэтому обеспечивай себя сам с помощью этой карты". Каждый раз, когда клиент получает что-либо со склада или что-либо возвращает в него, порядок на этом складе немного изменяется, но поскольку при этом выполняется автоматическое обновление, то карта, предоставленная дисковым сервером, будет точна. Однако когда клиентский компьютер через некоторое время снова обратится к серверу для поиска файлов с помощью своих старых карт, очень может быть, что склад будет уже организован по-другому, и клиент не сможет ничего найти.

В отличие от дискового, файловый сервер будет искать требуемое сам, не разрешая толпе сетевых клиентов обшаривать свой собственный жесткий диск. Когда рабочая станция запросит у него файл, файловый сервер ответит ему "Я должен переслать эти данные — уж лучше я сделаю все сам". После этого он передаст запрос драйверу файловой системы, который найдет файл и pošлет информацию о его местонахождении клиентскому приложению, после чего оно откроет этот файл. Клиент никогда не получит копию FAT-таблицы диска сервера, хотя и использует серверные средства обработки такого запроса. Поэтому в файловом сервере существует только одна копия FAT-таблицы, всегда соответствующая текущему моменту времени.

Обслуживание файловых операций является одной из наиболее распространенных функций сетевого сервера, позволяющих решить несколько очень полезных задач. Во-первых, с помощью соответствующих средств обеспечивается централизация хранения данных для облегчения их последующего архивирования. Во-вторых, файловые серверы предоставляют доступ к одному файлу сразу нескольким пользователям. Конечно, во многих случаях клиентский компьютер может предоставить весь свой жесткий диск для совместного использования всей сетью, так что локально хранимые файлы становятся доступными всей сети (что обычно делается в одноранговой сети). Однако при этом могут возникнуть проблемы, если, например, сразу 40 человек в сети предоставят некоторую часть своих жестких дисков для совместного использования всей остальной сети. Это может привести к тому, что отдельные файлы найти будет чрезвычайно сложно.

Файловые серверы имеют только один недостаток: такой сервер должен все время находиться в режиме оперативного доступа, и содержимое его

жесткого диска должно регулярно архивироваться, в противном случае он становится хуже, чем бесполезным. Если файлы хранятся на клиентских компьютерах, то при отключении одного из них все остальные еще могут продолжать работу. Но если выключится файловый сервер, работа сети будет парализована. Поддержка операций архивирования и необходимость обеспечения надежности функционирования являются критическими требованиями при организации работы хорошего файлового сервера.

Оборудование файлового сервера

Что же требуется для создания хорошего файлового сервера? Емкие и надежные жесткие диски для хранения информации, достаточный объем оперативной памяти, позволяющий эффективно обслуживать запросы на открытие и сохранение файлов, а также надежная система архивации. Все это и рассмотрено в последующих разделах.

Требования к жесткому диску

Диски должны быть быстрыми, с набором соответствующих аппаратных функций. Возвратившись к описанию жестких дисков, вспомним, что наилучшим типом жестких дисков для файлового сервера почти наверняка являются диски с интерфейсом SCSI. Они дороже дисков EIDE, но быстрее обслуживают множественные запросы и удобнее в использовании, поскольку их легко добавлять в SCSI-цепочку.

Примечание

Хорошая сетевая операционная система должна содержать набор средств для расширения томов, т.е. должно быть предусмотрено добавление нового дискового пространства к ранее созданному разделу без изменения структуры логических дисков так, чтобы два и более физических дисков могли рассматриваться в виде одного тома.

Сколько нужно памяти

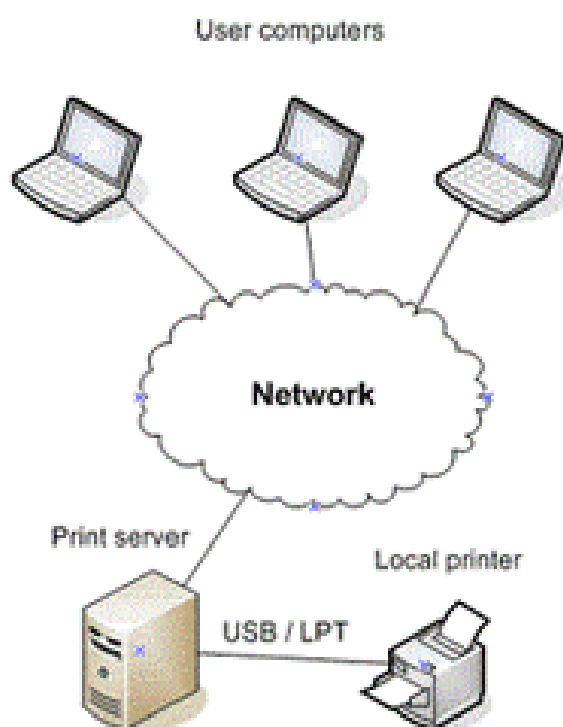
Требуется достаточный объем оперативной памяти для поддержки всех запросов чтения и записи файлов, поступающих на сервер. Сколько для этого нужно памяти? Ответить на этот вопрос нелегко. При работе сервера приложений (который будет рассмотрен в конце этой главы) можно воспользоваться эмпирическим правилом, в соответствии с которым следует предоставлять каждому пользователю 4—8 Мбайт оперативной памяти. Но файловый сервер работает не так напряженно, как сервер приложений, поскольку работа файлового сервера состоит всего лишь в обработке запросов чтение/запись. После открытия файла память файлового сервера уже не расходуется, в то время как сервер приложений размещает файлы приложения в своей собственной памяти.

Защита файлов

Вам потребуется система архивирования. Основная причина использования файлового сервера в качестве главного хранилища информации — это возможность эффективной защиты данных компании (или ваших личных данных, если сервер — часть домашней сети). Если данные не являются важными, они, вероятно, и не должны находиться в главном хранилище. Методы архивации не обязательно должны быть самыми передовыми, однако вы должны их знать и уметь восстанавливать с их помощью архивированные данные при повреждении жесткого диска.

Архивирование — весьма важная операция для большинства типов серверов, но для файлового сервера она, безусловно, является жизненно важной.

Принт-сервер



Принт-сервер (от англ. *print server* - сервер печати) - устройство, позволяющее группе пользователей проводных и беспроводных сетей совместно использовать принтер.

Развитие технологии

Летом 1991 г. компанией Hewlett-Packard впервые была представлена технология JetDirect, позволяющая непосредственно подключать принтер к локальной вычислительной сети. Первым ее представителем была интерфейсная карта XIO (Extended Input/Output), поддерживающая стандарт Ethernet и разнообразные сетевые протоколы (TCP/IP, IPX/SPX, AppleTalk и DLC/LLC). Первоначально для каждого протокола требовалась своя карта, в ходе последующего развития добавлялись новые типы разъёмов. В 1992 г. была представлена карта с коннекторами RJ45 и BNC, а в 1993 г. - первое **внешнее устройство** с параллельным портом (LPT). Это позволило

подключать карты JetDirect практически к любому принтеру, делая его сетевым.

Увеличение спектра задач в области печатной продукции и расширение ассортимента цифровых устройств привело к созданию технологии NetUSB, которая представляет собой эмулятор USB-интерфейса по сети. Впервые технологию NetUSB начала применять компания TrendNet (2005 - 2007). За ней последовала компания D-Link, выпустив своё решение (SharePort). Использование любой из этих технологий выглядит так, словно принтер подключен напрямую к компьютеру. Данные технологии позволяют использовать подобным образом и другие устройства: МФУ, сканер, флэш-карты, кард-ридер, внешний жесткий диск, фотоаппарат или видеокамеру. На сегодняшний день выпуском принт-серверов занимается любая компания, производящая сетевое оборудование.

Общие сведения

Принт-сервер является устройством, подключаемым к локальной сети, к которому подключаются один или несколько принтеров (количество подключаемых аппаратов определяется типом устройства). Существует два типа принт-серверов: внутренние и внешние. Внешние принт-серверы в большинстве случаев работают с любыми аппаратами, вне зависимости от компании производителя, а внутренние - исключительно с принтерами производителя принт-сервера. Но, вне зависимости от типа принт-сервера, он является "прозрачным" для операционных систем и требует только корректной настройки параметров для используемых протоколов передачи данных в сети (TCP/IP , Internet Printing Protocol, Line Printer Daemon Protocol, NetWare, NetBIOS/NetBEUI).

В основном модели принт-серверов отличаются типом и количеством портов используемых для подключения печатающих устройств, скоростью работы по сети, габаритами, а также спектром сетевых протоколов, которые он способен поддерживать и, как следствие, возможностью работы в "многооперационной" сети (то есть локальной сети, к которой подключены ПК с операционными системами различных типов).

Как правило, в комплекте с принт-сервером поставляется программное обеспечение, позволяющее настраивать параметры работы устройства и обладающее расширенными средствами диагностики и настройки, а также позволяющее выполнять подключение сетевого принтера на клиентском компьютере .

Все модели принт-серверов поддерживают настройку параметров через web-интерфейс, доступ к которому имеет любой компьютер локальной сети.

Принцип работы

Принт-сервером является компьютер или отдельное устройство, которое управляет работой печатающих устройств в одной сети. В зависимости от модификации, для подключения к сети принт-сервер использует либо порт

10/100BASE-T Ethernet, либо подключается с помощью беспроводного канала связи (IEEE 802.11 b/g/n). Для подключения принтера имеются один или несколько портов LPT или USB. Для того, чтобы принт-сервер мог нормально функционировать, он должен быть подключен к сети и к нему должен быть подключен принтер.

Принцип работы принт-сервера **очень прост** - производится сбор запросов на печать, поступающих по сети к принтеру от разных компьютеров, и передача по USB-порту информации на принтер. В результате чего, принтер остается формально подключенным к одному устройству, хотя на самом деле он находится в сети.

Принт-сервер поддерживает протокол преобразования данных печати, что позволяет конвертировать данные, посланные с клиентских компьютеров, в вид, понятный для печатного устройства. Это значит, что принт-сервер "понимает" различные сетевые протоколы, которые могут быть использованы в одно и то же время на различных операционных системах.

Итоги современного развития и перспективы

В настоящее время в корпоративной среде, как правило, внедряются принтеры со встроенными принт-серверами, поэтому, как отдельное устройство принт-сервер не имеет ярко выраженной актуальности в будущем. С другой стороны, в домашнем сегменте принт-сервер находит широкое применение, но сейчас появляются устройства, объединяющие в себе функции маршрутизатора и принт-сервера, что снижает востребованность последнего.

Ведущие производители

Hewlett-Packard

TP-Link

D-LINK

TrendNet

Почтовый сервер

Почтовые серверы – это серверы, получающие и отправляющие электронные сообщения.

Сервер, получающий электронные сообщения, работает по протоколу POP (Post Office Protocol).

Сервер, отправляющий электронные сообщения работает по протоколу SMTP (Simple Mail Transfer Protocol).

Почтовый сервер, сервер электронной почты, мейл-сервер — в системе пересылки электронной почты так обычно называют агент пересылки сообщений (англ. *mail transfer agent, MTA*). Это компьютерная

программа, которая передаёт сообщения от одного компьютера к другому. Обычно почтовый сервер работает «за кулисами», а пользователи имеют дело с другой программой — клиентом электронной почты (англ. *mail user agent*, MUA).

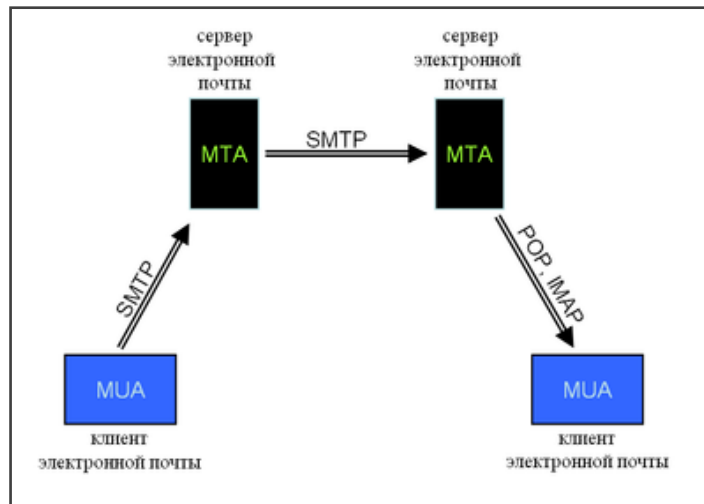


Схема взаимодействия

К примеру, в распространённой конфигурации агентом пользователя является Outlook Express. Когда пользователь набрал сообщение и посылает его получателю, почтовый клиент взаимодействует с почтовым сервером, используя протокол SMTP. Почтовый сервер отправителя взаимодействует с почтовым сервером получателя (напрямую или через промежуточный сервер — релей). На почтовом сервере получателя сообщение попадает в почтовый ящик, откуда при помощи агента доставки сообщений (mail delivery agent, MDA) доставляется клиенту получателя. Часто последние два агента совмещены в одной программе, хотя есть специализированные MDA, которые в том числе занимаются фильтрацией спама. Для финальной доставки полученных сообщений используется не SMTP, а другой протокол — часто POP3 или IMAP — который также поддерживается большинством почтовых серверов. Хотя в простейшей реализации MTA достаточно положить полученные сообщения в личный каталог пользователя в файловой системе центрального сервера («почтовый ящик»).

Помимо непосредственной передачи писем на почтовом сервере возможна реализация:

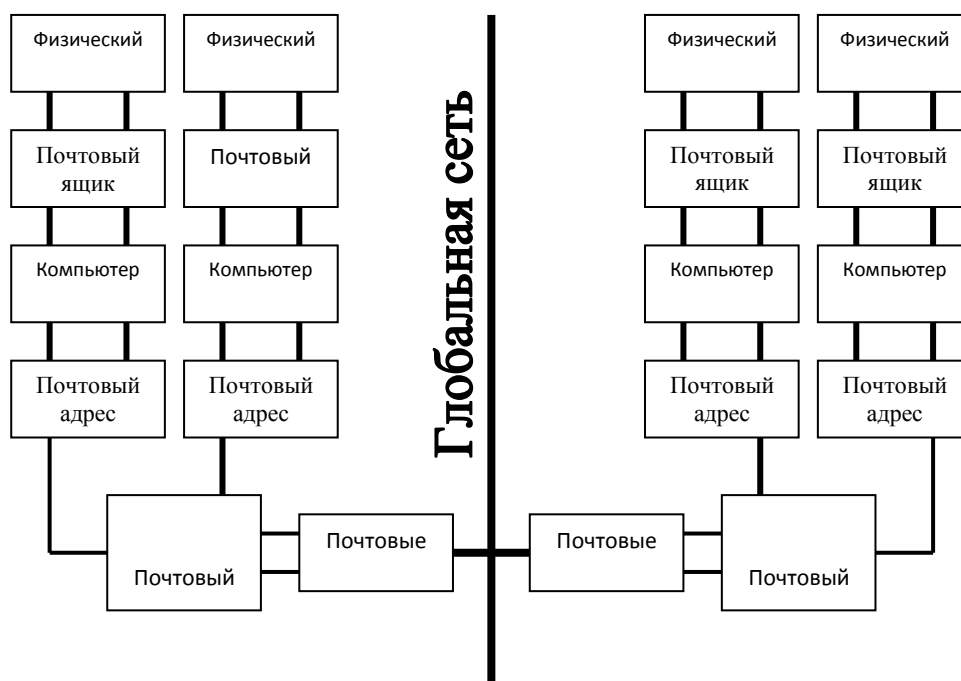
- учета почтового трафика
- осуществления антивирусной проверки почты
- изменение параметров почтового ящика для каждого пользователя
- осуществление резервного копирования поступающей почты
- осуществление спам-фильтрации почты, с возможностью обучения спам-фильтра

- обеспечение возможности задания квот на размер почтовых ящиков и т.д.

Электронная почта

Электронная почта (e-mail, от латинского "electronic mail").

Электронная почта, как и обычная, работает с системой электронных "почтовых отделений" – почтовых серверов, которые обеспечивают пересылку писем по глобальным сетям. Они взаимодействуют с помощью почтовых протоколов, обеспечивающих пересылку и распознавание передаваемой в сети информации. Компьютеры-клиенты почтовых серверов обслуживают пользователей электронной почты. Каждый получает свой почтовый адрес и свой "почтовый ящик" на этом компьютере, т.е. область памяти, а также пароль для доступа к нему.



С помощью почтовой программы можно создавать сообщения, считывать их с почтового сервера, работать с адресной книгой, хранить и организовывать письма в папках "почтового ящика", готовить файлы для пересылки и преобразовывать их в нужный формат после получения и др.

С помощью почтовой программы пользователь создает сообщение адресату, задает адрес, отправляет сообщение, для чего соединяется с почтовым сервером. Во время соединения почтовый сервер запрашивает имя пользователя и его пароль. Иначе сеанс связи не состоится. После

соединения подготовленная почта автоматически отправляется на сервер и далее через передачу от одного к другому почтовому серверу достигает адресата. Сразу после отправки корреспонденции автоматически происходит получение почты клиентом. Считанные в его область памяти сообщения и файлы сортируются и раскладываются по почтовым ящикам пользователей. Адресат при загрузке своего ящика видит разложенные по папкам сообщения: новые, старые, отправленные. Он может удалять, сортировать и классифицировать их по своему разумению.

Структура адреса электронной почты

При пересылке информации большое значение имеет адресация, поскольку без нее не может быть найден получатель. Все знают печальную историю Ваньки Жукова, который отправил письмо "на деревню дедушке". Адрес обычной почты оформляются по определенным почтовым правилам.

Существующие правила оформления электронных адресов иные. Адреса электронной почты имеют более четкую логическую структуру. Они состоят из иерархической последовательности доменов – частей, например:

user@gym2.spb.su

kimmeria@sch.spb.ru

sviend@comp.kiev.ua

Все адреса состоят из двух частей, разделенных символом @ (читается "эт"). При прочтении слева направо до этого знака отображаются имена пользователей (получателей). Это может быть имя начальника почтового отделения - "пост мастера" (post master), выдуманные или истинные имена пользователей электронной почты, на которые приходит корреспонденция. Их может быть зарегистрировано на одном и том же компьютере много. Часть адреса, находящаяся справа от @, определяет компьютер, подключенный к сети, город и страну или название сети, в которой пользователь зарегистрирован. Адреса делятся на части, которые называются доменами.

Что такое домен

Рассматривая домен справа налево и разбив его по точкам на отдельные слова, получим поддомены, поочередно уточняющие, где этот почтовый ящик искать. В аналогии с обычной почтой домен – это адрес (строка "Куда" на конверте), а поддомены - название страны, города, улицы, номер дома.

Домен не описывает путь, по которому следует передавать сообщение, а только объясняет, где находится адресат; точно так же адрес на почтовом конверте - это не описание дороги, по которой должен идти почтальон, чтобы доставить письмо, а место, в которое он должен в конце концов его принести. В обоих случаях почтовые службы сами выбирают маршрут из соображений экономии времени и денег. Обычно существует

несколько путей, по которым можно доставить сообщение в указанное место, и, отправляя письмо, Вы не знаете, по какому из путей оно на этот раз пойдет.

Самый правый поддомен (в нашем случае ru) называется доменом верхнего уровня и чаще всего обозначает код страны, в которой находится сервер. Код ru - это Россия, kz – Казахстан. Каждый код состоит из двух латинских букв. Например, код uk обозначает Великобританию, и почтовый ящик с адресом mathew@montis.co.uk следует искать в английской сети JANET.

Домен верхнего уровня - не всегда код страны. В Соединенных Штатах встречаются такие, например, домены верхнего уровня, как edu - научные и учебные организации, или gov – правительственные учреждения:

lamaster@george.arc.nasa.gov

Если почтовая служба видит в правой части домена поддомен такого вида, она уже знает, что адресат находится в США, поэтому код страны us не нужен. Такие обозначения сложились в американской научной сети ARPANET еще до того, как ее связали с сетями в других странах, а сейчас они сохраняются только по привычке. Как правило, во все места, которые адресуются по типу организации, можно добраться и используя код страны. Из соображений простоты и единообразия лучше пользоваться адресами с кодами стран.

Обычно такие адреса используются, если эта сеть понимает адреса в формате, отличном от RFC822. Тогда Вы пишете адрес типа имя@машина.сеть, а мост между Вашей сетью и сетью адресата преобразует его к нужному виду.

Поддомены, расположенные правее домена верхнего уровня, уточняют положение адресата внутри этого домена (внутри России для ru, среди военных организаций США для mil, или в сети BITNET для bitnet). К примеру, в адресе avg@hq.demos.ru поддомен demos обозначает организацию внутри России, а hq – группу машин внутри demos.

В адресе lamaster@george.arc.nasa.gov домен верхнего уровня gov означает, что адресат находится в одном из правительственных учреждений США, первый поддомен nasa уточняет, в каком именно - NASA, второй поддомен arc называет подразделение NASA - Ames Research Center, а george указывает на конкретную машину в этом подразделении.

Если письмо адресуется по имени сети, в которую его надо послать, адрес (домен) состоит только из домена верхнего уровня - имени сети и еще одного поддомена - имени машины в этой сети. Разбираться, где находится данная машина, выпадает на долю почтовых служб этой сети.

Когда необходимо достичь адреса, например, **ux.cso.uiuc.edu**, компьютер должен преобразовать его в адрес. Чтобы это сделать, Ваш компьютер начинает просить помощи у серверов (компьютеров) DNS, начиная с правой части имени и двигаясь влево. Сначала она просит локальные серверы DNS найти адрес. Здесь существуют три возможности:

– Локальный сервер знает адрес, потому что этот адрес находится в той части всемирной базы данных, которую курирует данный сервер.

– Локальный сервер знает адрес, потому что кто-то недавно уже спрашивал о нём. Когда Вы спрашиваете об адресе, сервер DNS(Domain Name System) некоторое время держит его «под рукой» на тот случай, если чуть позже о нём спросит ещё кто-нибудь. Это значительно повышает эффективность работы системы.

– Локальный сервер не знает адрес, но знает, как его определить.

Как локальный сервер определяет адрес? Его программное обеспечение знает, как связаться с *корневым* сервером, который знает адреса серверов имён домена высшего уровня (крайней правой части имени, например, **edu**). Ваш сервер запрашивает у корневого сервера адрес компьютера, отвечающего за домен **edu**. Получив информацию, он связывается с этим компьютером и запрашивает у него адрес сервера **uiuc**. После этого Ваше программное обеспечение устанавливает контакт с этим компьютером и спрашивает у него адрес сервера домена **csu**. Наконец, от сервера **csu** он получает цифровой адрес **ux**, компьютера, который и был целью данной прикладной программы.

Распространённые серверы электронной почты

- AfterLogic MailSuite Pro
- AMS server
- Apache James (Apache Java Enterprise Mail Server, в рамках Apache Software Foundation)
- CommuniGate Pro
- Courier Mail Server — свободный почтовый сервер для Linux, SMTP/POP3/IMAP
- Courier Mail Server — проприетарный почтовый сервер для Windows, SMTP/POP3
- Eserv — отечественный почтовый сервер (shareware), GPL с 2003 г., SMTP/POP3/IMAP
- Exim
- HMailServer
- Hula
- IBM Lotus Domino
- IceWarp Mail Server
- Kerio Connect
- LinuxWizard
- MDaemon Mail Server
- MailEnable
- Microsoft Exchange Server
- NextMail (русская разработка Корпорации НЕКСТ на базе qmail)
- Office Mail Server (свободный почтовый сервер для windows)
- Postfix
- Proofpoint
- qmail
- Sendmail
- SmarterMail
- UserGate Mail Server (русская разработка компании Entensys)

- Zimbra
- ZMailer

Программное обеспечение электронной почты

Для работы с электронной почтой необходимы программы, определяющие разные виды соединения с почтовым сервером. Эти программы работают с различными системными оболочками и обладают разными возможностями. Наиболее распространенные программы обмена электронной почтой обеспечивают автоматическую связь с сервером. Такая программа сама соединяется с сервером, отправляет и получает почту. Пользователь ждет результата. Он может работать с корреспонденцией в режиме отключения от сервера. Это экономически вполне обоснованный вариант работы, т.к. клиент оплачивает время соединения. При необходимости проверки почты в почтовом ящике или отправки сообщения пользователь опять может соединиться с сервером.

Почтовые программы Internet предоставляют возможность писать сообщения, находясь в сеансе с сервером, заказывать новости, сообщать информацию авторам Web-сайтов. Вы можете получать почту и одновременно осуществлять поиск информации в сети. Однако если вы платите за время связи с сервером, не стоит в это же время создавать сообщения.

Пакеты электронной почты выполняют такие функции, как подготовка и обработка текстов, чтение, удаление, пересылка корреспонденции, ввод адресов и тем сообщений, преобразование в нужный формат отсылаемых файлов. Любой пакет электронной почты имеет встроенный текстовый редактор для обработки сообщений. Многие программы обмениваются текстами в кодах ASCII. Однако распространены и другие кодировки текстов, например, KOI-8, WIN-95. Сообщения на национальных языках, передаваемые с помощью различных кодировок, не всегда читаются в других системах. Тексты, создаваемые редакторами электронной почты, отправляются как сообщения (messages). Если же они создаются в других редакторах, нужно позаботиться, чтобы получатель смог их прочесть, и отправлять их как файл, а не как сообщение – только если вы уверены, что получатель может открыть их в таком же приложении.

Для запуска пакетов электронной почты Internet Mail, Outlook Express, Exchange Mail и пр. необходимо найти пиктограмму программы и щелкнуть на ней указателем мыши.

Все почтовые сообщения, имеющие отношение к данному пользователю, хранятся в четырех папках: *Входящие*, *Исходящие*, *Отправленные*, *Удаленные*. После считывания с сервера почта поступает в папку *Входящие*. После прочтения остается там же, но уже помечается как прочитанное. В папке *Удаленные* хранятся те сообщения, которые были удалены пользователем. Ее необходимо очищать. В папке *Исходящие*

отображаются те сообщения, которые пользователь создал для отправки. В момент соединения с сервером письмо перемещается в папку *Отправленные*. Если сеанс связи был неудачным, то сообщение останется в папке *Исходящие*.

Mail.ru

Mail.Ru - главный коммуникационный портал российского Интернета. Его ежемесячная аудитория превышает 50 миллионов уникальных посетителей.

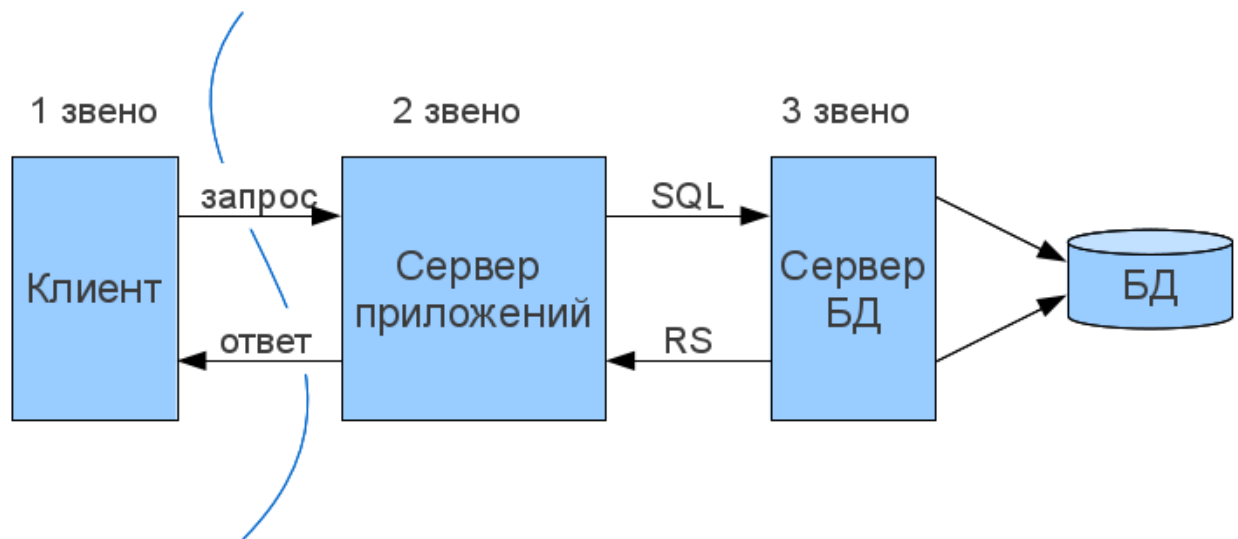
Mail.Ru занимает лидирующую позицию среди бесплатных почтовых сервисов, предоставляя своим пользователям почтовый ящик неограниченного размера с защитой от спама и вирусов, переводчиком, проверкой правописания, архивом для хранения фотографий и т.д. Через почтовые ящики Mail.Ru ежедневно проходит более 35 миллионов писем.

Mail.Ru – это не только почтовая служба, но и более 40 интернет-сервисов. Среди них социальная сеть Мой Мир@Mail.Ru, насчитывающая более 40 млн. зарегистрированных профайлов, крупнейший фотохостинг Фото@Mail.Ru, пользователи которого хранят около 490 млн. фото, проекты Видео@Mail.Ru, Блоги@Mail.Ru, Игры@Mail.Ru, Поиск@Mail.Ru, Авто@Mail.Ru, Недвижимость@Mail.Ru, Путешествия@Mail.Ru и многие другие.

Серверы приложений

Вынесение прикладной логики в отдельный уровень представляет разработчикам дополнительную гибкость в создании распределенных информационных систем. Размещение и выполнение программ на стороне сервера снижает требования к аппаратному обеспечению клиентов и уменьшает проблемы обеспечения совместимости в гетерогенной сетевой среде.

Сервер приложений — это сервисная программа, которая обеспечивает доступ клиентов к прикладным программам, выполняющимся на сервере. Сервер приложений обычно выделяется как среднее звено в трехуровневой клиент-серверной архитектуре (3-tier):



Модель "сервер приложений"

1. Первый уровень, интерфейсный, как правило, графический (GUI).
2. Средний уровень, исполнимый программный код, размещенный обычно на выделенном сервере.
3. Третий уровень, фоновый — базы данных. Сюда же относятся, унаследованные средства доступа к данным и управления транзакциями.

В сетевой среде сервер приложений является посредником между клиентами и серверами баз данных.

Бизнес-логика может быть реализована на стороне сервера как целиком (удаленный код), так и частично (распределенный код). В первом случае к серверу могут обращаться терминалы и «тонкие» клиенты и такое взаимодействие соответствует модели «сервер терминалов». «Толстые» и rich-клиенты могут получать компоненты серверного приложения и выполнять их на своей стороне (например javascript, апплеты, flash).

Мобильный софт

Задачи, решаемые серверами приложений хорошо иллюстрируются на примере мобильных сервисов. Возможности мобильных устройств изначально ограничены физическими размерами и временем автономной работы (остальные ограничения, в основном, вытекают из этих двух). Мобильное приложение разрабатывается с учетом этих ограничений, но так как софт для телефона должен быть адаптирован для использования на конкретной модели, то процесс разработки усложняется. Разделив мобильное приложение на клиентскую (представление данных) и серверную (прикладная логика) части, разработчик получает следующие возможности:

- урезанная по функционалу клиентская часть получается менее требовательной к ресурсам;
- для поддержки новых устройств нужно адаптировать только фронт-энд, не затрагивая прикладную логику;
- изменения в программе (расширение функциональности, исправление ошибок и т. п.) выполняются на сервере приложений и распространяются на всех клиентов.

Клиенты могут взаимодействовать с приложениями через API сервера (Java-клиент <—> контейнер сервлетов <—> сервлет). Большую гибкость и универсальность представляет взаимодействие через сторонние сервисы, в первую очередь — через веб-сервер.

Понятие сервера приложений традиционно связывают с платформой Java, указывая на то, что сервер Java-приложений представляет реализацию спецификации сервлетов, возможно в виде JSP, и еще некоторые сервисные услуги, в первую очередь соединение с базой данных.

Но это нечто большее и меньшее одновременно: *сервер приложений предоставляет среду, в которой прикладные программы могут работать, независимо от того, что и как именно они делают.*

Поэтому, чтобы ответить на вопрос, является ли (и в какой степени) некое сервисное ПО сервером приложений, стоит сравнить его заявленные функции со списком атрибутов, присущих этой категории:

- Предоставляет модель контейнера для приложений.
- Предоставляет сервисные услуги для программ.
- Обеспечивает управление приложениями и/или представляет средства их разработки.
- Соответствует индустриальным спецификациям и стандартам.
- Добавим сюда же обслуживание веб-страниц, ввиду реальной востребованности технологий на основе WWW.

Реализации

По приведенным признакам в рассматриваемую категорию попадают, например, традиционные терминал-серверные системы, технология CGI, контейнеры Java-сервлетов и др.

Унаследованные решения

Серверы терминалов представляют среду для удаленного выполнения программ, в качестве которой выступает сама операционная система. Доступ

к ним осуществляется по протоколам удаленного управления (telnet, ssh, RDP, VNC и т. п.) из клиентского ПО (эмулятор терминала, средства управления удаленным рабочим столом и т.п.). Управление запущенной программой выполняется через эмулируемый на клиенте пользовательский интерфейс (текстовый или графический) операционной системы. На серверной стороне взаимодействие программ с ОС реализуется через системные вызовы. Управление также осуществляется средствами операционной системы. Разработка может вестись на любом языке, доступном в конкретной ОС.

Общий шлюзовый интерфейс (CGI) — технология доступа к приложениям через веб-сервер. Отличия от сервера терминалов здесь в том, что пользовательский интерфейс предоставляется в виде веб-страниц. Запросы веб-клиентов, обращенные к программам, размещенным в выделенном каталоге (как правило cgi или cgi-bin) перенаправляются на их вход через стандартный поток ввода (stdin). Результаты выполнения в виде гипертекста приложение возвращает веб-серверу через stdout.

Серверы Java-приложений

Платформа Java является индустриальным стандартом, позволяющим создавать из унифицированных компонентов интероперабельные программные решения для самых разных систем, в которых может быть запущена виртуальная машина Java (JVM).

Контейнер сервлетов — один из архитектурных компонентов J2EE, представляющий окружение для выполнения сервлетов. Сервлет — это Java-приложение, выполняющееся на стороне сервера (в отличие от апплета). Контейнер сервлетов может работать как полноценный самостоятельный сервер, но чаще используется (интегрируется) совместно с другим серверным ПО. Обеспечивает обмен данными между сервлетом и клиентами, берёт на себя выполнение таких функций, как создание программной среды для функционирующего сервлета, идентификацию и авторизацию клиентов, организацию сессии для каждого из них.

Концепция сервлет-контейнера позволяет создавать как универсальные, так и специализированные серверы приложений (например, для мобильных сервисов).

Примером реализации контейнера сервлетов является Apache TomCat, который используется в таких серверах приложений как Apache Geronimo, JBoss, GlassFish, IBM WebSphere Application Server (WAS).

Другие решения

Компания Microsoft представляет собственные решения для поддержки бизнес-логики и сервисной инфраструктуры на основе ОС Windows Server и технологии .NET Framework. Основным средством разработки является язык C#.

Язык python, получивший популярность во многом благодаря Google, является основным средством разработки для сервера веб-приложений Zope.

Для сценариев на языке PHP, широко используемом для создания веб-сайтов, компания Zend Technologies (разработчик самого языка PHP) создала сервер приложений Zend Server.

Серверы приложений: плюсы и минусы

Преимущества

Целостность кода и данных

Размещение бизнес-логики на выделенном сервере или ограниченном числе серверных компьютеров гарантирует доступ к обновленному и модернизированному ПО для всех клиентов. Это исключает риск доступа и управления данными из устаревших и, возможно, несовместимых программ.

Централизованное управление

Изменения в конфигурации прикладных программ, такие как, например, смена сервера баз данных, выполняются централизованно.

Безопасность

Централизованные средства, через которые поставщик услуг (сервис-провайдер) может управлять доступом к данным и компонентам приложения, позволяют выполнять проверку подлинности потенциально ненадежных клиентов в среднем слое и не затрагивать уровень базы данных.

Производительность

Сервер приложений может решать задачи балансировки сетевого трафика и распределения нагрузки между другими физическими серверами системы.

Общая стоимость владения

Совокупность перечисленных выше преимуществ, а в дополнение к ним перераспределение затрат на оборудование с клиентской на серверную сторону, может привести к экономии средств для организации. Так же на снижении общей стоимости владения может отразиться практика аренды

программного обеспечения. Справедливости ради нужно отметить, что стоимость самого серверного ПО, а также затраты на его внедрение и сопровождение могут быть весьма высокими.

Недостатки

Централизация

Системы, построенные на основе сервера приложений, имеют один основной недостаток, присущий всем централизованным решениям — «падение» сервера приведет к недоступности программ для всех клиентов. К тому же эффекту приведут и неполадки в сетевом подключении.

Защита информации

Эта проблема, в принципе, актуальна для любых сетевых решений, использующих для передачи данных инфраструктуру публичных сетей.

Роль сервера приложений

Сервер приложений является расширенной версией сервера в операционной системе Windows Server® 2008. Новая версия сервера приложений предоставляет интегрированную среду для развертывания и выполнения

пользовательских дел на сервере приложений. Эти приложения отвечают на запросы,

поступающие по сети от удаленных клиентских компьютеров или из других приложений.

Как правило, для развертывания и запуска на сервере приложений пользователь производит одно или более из следующих действий:

- Internet Information Services (IIS) (протокол передачи гипертекста (HTTP) сервер, встроенный в Windows Server)
- Microsoft® .NET Framework версии 3.0 и 2.0. (Если у вас есть приложения, построенные с .NET Framework 3.5, вы можете загрузить и установить .NET Framework 3.5 на операционной системы.)
- ASP.NET
- COM +
- Очереди сообщений
- Веб-службы, которые построены с Windows Communication Foundation (WCF)

Мы рекомендуем использовать роль сервера

приложений Windows Server 2008 выполняемую приложениями, которые зависят от

служб роли или компонентов, которые являются частью комплексной роли сервера приложений и выбрать во время процесса установки. Примером может быть определенная конфигурация Microsoft BizTalk® Server, которая использует набор

служб роли или компонентов, которые являются частью среды сервера приложений.

Обычно роль сервера приложений необходима при развертывании бизнес-приложения в рамках вашей организации (или разработанного независимый поставщик программного обеспечения (ISV) для вашей организации), и когда разработчик указал, что конкретная роль службы требуется. К примеру ваша организация может иметь приложение обработки заказа, доступ к записи клиентов, которые хранятся в базе данных. Приложение получает доступ к сведения о клиенте через набор веб-служб WCF. В этом случае можно настроить один компьютер Windows Server 2008 в качестве сервера приложений и базы данных можно установить на том же компьютере или на другом компьютере.

Не все серверные приложения выгодны для установки роли сервера приложений. Например роль сервера приложений не является необходимой для поддержки сервера Microsoft Exchange Server или Microsoft SQL Server на Windows Server 2008.

Чтобы определить, если роль сервера приложений является полезным для вашей организации бизнес-приложений, есть администраторы тесно связанные с разработчиками приложений для выявления требований приложения, например, она использует ли .NET Framework 3.0 или COM + компоненты.

Что такое сервер приложений?

Сервер приложений предоставляет следующее:

- Среда выполнения поддерживает эффективное развертывание и управление высокопроизводительных серверных бизнес-приложений. Эти приложения способны обслуживать запросы от удаленных клиентских систем, включая веб-браузеры, соединения из Интернета или из корпоративной сети или интрасети и систем удаленного компьютера, которые может отправлять запросы в виде сообщений.
- .NET Framework 3.0, который предоставляет разработчикам упрощенную модель программирования для подключенных серверных приложений. Разработчики могут использовать встроенный .NET Framework библиотеки для многих функций приложения, включая ввода/вывода (I/O), числовые и обработка текста, доступ к базе данных, XML обработку, управление транзакциями, рабочий процесс и веб-служб. Для системных администраторов .NET Framework обеспечивает безопасную и высокую производительность выполнения средой выполнения для серверных приложений, а также упрощенные приложения настройки и развертывания среды.
- Установка Windows Server 2008 в новый, удобный для пользователя мастера добавления ролей, помогает вам выбрать службы ролей и функции, которые необходимы для запуска приложений. Мастер добавления

ролей автоматически устанавливает все компоненты, необходимые для данной роли службы и делает его более легким для вас, для создания и предоставления компьютер в качестве сервера приложений для бизнес-приложений.

Кто будет заинтересован в этой роли?

Эта информация о роли сервера приложений является главным образом для информационных технологий (ИТ) специалистов, ответственных за развертывание и обслуживание Организации линии бизнес-приложений (LOB). Бизнес-приложения обычно разрабатываются в вашей организации или для вашей организации.

Среда сервера приложений состоит из одного или нескольких серверов под управлением Windows Server 2008, настроенных с роли сервера приложений. Это включает серверы, выполните следующие действия:

- Приложения, построенные с .NET Framework 3.0
- Приложения, построенные для использования COM +, очереди сообщений, веб-службы и распределенные транзакции
- Подключение к интрасети или Интернету для обмена информацией
- Приложения, которые предоставляют или использовать веб-службы
- Приложения, которые предоставляют веб-страниц
- Взаимодействовать с другими удаленными системами, работающие на разных платформах и операционных системах

Расширенный среды сервера приложений может также включать следующее:

- Домен клиентские компьютеры и их пользователей
- Компьютеры, которые используются главным образом для управления серверами приложений
- Серверы инфраструктуры, которые работают ресурсы, такие как доменных служб Active Directory (AD DS) или другие репозитории протокола LDAP (Lightweight Directory Access Protocol), службы сертификации, шлюзы безопасности, процесс серверы, интеграции серверов, приложений или шлюзы данных или базы данных

Какие новые возможности предоставляет эта роль?

Новый, расширена версию сервера приложений, роль устанавливается с помощью мастера добавления ролей в диспетчере сервера. Администраторы, которые имеют бизнес-приложений, построенных с .NET Framework 3.0 может обнаружить, что настройка среды размещения для этих приложений проще, с этой ролью сервера. Мастер добавления ролей руководство администратора через процесс выбора служб роли или функциональных возможностей, доступных в этой роли и могут быть необходимы для выполнения конкретных бизнес-приложений.

Ядро сервера приложений

Ядро сервера приложений – это группа технологий, устанавливаемых по умолчанию при установке роли сервера приложений.

По существу, это ядро сервера приложений.NET Framework 3.0. (Если у вас есть приложения, построенные с.NET Framework 3.5, вы можете загрузить и установить.NET Framework 3.5 на операционной системы.)

Windows Server 2008 включает.NET Framework 2.0, независимо от любой роли сервера, которая устанавливается. .NET Framework 2.0 содержит Common Language Runtime (CLR), которая обеспечивает среды выполнения кода, которая обеспечивает безопасное выполнение кода, его упрощенное развертывание, и поддержку совместного использования нескольких языков, а также обширные библиотеки для создания приложений.

Добавляетядро сервера приложений.NET Framework 3.0 возможности базовой линии.NET Framework 2.0 возможности. Для получения дополнительных сведений.NET Framework 3.0, см.Центр разработчиков NET Framework (<http://go.microsoft.com/fwlink/?LinkId=81263>).

Почему важна эта функциональная возможность?

Ключевые компоненты ядра сервера приложений установлены как набор библиотек кода и.ЧИСТЫЕ сборки. Ниже приведены ключевые компоненты ядра сервера приложений:

- Windows Communication Foundation (WCF)
- Windows Workflow Foundation (WF)
- Windows Presentation Foundation (WPF)

Из этих трех, WCF и WF часто используются в серверных приложениях, а также-приложениях клиента. WPF используется преимущественно в клиентских приложениях, и это не обсуждается далее здесь. Дополнительные сведения о WPF, см. в Windows Presentation Foundation (<http://go.microsoft.com/fwlink/?LinkId=78407>).

WCF является Microsoft унифицированную модель программирования для создания связанных приложений, использующих веб-службы для взаимодействия друг с другом. Эти приложения являются также известен как сервис ориентированных приложений (SOA), и они становятся все более важным для бизнеса. Разработчики могут использовать WCF для создания SOA-приложений, которые используют безопасные, надежные, транзакционные веб-служб, общаться на платформах и взаимодействовать с существующими системами и приложениями в вашей организации.

WCF позволяет разработчикам создавать или комбинировать различные технологии, которые сегодня доступны для создания распределенных приложений (COM + и.Услуги NET Enterprise, очереди сообщений.NET

Remoting, ASP. Чистый веб-служб и расширения веб-служб (WSE)) таким образом, чтобы иметь смысл для бизнеса и вычислительной среды вашей организации. Для получения дополнительных сведений о WCF видеть Windows Communication Foundation? (<http://go.microsoft.com/fwlink/?LinkId=81260>).

WF – это модель и ядро программирования для создания приложений, поддерживающих бизнес-процессы, быстро на Windows Server 2008. Рабочий процесс — это набор мероприятий, которые описывают процесс реального мира, такие, как процесс заказа покупки. Рабочий процесс обычно описывается и рассматривать графически — то, как блок-схемы. Описание рабочего процесса часто называют «модели». Рабочие элементы проходят через модель рабочего процесса от начала до конца.

Рабочие элементы или видов деятельности в рамках модели может выполняться людьми или системами или компьютеров. Хотя это можно описать рабочий процесс в традиционных языков программирования как ряд шагов и условий, для более сложных рабочих процессов или рабочие процессы, которые поддерживают более простые изменения, проектирование процесса графически и хранения что дизайн как модель обычно является гораздо более надлежащие и гибкие.

WF поддерживает рабочий процесс системы и документооборота в различных сценариях, включая следующие:

- Рабочий процесс в бизнес-приложений
- Последовательный поток экранов, страниц и диалоговые окна, представленные в ответ на взаимодействие пользователя с пользовательским интерфейсом (UI) для данного пользователя
- Документ центре рабочий процесс, например, обработка заказа на покупку или медицинские записи
- Взаимодействие документооборота, таких, как отправка электронной почты для бизнес-клиентов и получение электронной почты от клиента
- Составной рабочий процесс для SOA
- Бизнес правил-управляемые делом рабочий процесс, например: «В понедельник в 17: 00, отправить запрос на обновление каталога для деловых партнеров».
- Рабочий процесс для управления системами

Для дополнительной информации о WF, смотрите Windows Workflow Foundation (<http://go.microsoft.com/fwlink/?LinkId=82119>).

Что работает по-другому?

Хотя есть роль сервера приложений в Windows Server 2003, Новая, расширенная роль сервера приложений, доступных в ОС

Windows Server 2008 не просто обновление от средства настройки сервера

приложений, который включен в Windows Server 2003 или более ранних версий операционной системы. Так как роль функциональность является совершенно новой, администраторы должны осознавать, что не существует миграции пути для средства настройки сервера приложений с Windows Server 2003 или более ранних операционных систем.

Как решить эти проблемы?

Если обновления сервера до Windows Server 2008 с Windows Server 2003 или более ранних версий операционной системы, и вы хотите использовать возможности роли сервера приложений, необходимо переустановить роль сервера приложений с помощью мастера добавления ролей в диспетчере сервера. Как настроить Windows Server 2008 с правильным применением служб с помощью мастера добавления ролей в диспетчере сервера, можно легко переместить ваши приложения с Windows Server 2003 до Windows Server 2008.

Когда следует использовать роль сервера приложений?

Если на сервере бизнес-приложений, которые необходимо развертывать и управлять требуется одно или несколько из следующих технологий: Microsoft.NET Framework 3.0, очередь сообщений, COM + или распределенные транзакции, следует настроить ваш сервер в роли сервера приложений.

Как подготовиться к установке?

Как часть вашей подготовки для установки роли сервера приложений создаете перечень приложений, которые будут выполняться на этом сервере. Если вы являетесь администратором, работа с разработчиками или ISV, который разработал приложения для определения поддержки технологий и конфигурации, которые должны присутствовать на сервере для запуска приложений. Затем сопоставьте эти технологии служб ролей, которые описаны в следующих разделах, так что вы можете выбрать и должным образом настроить службы во время установки роли сервера. Обычно разработчик или ISV список из технологий, которые должны быть установлены для этого приложения, например, .NET Framework 3.0.

Веб-сервер

Этот параметр устанавливает службы IIS версии 7.0, веб-сервер, встроенный в Windows Server 2008. IIS была доступна в Windows Server в течение многих лет, но значительно был пересмотрен для Windows

Server 2008 для обеспечения улучшения в производительности, безопасности, управления, поддержки, надежности и модульности.

Службы IIS предоставляют следующие основные преимущества:

- IIS делает возможным для сервера приложений провести внутренние или внешние веб-сайты или услуги со статическим или динамическим содержимым.
- Службы IIS обеспечивают поддержку для запуска ASP.NET приложения, которые доступны в веб-обозревателе.
- Службы IIS обеспечивают поддержку для запуска веб-служб, которые построены с Microsoft WCF или ASP.NET.

Доступ К сети COM +

Этот параметр добавляет COM + доступ К сети для удаленного вызова приложений, которые построены и размещенных в COM + и корпоративных служб компонентов. Такие приложения называют также компонентами корпоративных служб.

Доступ К сети COM + является одной из возможностей удаленного вызова, поддерживается в Windows Server начиная с Windows 2000 Server, и он по-прежнему поддерживаться в Windows Server 2008. Более новые приложения обычно использовать WCF для поддержки удаленного вызова, потому что WCF обеспечивает взаимодействие на нескольких платформах.

Служба активации процессов Windows

Этот параметр добавляет службы активации процессов Windows (WAS). Можно запустить и останавливать приложения, динамически, основываясь на сообщениях, получаемых по сети через HTTP, очереди сообщений, TCP и именованные каналы протоколов. Динамические запуск и остановка приложений означает, что более эффективное использование ресурсов сервера. БЫЛО это новая услуга в Windows Server 2008.

Общий доступ К портам Net.TCP

Этот параметр добавляет портам Net.TCP. Эта служба роли позволяет нескольким приложениям использовать один TCP-порт для входящих сообщений. Например SOA, который построен с WCF могут совместно использовать тот же порт. Совместное использование портов требуется часто когда настройки брандмауэра или сетевые ограничения позволяют лишь ограниченное число открытых портов или когда несколько отдельных экземпляров приложения WCF

должна
быть запущена и в то же время.

Таким образом, чтобы несколько WCF приложения могут совместно использовать порты (мультиплексирование), портам Net.TCP выполняет мультиплексирование. Портам Net.TCP принимает входящие запросы на подключение, используя протокол TCP. Затем автоматически перенаправляет входящие запросы различным службам WCF, основанный на целевых адресов запросов.

Общий доступ к портам работает только тогда, когда приложения WCF используют протокол net.tcp для входящих соединений. Общий доступ к портам Net.TCP – это новая услуга в Windows Server 2008.

Распределенные транзакции

Приложения подключиться и выполнять обновления нескольких баз данных или других транзакций ресурсов могут потребовать, чтобы эти обновления выполняются с «логики» семантики транзакций —

это технология, которая обеспечивает, что каждая часть транзакции является полным или откат всей транзакции ее в первоначальное состояние.

Поддержка распределенных транзакций в Windows Server 2008 предоставляет способ для приложений для этого требования выполнены. Поддержка распределенных транзакций в Windows Server с Microsoft Windows NT ® Server 4.0, и эта поддержка продолжается в Windows Server 2008.

Доступна ли эта роль во всех выпусках Windows Server 2008?

Сервер приложений доступен в следующих выпусках Windows Server 2008:

- Windows Server 2008 стандарт
- Windows Server 2008 Enterprise
- Windows Server 2008 Datacenter
- Windows Server 2008 для систем на базе Itanium

Роль сервера приложений не доступны в следующем выпуске Windows Server 2008:

- Windows Web Server 2008

Он ведет себя по-разному в некоторых изданиях?

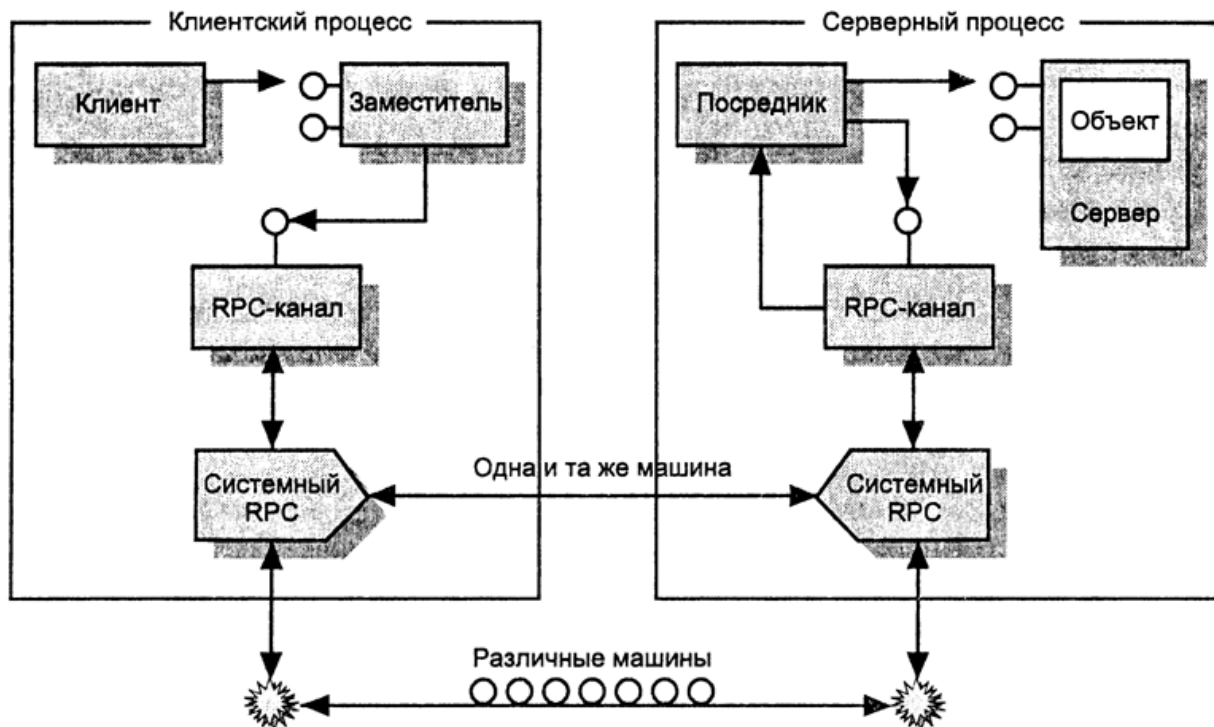
Сервера приложений, поведение не меняется основаны на выпуске Windows Server 2008.

Доступен в 32-разрядных и 64-разрядных версиях?

Сервер приложений доступен в 32-разрядных и 64-разрядных версиях Windows Server 2008.

Интерфейс сервера приложений

Состав и назначение основных частей программного обеспечения



Как видно из рисунка, взаимодействие клиента и сервера может происходить как локально, то есть в пределах одной машины, так и в распределенной среде. Когда клиент вызывает метод сервера, этот вызов перехватывается частью кода, которая называется функцией - заместителем (проху). “Заместитель” является представителем настоящего сервера и располагается в адресном пространстве клиента. “Заместитель” получает вызов метода, упаковывает параметры, которые будут посланы локальному серверу и вызывает соответствующий метод при помощи RPC. Клиент же не знает о существовании “Заместителя” и не интересуется его “деятельностью”.

Со стороны сервера функция-посредник (stub) получает от “Заместителя” вызов метода, распаковывает параметры и вызывает соответствующий метод реального сервера. Сервер, выполнив клиентский запрос, может вернуть некоторый результат. “Посредник” перехватывает результат, упаковывает любые возвращаемые значения и направляет эту информацию соответствующему “Заместителю”. “Заместитель” получает возвращаемые значения, распаковывает их и передает клиенту.

Рассмотрим подробнее работу “Заместителя” и “Посредника”.

“Заместитель” - это задача, которая обеспечивает разбор клиентских запросов и запуск требуемых процедур на сервере приложений. Представляет из себя приложение, обеспечивающее интерфейс для авторизации на сервере, реализацию процедурных вызовов, объектных вызовов и информационных запросов для наблюдения за ходом исполнения запросов. Обеспечивает вызов call-back’ов, установленных в клиентской

части. Запросы клиентского приложения передает по транспортному протоколу на сервер приложений, а ответы принимает и передает клиентскому ПО.

“Посредник” в принципе аналогичен “Заместителю”, за исключением того, что запросы не передаются в сеть, а обращение к серверному объекту производится напрямую. Он представляет из себя объектную библиотеку для сборки с прикладным программным обеспечением сервера приложений. Для сессии каждого клиента создается отдельный экземпляр “Посредника”. На сервере приложений также необходима некоторая задача, которая будет обрабатывать приходящие запросы, разбирать их и осуществлять вызов требуемых процедур на сервере через “Посредника”. Необходимо также обеспечить поддержку очереди запросов для каждого клиента и отвержение запросов, если до этого был выполнен блокирующий вызов.

Программный интерфейс “Посредника” и “Заместителя” для работы с прикладными объектами сервера приложений

Название метода	Назначение	Передаваемые параметры
Login	Авторизация пользователя на сервере приложений	<ol style="list-style-type: none"> Имя пользователя Пароль Роль
CallASproc	Вызов процедуры или функции сервера приложений	<ol style="list-style-type: none"> Имя функции Список параметров вызова Тип вызова Возникшее исключение
CallASObjMethod	Вызов метода <i>прикладного</i> объекта сервера приложений	<ol style="list-style-type: none"> Указатель на объект или на один из его интерфейсов Имя метода Список параметров вызова Тип вызова Возникшее исключение
GetASObjProp	Получение свойства <i>прикладного</i> объекта сервера приложений	<ol style="list-style-type: none"> Указатель на объект или на один из его интерфейсов Имя свойства
SetASObjProp	Установка значения свойства <i>прикладного</i> объекта сервера приложений	<ol style="list-style-type: none"> Указатель на объект или на один из его интерфейсов Новое значение свойства
RegisterVar	Создает переменную и устанавливает ее значение	<ol style="list-style-type: none"> Имя переменной Значение переменной
UnregisterVar	Уничтожает переменную	<ol style="list-style-type: none"> Имя переменной
GetVar	Возвращает значение переменной	<ol style="list-style-type: none"> Имя переменной

SetVAr	Устанавливает значение переменной	<ol style="list-style-type: none"> 1. Имя переменной 2. Новое значение переменной
TieASObject	Связывает зарегистрированный объект с символьным именем	<ol style="list-style-type: none"> 1. Символьное имя 2. Интерфейс объекта
UntieASObject	Освобождает символьное имя, не удаляя объекта	<ol style="list-style-type: none"> 1. Символьное имя

Login – регистрация пользователя на сервере приложений. При выполнении этой функции производятся все действия по регистрации нового пользователя и созданию его

“жизненного пространства”.

CallASProc – вызывает метод сервера приложений с параметрами. Тип вызова указывает, блокирующий вызов или нет. Возвращает список параметров с результатами выполнения. Для реализации метода “Посреднику” необходимо обратиться к библиотеке функций сервера приложений. “Заместитель” должен упаковать имя функции и параметры и отправить запрос в сеть. Функция должна произвести разбор входных параметров – выделить из них объектные ссылки и проверить объекты, определенные ими на зарегистрированность у “Посредника”. Проверка идентичности объекта производится сравнением с именем класса и его идентификатором, который должен быть предварительно прочитан при регистрации объекта. Также необходимо выделить параметры типа “call-back” и при наличии зарегистрировать их.

CallASObjMethod – вызывает метод прикладного объекта сервера приложений. “Посредник” должен проверить объект на зарегистрированность. Процедура вызова метода объекта аналогична вышеописанной, за исключением того, что вместо обращения к библиотеке функций сервера приложений происходит вызов метода объекта сервера приложений.

GetASObjProp – возвращает значение свойства объекта сервера приложений.

SetASObjProp – устанавливает значение свойства у указанного объекта.

Функции **TieASObject** и **UntieASObject** весьма полезны для реализации доступа к прикладным объектам сервера приложений посредством символьного имени, которое может сгенерировать клиент.

Также для успешного взаимодействия сервера приложений и клиента обоим компонентам, то есть “Посреднику” и “Заместителю”, необходимо иметь следующие функции и свойства :

KillProcess – аварийно завершает процесс, ранее запущенный

GetProcessCondition – запрашивает текущее состояние процесса. Возможные состояния – выполняется, в очереди, ждет ответа на call-back.

Synhronize – сверяет идентификаторы зарегистрированных процессов, объектов и call-back'ов на клиентской и серверной части. Для Посредника процессы и объекты, которых нет на клиенте молча разрушаются, процессы и объекты, которых нет на сервере, но есть на клиенте, разрушаются с генерацией исключения для прикладной программы. Для Заместителя процессы и объекты, которых нет на сервере создаются заново, а процессы и объекты, которых нет на клиенте, но есть на сервере молча разрушаются.

Logout – связь с сервером прерывается. Все начаты процессы аварийно завершаются, все созданные объекты разрушаются. По разрушению здесь подразумевается не разрушение прикладных объектов сервера приложений, а уничтожение указателей на их интерфейсы, которых может быть несколько.

Методы только Посредника:

InitCallBack – иницирует вызов call-back'а на стороне клиента с параметрами. Call-back может вызываться тремя способами : блокирующим, неблокирующим и без

подтверждения. Это указывается в типе вызова call-back'а.

LockListener – запрещает обрабатывать входящие запросы. На все входящие запросы интерфейс отвечает, что он заблокирован. Количество вызовов должно подсчитываться в счетчике вызовов блокировок.

UnLockListener – вычитает единицу из счетчика вызовов блокировки. Вновь разрешает обрабатывать запросы, если счетчик вызовов обнулится.

CheckLock – проверка заблокированности интерфейса.

DownServer – завершение работы сервера. Все начаты процессы завершаются, все созданные объекты разрушаются.

Одной из интересных особенностей такого протокола является механизм реализации так называемых callback–параметров. Это параметры, которые передаются процедуре или функции при ее вызове, однако они не несут в себе данных как таковых. Вместо этого в каждом параметре содержится информация о том, каким образом эти данные можно получить. Например, это может быть указатель на функцию, которая вернет необходимый результат, или, если речь идет об объекте, указатель на интерфейс объекта и на его метод.

Такой механизм удобен и позволяет экономить ресурсы сервера при обработке запросов.

Веб-сервер

Веб-сервер (web-server) - это сервер, отвечающий за прием и обработку запросов (HTTP-запросов) от клиентов к веб-сайту. В качестве клиентов обычно выступают различные веб-браузеры. В ответ веб-сервер выдает клиентам HTTP-ответы, в большинстве случаев, вместе с HTML-страницей, которая может содержать: всевозможные файлы, какие-то изображения, медиа-поток или любые другие данные.

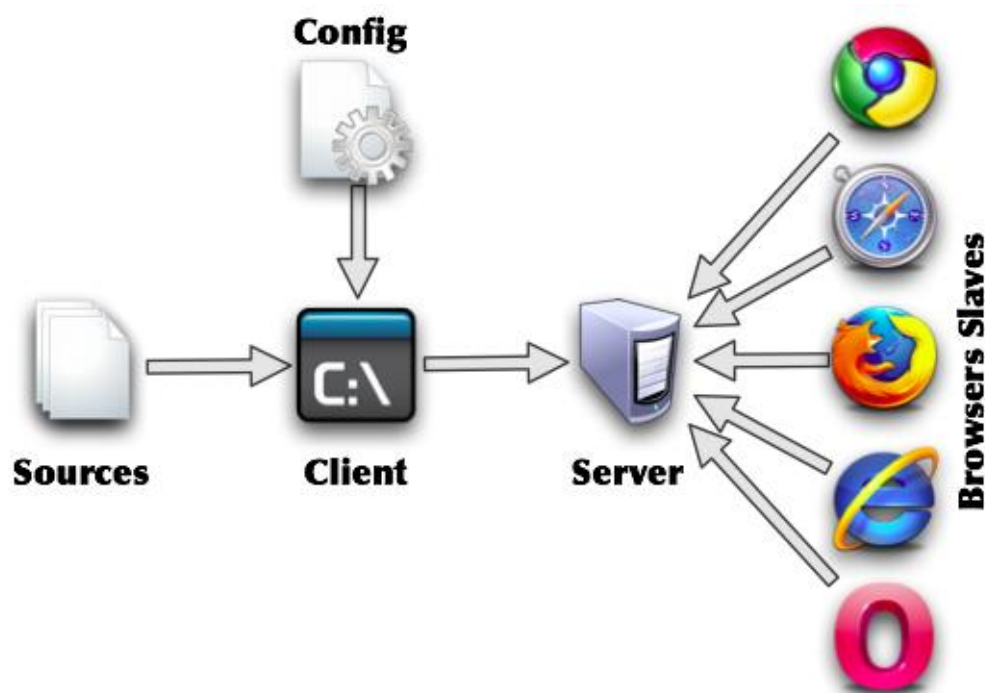
Также веб-сервер выполняет исполнение скриптов, например, таких как CGI, JSP, ASP и PHP, которые отвечают за организацию запросов к сетевым службам, базам данных, доступу к файлам, пересылке электронной почты и другим приложениям электронной коммерции.

Термин Веб-сервер также применяется к техническим устройствам и программному обеспечению, которые выполняет функции веб-сервера. Это может быть какой-нибудь компьютер, который специально выделен из группы персональных компьютеров или рабочая станция, на которых установлено и работает сервисное программное обеспечение.

Клиент пользователя, которым преимущественно является веб-браузер, передает веб-серверу запросы на получение ресурсов, обозначенных URL-адресами. Ресурсы - это HTML-страницы, цифровой медиа контент, медиа-потoki, различные изображения, файлы данных, или любые другие данные, необходимые клиенту. В ответ веб-сервер передает клиенту запрошенные им данные. Этот обмен происходит с помощью протокола HTTP.

HTTP (англ. HyperText Transfer Protocol - протокол передачи гипертекста) – это сетевой протокол прикладного уровня передачи данных. Основным принципом протокола HTTP является технология «клиент-сервер», обеспечивающая взаимодействие сети и пользователя.

В случае малой организации веб-сервер может быть целостной системой, которая будет состоять из: HTTP-сервера – служит для запросов к веб-страницам, FTP-сервера – применяется для загрузки файлов через Интернет, NNTP-сервера – выполняет доступ к группам новостей, а также SMTP-сервера - для электронной почты.



История

Изобретателем первого веб-сервера считается британский ученый Тим Бернерс-Ли. Работая с 1980 года в Европейской лаборатории по ядерным исследованиям (фр. Conseil Européen pour la Recherche Nucléaire, CERN) консультантом по программному обеспечению, он и приступил к своим, ставшим впоследствии, знаменитым разработкам. Собственно там, в Женеве в Швейцарии, он для своих собственных потребностей разработал программу «Энквайр» (англ. enquire - спрашивать), которая использовала случайные ассоциации для хранения данных и заложила концепцию для основы Всемирной паутины.

В 1989 году Тим Бернерс-Ли, работал над внутренней сетью организации CERN и предложил основать глобальный гипертекстовый проект, который заключался в публикации гипертекстовых документов, связанных между собой гиперссылками. Внедрение этого проекта, по его мнению, облегчило бы объединение, поиск и обмен информацией для ученых CERN. Для осуществления проекта Тим Бернерс-Ли вместе со своими помощниками изобрел идентификаторы URI и URL, протокол HTTP, а также язык HTML. Все эти технологии теперь широко применяются в современном Интернете и без них уже не обойтись.

В результате выполнения этого проекта Бернерс-Ли разработал первый в мире веб-сервер, называвшийся «httpd», а также первый в мире гипертекстовый веб-браузер для компьютера NeXT, получивший название WorldWideWeb (Всемирная паутина).

Первый веб-браузер работал на платформе NeXTSTEP - объектно-ориентированной,

многозадачной операционной системе, и был разработан с помощью Interface Builder. Интерфейс веб-браузера был очень простым, и почти вся информация отображалась в текстовом формате только лишь с несколькими изображениями. Помимо стандартного протокола FTP, Тим Бернерс-Ли использовал новый, только изобретенный им, протокол HTTP. В период с 1991 по 1993 год Бернерс-Ли усовершенствовал технические свойства своих новых разработок: идентификаторов URI и URL, протокола HTTP и языка HTML и опубликовал их. Позже веб-браузер был переименован в "Nexus", чтобы не возникла путаница между названием операционной системы, на которой был разработан браузер и его названием.

Первый в мире веб-сервер и первый веб-браузер работали на персональном компьютере NeXTSTEP и сейчас этот компьютер выставлен в музее CERN (Микрокосм) на всеобщее обозрение.

Первый в мире веб-сайт Тим Бернерс-Ли разместил по адресу "<http://info.cern.ch>", сейчас этот сайт хранится в архиве. Первый сайт появился в Интернете 6 августа 1991 года. На этом веб-сайте было описание, что же такое Всемирная паутина, как правильно установить веб-сервер, как приобрести веб-браузер и прочая техническая информация. Этот сайт также представлял собой первый в мире интернет-каталог. Бернерс-Ли разместил на сайте список ссылок на другие сайты и регулярно обновлял его.

12 декабря 1991 года в Стэнфордском центре линейного ускорителя (SLAC) в США был установлен первый в мире веб-сервер.

Основные и дополнительные функции веб-сервера:

- Прием запросов от веб-браузеров по протоколу стандарта HTTP с использованием сетевых протоколов TCP/IP;
- Выполнение поиска и отсылки файлов с гипертекстом или каких-либо документов в браузер по протоколу HTTP;
- Обслуживание и обработка запросов, типа: mailto:..., FTP, Telnet и т.п.;
- Запуск прикладных программ на веб-сервере с последующей передачей и возвратом параметров обработки через стандарт интерфейса CGI;
- Работа и обслуживание навигационных карт изображений (Image map);
- Загрузка Java-приложений;
- Администрация и оперативное управление сервером;
- Авторизация пользователей и их аутентификация;
- Ведение регистрационного журнала обращений пользователей к различным ресурсам;
- Автоматизированная работа веб-страниц;
- Поддержка страниц, которые генерируются динамически;
- Поддержка работы протокола HTTPS для защищенных соединений с клиентами.

Описание работы веб-сервера

Веб-браузеры поддерживают связь с веб-серверами с помощью протокола передачи гипертекстовых сообщений (HypertextTransferProtocol, HTTP). Это простой протокол запросов и ответов для пересылки информации с использованием протокола TCP/IP. Веб-сервер получает запрос, обнаруживает файл, посылает его браузеру, а затем разрывает соединение. Графическая информация, которая имеется на странице, обрабатывается таким же образом. Далее настает очередь веб-браузера вывести на монитор пользователя, загруженный из сети HTML-документ.

Кроме HTML-страниц и графики, веб-серверы могут хранить любые файлы, в том числе текстовые, документы текстовых процессоров, видеофайлы и аудиоинформацию. На сегодняшний день, если не учитывать анкет, которые заполняют пользователи, основная часть веб-трафика передается в одном направлении - браузеры считывают файлы с веб-сервера. Но это положение изменится после общего принятия описанного в проекте HTTP 1.1 метода PUT, который позволяет записывать файлы на веб-сервер. Сегодня метод PUT используется в основном пользователями, создающими веб-страницы, но в перспективе он может пригодиться и остальным пользователям для обратной связи с информационными центрами. Запросы методом PUT намного проще, чем обыкновенная POST загрузка файлов на веб-сервер.

На веб-сервере также выполняют свою работу различные приложения, наибольшую популярность среди которых, получили поисковики и средства связи с базами данных. Для разработки этих приложений применяются такие стандарты, как общий шлюзовый интерфейс (CommonGatewayInterface, CGI), языки сценариев JavaScript, а также языки программирования Java и VisualBasic. Кроме интерфейса стандарта CGI, некоторые фирмы-разработчики веб-серверов создали интерфейсы прикладного программирования (API) такие как, например, Netscape Server API и Internet Server API, которые созданы компаниями Microsoft и Process Software AG. Эти интерфейсы позволяют разработчикам непосредственно обращаться к конкретным функциям веб-сервера. Некоторые веб-серверы обладают связующим программным обеспечением (middleware) для подключения к базам данных, работа с которыми может потребовать профессиональных знаний в программировании.

Базовые функции поиска помогают пользователям отсортировывать нужную им информацию, а утилиты для связи с базами данных предоставляют пользователям веб-браузеров доступ к этой информации.

Обзор веб-серверов

Критериями для выбора веб-сервера могут быть разные характеристики: установка, настройка конфигурации, управление сервером, администрирование, управление размещаемой на сервере информации, защита этой информации, контроль доступа, функции разработки приложений, а также производительность.

Большинство веб-серверов устанавливается легко и быстро.

Самая сложная часть процесса инсталляции - это проведение конфигурации нескольких имен доменов на одном физическом устройстве или другими словами организация виртуальных серверов.

Веб-серверы имеют средства для управления информационным модулем, характеризующие общую организацию веб-узла, а также обладают инструментами для проверки правильности внутренних и внешних гипертекстовых связей. Пакет LiveWire фирмы Netscape Communications, который поставляется вместе с Novell Open Enterprise Server (OES) и дополнительно предлагаемый с сервером FastTrack, обладает утилитой управления узлом, которая формирует список всех связей выбранной страницы. Эта утилита также предоставляет общий перечень всех некорректных связей, которые обнаруживает. Программа WebView компании «O'Reilly & Associates» обладает такой же функцией и может выводит на экран подробное дерево файлов, в котором все некорректные связи выделяются красным цветом.

Также имеются и элементарные средства для управления содержательным материалом. Веб-администраторы должны выбирать, где хранить файлы и как именно будет осуществляться доступ к этим файлам со стороны пользователей, которые будут обращаться на веб-сервер. Для этого требуется устанавливать соответствие между логическими URL и физическими каталогами файлов. Каждое программное обеспечение выполняет эту операцию своим уникальным способом.

С увеличением популярности веб-серверов и все более широкого их применения в интрасетях, усиливается коммерческая активность в Интернете, поэтому возрастает важность защиты информации. Чаще всего системы обеспечения безопасности веб-сервера оказываются или избыточными, или недостаточными для современных интрасетей. Если вам необходимо ограничить доступ к определенной информации внутри компании, то у вас есть выбор между использованием незашифрованных паролей, которые передаются по каналам связи, и применением протокола SSL (англ. Secure Sockets Layer - уровень защищенных сокетов) - сложного и медленного метода, который используется для шифровки паролей и данных.

Для того чтобы организовать работу отдельных пользователей и их групп могут быть использованы внутренние приложения сервера или определенные функции операционной системы. Для того чтобы организовать работу отдельных пользователей и их групп могут быть использованы внутренние приложения сервера или определенные функции операционной системы. В пакетной службе Microsoft IIS предусмотрено применение средств базовой сетевой ОС Windows NT.

Пакет NetWare Web Server фирмы Novell, Inc. целиком интегрирован со службами адресных каталогов (NetWare Directory Services, NDS). Конечно же, наладить работу пользователей из общего центра очень удобно, но это может принести возможную угрозу для безопасности. Пароли распространяются по каналам связи в незашифрованном виде, и если их перехватят, то подвергнется риску не только веб-сервер, но и безопасность всей сетевой операционной системы.

Разработка приложений - это одна из наиболее основных функций веб-сервера. Среда разработки приложений и инструменты подключения к базам данных очень важны для того, чтобы расширить возможности веб-сервера. Это важно, потому что разработка приложений зависит от различных своеобразных деталей интерфейса прикладного программирования (англ. application programming interface, API), а также от особенностей языков программирования или индивидуальных предпочтений программистов.

Веб-серверы могут обслуживать различные системы от малой интрасети предприятия до крупных информационных веб-центров, которыми пользуются миллионы людей.

Для малых корпоративных интрасетей, лучше всего подойдет пакет Internet Information Server (IIS), созданный и распространяемый компанией Microsoft. IIS отличается достаточно простой инсталляцией и простыми настройками конфигурации. Этот пакет веб-сервера отлично интегрирован со средствами управления доступом, инструментом контроля параметров системы Performance Monitor (Системный монитор), а также с программой просмотра журнала событий Event Viewer. Еще веб-сервером IIS представляется несколько инструментов для динамической передачи информации из баз данных. IIS отличается очень высоким быстродействием. Компоненты IIS поддерживают такие протоколы, как: HTTP, HTTPS, FTP, NNTP, SMTP, POP3.

С целью облегчить создание информационных веб-центров, с большинством веб-серверов уже поставляются утилиты и инструменты для управления содержательным материалом. Кроме HTML-редакторов и конвертеров форматов документов, самыми полезными являются средства контроля URL, которые гарантируют работоспособность всех гипертекстовых связей вашего веб-узла.

В общем-то, любой персональный компьютер, который подключен к сети Интернет, можно сделать веб-сервером, если установить на него специальное серверное программное обеспечение.

Самые распространенные веб-серверы: Apache (компания Apache Software Foundation), IIS (компания Microsoft) и iPlanet server (от компаний Sun Microsystems и Netscape Communications Corporation). Сейчас на рынке программного обеспечения для веб-серверов, существует огромный выбор продуктов, как коммерческих, так и бесплатных.

Одним из самых распространенных веб-серверов, является Apache от компании Apache Software Foundation. По ориентировочным подсчетам, он используется на 65% всех веб-серверов в мире. Одно из основных достоинств программного обеспечения Apache - это то, что он распространяется бесплатно. Разработчики регулярно устраняют найденные ошибки и предоставляют хорошую поддержку пользователей. Данный веб-сервер поддерживает большое количество модулей, утилит и дополнений. С самого начала Apache разрабатывался как программное обеспечение для администраторов и опытных пользователей, поэтому у него есть недостаток - это сложность настройки и обслуживания.

Далее по популярности идет веб-сервер IIS от компании Microsoft. По данным компании Netcraft веб-сервер IIS составляет 12,46% от общего числа веб-серверов. Этот продукт входит в состав серверного программного обеспечения семейства Windows NT. Его основные преимущества - это стабильность, высокая скорость работы, а также возможность подключать дополнительные модули. Компания Microsoft стремится к тому, чтобы любой пользователь смог пользоваться ее продуктами без помощи специалистов, если ему нужно решить стандартные задачи. Поэтому система IIS очень проста в установке, настройке и обслуживании. Веб-сервер поддерживает технологию .NET, набирающую, в последнее время, популярность в среде разработчиков и профессиональных пользователей. Эти достоинства выводят веб-сервер IIS на новый уровень и можно ожидать, что его использование возрастет.

Другие известные веб-серверы:

nginx — свободный веб-сервер и почтовый прокси-сервер, разрабатываемый Игорем Сисоевым. Простой, быстрый и надежный сервер. Работает в Linux и других Unix-подобных операционных системах, а также в Windows. Пользуется популярностью на крупных веб-сайтах.

lighttpd — свободный веб-сервер. Разработчик Ян Кнешеке. Быстрый и безопасный веб-сервер. Работает в Linux и других Unix-подобных операционных системах, а также в Windows;

Google Web Server — веб-сервер, который основан на Apache и используется компанией Google для организации своей веб-инфраструктуры;

Resin — свободный веб-сервер и сервер приложений для Java. Разработчик компания Caucho Technology, Inc.

Cherokee — свободный веб-сервер, который управляется только через веб-интерфейс. Написан на языке программирования Си.

Rootage — веб-сервер, который написан на языке программирования Java. Работает в Linux и Windows;

THTTPD — простой, маленький, быстрый и безопасный веб-сервер. Разработчик компания ACME Labs Software.

Клиенты веб-сервера

Обычно, клиентом является веб-браузер. Но также обращаться к веб-серверу могут и другие разнообразные устройства и программы:

- Веб-браузер, который установлен на стационарном персональном компьютере;
- Веб-браузер, который установлен на КПК или другом переносном устройстве;
- Мобильные телефоны и смартфоны, с помощью которых пользователь получает доступ к ресурсам веб-сервера по WAP-протоколу;
- Различные программы, которые могут обращаться к веб-серверу самостоятельно для обновления либо получения другой информации. Пример - различные антивирусы,

которые периодически обращаются к веб-серверу, чтобы обновить базу данных;
- Разные цифровые устройства, а также некоторая бытовая техника.

Характеристика сервера Apache

Без преувеличения можно сказать, что это самый распространенный Web- сервер в мире. Сейчас имеются версии фактически для всех известных платформ и операционных систем, в том числе и для Windows NT. Поддержка Windows NT появилась лишь в последней версии, и это еще больше повысило популярность Apache в мире. Его несомненными достоинствами являются надежность, исключительная производительность и огромный набор функций и дополнительных модулей. Но все же «изюминкой» этого сервера является свободное бесплатное распространение. Это дает возможность, помимо экономии денег, быстро исправлять ошибки и вносить в код программы необходимые дополнения. Надо отдать должное тем, кто занимается созданием Web-сервера Apache, — эти люди безвозмездно и, главное, очень быстро справляются с замеченными ошибками. Кстати, отмечу, что таким человеком сможет стать любой, желающий внести свой вклад в развитие этого сервера. Версия 1.3.x считается одной из самых стабильных и быстрых среди всего семейства Apache. Основными свойствами Apache являются поддержка кросс-платформ, протокол HTTP/1.1, модульная структура, защита, запись log-файлов. Кроме стандартной поставки имеется большое количество модулей, которые расширяют возможности Apache. Свободное распространение и открытый код позволяют создавать наиболее защищенные модули. В этом с Apache трудно соревноваться любому коммерческому серверу. Но, несмотря на все эти достоинства, есть и серьезный пробел, который я бы все-таки не назвал недостатком: у Apache нет красивой программы установки и управления. Во время установки приходится иметь дело с командной строкой. Более того, при установке под UNIX иногда приходится даже вносить коррективы в include-файлы. Конечно, есть подробное описание того, что и как необходимо изменить, однако, на первый взгляд это кажется не очень удобным. При работе приходится иметь дело не с оконной программой, а с текстовыми конфигурационными файлами. Техническая поддержка проявляется не в виде красивых help-файлов или фирм, предоставляющих специалистов, а в основном через телеконференции и обмен мнениями системных администраторов. На самом деле фирмы, предоставляющие услуги технической поддержки, все же есть, но это стоит денег. Но такой нетипичный для серьезного продукта стиль не делает Apache менее популярным, и на рынке он является самым серьезным конкурентом для всех WWW-серверов.

Краткие характеристики Web сервера Apache:

- последняя версия: 2.0 alpha 4;
- последняя официальная версия: 1.3.12;
- бесплатный, открытый код;
- операционные системы: NetBSD, Digital UNIX, BSDI, AIX, OS/2, SCO, HP-UX, Novell NetWare, Macintosh, Be OS, Windows NT, Linux, Windows 95, FreeBSD, Windows 98, IRIX, Solaris;
- может создавать несколько log-файлов; под Windows NT запускается как сервис и/или программа, под UNIX может запускаться из inetd; допускает настройку на несколько портов;
- поддерживает Windows CGI, HTTP/1.1, в том числе и HTTP/1.1 PUT; есть функция автоматического ответа при изменении документа; поддерживает Microsoft ISAPI;
- возможен запрет доступа с конкретных адресов, к конкретным документам, запрет запуска CGI скриптов, доступ конкретных пользователей; допускаются изменения без перезапуска сервера;

- поставляется вместе с полным исходным кодом; поддерживает другие протоколы (ftp, telnet); имеет настройку на пользовательские директории; содержит модуль проху.

Данные Webcompare (<http://webcompare.internet.com>) и Swatch (<http://serverwatch.internet.com>) для сервера Apache:

- Процент на рынке Web-серверов — 58,85%
- Количество — 3 570 377

Рейтинг (по пятибалльной системе):

- надежность — 5
- производительность — 5
- простота использования — 3
- техническая поддержка — 4,5

Характеристика Internet Information Server от Microsoft

Сервер IIS является лучшим для Windows NT. Это и понятно, потому что вряд ли кто-нибудь, кроме разработчиков операционной системы, может сделать программу, полностью использующую возможности этой системы. Версия 4.0 поставлялась бесплатно вместе с Service Pack, а версия 5.0 поставляется исключительно как часть Windows 2000. Версия 5.0, по мнению специалистов, отличается более высокой надежностью, более тесной интеграцией с Windows и целым набором новых свойств. Установка и конфигурирование занимают не более 10 минут. Особенно приятно, что теперь не нужно устанавливать какие-либо Service Pack' и не нужно перегружать систему после установки. При всем этом сам сервер занимает около 30 Мбайт на диске и вполне работает на машине Pentium 200 МГц с объемом памяти 128 Мбайт. Однако установить и запустить сервер можно только на Windows 2000 Server. Помимо удобной установки, удобно реализована и система управления сервером. Существует несколько полезных программ шаблонов (Wizards) для выполнения частых и рутинных операций. Вообще, справедливости ради нужно отметить следующее: в том, что касается удобства, мало кто сможет соревноваться с программами Microsoft. К тому же IIS поддерживает известные и включает в себя новые стандарты защиты. Так, есть поддержка известных методов SSL 3.0, Kerberos 5.0, и нового метода Fortezza (новый стандарт безопасности). Наиболее интересной и отличительной чертой IIS является поддержка WebDAV (Web-based Distributed Authoring and Versioning). Это недавно появившийся стандарт, который позволяет превращать внутренние сети в единое пространство, используя ресурсы соседних компьютеров как свои. Для пользователей IIS это означает, что они могут с большим удобством разделять свои рабочие файлы и иметь при этом возможность блокировать файлы. Вместе с тем существуют и некоторые проблемы, и несовместимость. Не совсем корректно происходит работа одновременно с Front Page Server, иногда при изменении конфигурации приходится перезапускать систему, встречаются ошибки при использовании системы удаленного администрирования. Но в целом версия 5.0 является значительным шагом вперед по сравнению с 4.0 в плане надежности и удобства.

Приведем кратко основные характеристики:

- последняя версия: 5.0;
- цена: поставляется вместе с Windows 2000 Server;

- операционные системы: Windows 2000 Server, Advanced Server;
- поддержка ASP, Microsoft API, ODBS;
- может создавать несколько log-файлов; протоколирование производительности; возможность создания log-файлов для каждого CGI-скрипта; под Windows NT запускается как сервис и/или программа, под UNIX может запускаться из inetd; допускает настройку на несколько портов;
- в поставку включен SNMP-агент; поддерживает доступ к переменным состояния сервера из CGI, HTTP/1.1, в том числе и HTTP/1.1 PUT; есть функция автоматического ответа при изменении документа; поддерживает Microsoft ISAPI;
- возможен запрет доступа с конкретных адресов, к конкретным документам, доступ конкретных пользователей; есть поддержка S-HTTP; допускаются изменения без перезапуска сервера; есть поддержка SSL второй и третьей версий и авторизации;
- графическая программа установки; графическая программа управления; поддерживает другие TCP-протоколы (ftp, telnet); присутствует программа для оценки производительности во время работы; поддержка директорий пользователей; встроенный алгоритм поиска; возможность удаленного управления.

Данные Webcompare (<http://webcompare.internet.com>) и Swatch (<http://serverwatch.internet.com>) для сервера IIS:

- Процент на рынке Web-серверов — нет данных
- Количество — нет данных

Рейтинг (по пяти бальной системе):

- надежность — 5
- производительность — 5
- простота использования — 5
- техническая поддержка — 5.

Хранимые процедуры

Понятие хранимой процедуры

Хранимые процедуры представляют собой группы связанных между собой операторов SQL, применение которых делает работу программиста более легкой и гибкой, поскольку выполнить хранимую процедуру часто оказывается гораздо проще, чем последовательность отдельных операторов SQL. Хранимые процедуры представляют собой набор команд, состоящий из одного или нескольких операторов SQL или функций и сохраняемый в базе данных в откомпилированном виде. Выполнение в базе данных хранимых процедур вместо отдельных операторов SQL дает пользователю следующие преимущества:

- необходимые операторы уже содержатся в базе данных;
- все они прошли этап синтаксического анализа и находятся в исполняемом формате; перед выполнением хранимой процедуры SQL Server генерирует для нее план исполнения, выполняет ее оптимизацию и компиляцию;
- хранимые процедуры поддерживают модульное программирование, так как позволяют разбивать большие задачи на самостоятельные, более мелкие и удобные в управлении части;
- хранимые процедуры могут вызывать другие хранимые процедуры и функции;
- хранимые процедуры могут быть вызваны из прикладных программ других типов;

- как правило, хранимые процедуры выполняются быстрее, чем последовательность отдельных операторов;
- хранимые процедуры проще использовать: они могут состоять из десятков и сотен команд, но для их запуска достаточно указать всего лишь имя нужной хранимой процедуры. Это позволяет уменьшить размер запроса, посылаемого от клиента на сервер, а значит, и нагрузку на сеть.

Хранение процедур в том же месте, где они исполняются, обеспечивает уменьшение объема передаваемых по сети данных и повышает общую производительность системы. Применение хранимых процедур упрощает сопровождение программных комплексов и внесение изменений в них. Обычно все ограничения целостности в виде правил и алгоритмов обработки данных реализуются на сервере баз данных и доступны конечному приложению в виде набора хранимых процедур, которые и представляют интерфейс обработки данных. Для обеспечения целостности данных, а также в целях безопасности, приложение обычно не получает прямого доступа к данным – вся работа с ними ведется путем вызова тех или иных хранимых процедур.

Подобный подход делает весьма простой модификацию алгоритмов обработки данных, тотчас же становящихся доступными для всех пользователей сети, и обеспечивает возможность расширения системы без внесения изменений в само приложение: достаточно изменить хранимую процедуру на сервере баз данных. Разработчику не нужно перекомпилировать приложение, создавать его копии, а также инструктировать пользователей о необходимости работы с новой версией. Пользователи вообще могут не подозревать о том, что в систему внесены изменения.

Хранимые процедуры существуют независимо от таблиц или каких-либо других объектов баз данных. Они вызываются клиентской программой, другой хранимой процедурой или триггером. Разработчик может управлять правами доступа к хранимой процедуре, разрешая или запрещая ее выполнение. Изменять код хранимой процедуры разрешается только ее владельцу или члену фиксированной роли базы данных. При необходимости можно передать права владения ею от одного пользователя к другому.

Хранимые процедуры в среде MS SQL Server

При работе с SQL Server пользователи могут создавать собственные процедуры, реализующие те или иные действия. Хранимые процедуры являются полноценными объектами базы данных, а потому каждая из них хранится в конкретной базе данных. Непосредственный вызов хранимой процедуры возможен, только если он осуществляется в контексте той базы данных, где находится процедура.

Типы хранимых процедур

В SQL Server имеется несколько типов хранимых процедур.

- Системные хранимые процедуры предназначены для выполнения различных административных действий. Практически все действия по администрированию сервера выполняются с их помощью. Можно сказать, что системные хранимые процедуры являются интерфейсом, обеспечивающим работу с системными таблицами, которая, в конечном счете, сводится к изменению, добавлению, удалению и выборке данных из системных таблиц как пользовательских, так и системных баз данных. Системные хранимые процедуры имеют префикс `sp_`, хранятся в системной базе данных и могут быть вызваны в контексте любой другой базы данных.
- Пользовательские хранимые процедуры реализуют те или иные действия. Хранимые процедуры – полноценный объект базы данных. Вследствие этого каждая хранимая процедура располагается в конкретной базе данных, где и выполняется.
- Временные хранимые процедуры существуют лишь некоторое время, после чего автоматически уничтожаются сервером. Они делятся на локальные и глобальные. Локальные временные хранимые процедуры могут быть вызваны только из того

соединения, в котором созданы. При создании такой процедуры ей необходимо дать имя, начинающееся с одного символа #. Как и все временные объекты, хранимые процедуры этого типа автоматически удаляются при отключении пользователя, перезапуске или остановке сервера. Глобальные временные хранимые процедуры доступны для любых соединений сервера, на котором имеется такая же процедура. Для ее определения достаточно дать ей имя, начинающееся с символов ##. Удаляются эти процедуры при перезапуске или остановке сервера, а также при закрытии соединения, в контексте которого они были созданы.

Создание, изменение и удаление хранимых процедур

Создание хранимой процедуры предполагает решение следующих задач:

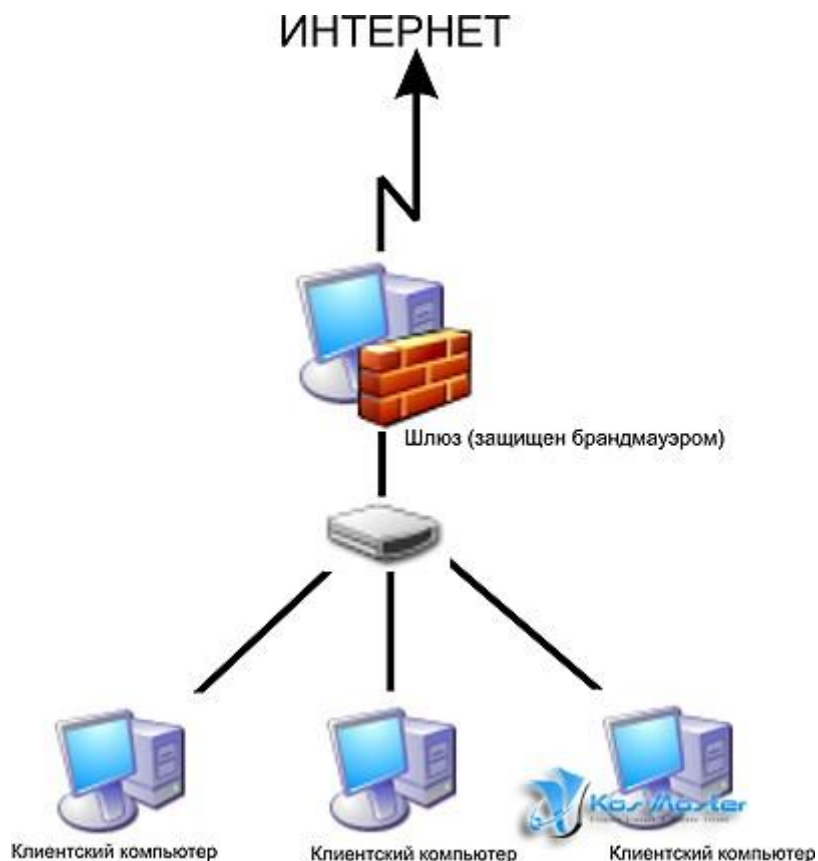
- определение типа создаваемой хранимой процедуры: временная или пользовательская. Кроме этого, можно создать свою собственную системную хранимую процедуру, назначив ей имя с префиксом `sp_` и поместив ее в системную базу данных. Такая процедура будет доступна в контексте любой базы данных локального сервера;
- планирование прав доступа. При создании хранимой процедуры следует учитывать, что она будет иметь те же права доступа к объектам базы данных, что и создавший ее пользователь;
- определение параметров хранимой процедуры. Подобно процедурам, входящим в состав большинства языков программирования, хранимые процедуры могут обладать входными и выходными параметрами;
- разработка кода хранимой процедуры. Код процедуры может содержать последовательность любых команд SQL, включая вызов других хранимых процедур.

Создание новой и изменение имеющейся хранимой процедуры осуществляется с помощью следующей команды:

```
<определение процедуры> ::=  
{CREATE | ALTER } PROC[EDURE] имя_процедуры  
    [ ;номер]  
    [{@имя_параметра тип_данных } [VARYING ]  
        [=default][OUTPUT] ][,...n]  
    [WITH { RECOMPILE | ENCRYPTION | RECOMPILE,  
        ENCRYPTION }]  
    [FOR REPLICATION]  
AS  
    sql_оператор [...n]
```

Брандмауэр

Многие организации присоединили или хотят присоединить свои локальные сети к Интернету, чтобы их пользователи имели легкий доступ к сервисам Интернета. Так как Интернет в целом не является безопасным, машины в этих ЛВС уязвимы к неавторизованному использованию и внешним атакам. Брандмауэр - это средство защиты, которое можно использовать для управления доступом между надежной сетью и менее надежной. Брандмауэр - это не одна компонента, а стратегия защиты ресурсов организации, доступных из Интернета. Брандмауэр выполняет роль стражи между небезопасным Интернетом и более надежными внутренними сетями.



Основная функция брандмауэра - централизация управления доступом. Если удаленные пользователи могут получить доступ к внутренним сетям в обход брандмауэра, его эффективность близка к нулю. Например, если менеджер, находящийся в командировке, имеет модем, присоединенный к его ПЭВМ в офисе, то он может дозвониться до своего компьютера из командировки, а так как эта ПЭВМ также находится во внутренней защищенной сети, то атакующий, имеющий возможность установить коммутируемое соединение с этой ПЭВМ, может обойти защиту брандмауэра. Если пользователь имеет подключение к Интернету у какого-нибудь провайдера Интернета, и часто соединяется с Интернетом со своей рабочей машины с помощью модема, то он или она устанавливают небезопасное соединение с Интернетом, в обход защиты брандмауэра.

Брандмауэры часто могут быть использованы для защиты сегментов интранета организации, но этот документ в основном будет описывать проблемы, связанные с Интернетом. Более подробная информация о брандмауэрах содержится в " NIST Special Publication 800-10 "Keeping Your Site Comfortably Secure: An Introduction to Internet Firewalls."

Брандмауэры обеспечивают несколько типов защиты:

- Они могут блокировать нежелательный трафик
- Они могут направлять входной трафик только к надежным внутренним системам
- Они могут скрыть уязвимые системы, которые нельзя обезопасить от атак из Интернета другим способом.
- Они могут протоколировать трафик в и из внутренней сети
- Они могут скрывать информацию, такую как имена систем, топологию сети, типы сетевых устройств и внутренние идентификаторы пользователей, от Интернета
- Они могут обеспечить более надежную аутентификацию, чем та, которую представляют стандартные приложения.

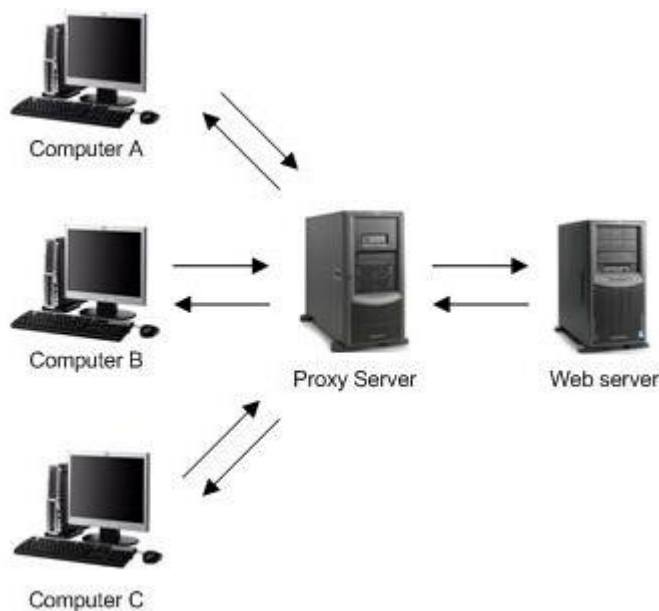
Каждая из этих функций будет описана далее.

Как и для любого средства защиты, нужны определенные компромиссы между удобством работы и безопасностью. Прозрачность - это видимость брандмауэра как внутренним пользователям, так и внешним, осуществляющим взаимодействие через брандмауэр. Брандмауэр прозрачен для пользователей, если он не мешает им получить доступ к сети. Обычно брандмауэры конфигурируются так, чтобы быть прозрачными для внутренних пользователей сети (посылающим пакеты наружу за брандмауэр); и с другой стороны брандмауэр конфигурируется так, чтобы быть непрозрачным для внешних пользователей, пытающихся получить доступ к внутренней сети извне. Это обычно обеспечивает высокий уровень безопасности и не мешает внутренним пользователям.

Прокси-серверы

Прокси-сервер (*proxy-server*) — это служба в компьютерных сетях, позволяющая клиентам выполнять косвенные запросы к другим сетевым службам.

Сначала клиент подключается к прокси-серверу и запрашивает какой-либо ресурс, расположенный на другом сервере. Затем прокси-сервер либо подключается к указанному серверу и получает ресурс у него, либо возвращает ресурс из собственного кеша (если имеется). В некоторых случаях запрос клиента или ответ сервера может быть изменен прокси-сервером в определенных целях. Также прокси-сервер позволяет защищать клиентский компьютер от некоторых сетевых атак.



Чаще всего прокси-серверы применяются для следующих целей:

- обеспечение доступа с компьютеров локальной сети в Интернет;
- кэширование данных: если часто происходят обращения к одним и тем же внешним ресурсам, то можно держать их копию на прокси-сервере и выдавать по запросу, снижая тем самым нагрузку на канал во внешнюю сеть и ускоряя получение клиентом запрошенной информации.

- сжатие данных: прокси-сервер загружает информацию из Интернета и передает информацию конечному пользователю в сжатом виде.
- защита локальной сети от внешнего доступа: например, можно настроить прокси-сервер так, что локальные компьютеры будут обращаться к внешним ресурсам только через него, а внешние компьютеры не смогут обращаться к локальным вообще (они "видят" только прокси-сервер).
- ограничение доступа из локальной сети к внешней: например, можно запретить доступ к определенным WEB-сайтам, ограничить использование интернета каким-то локальным пользователям, устанавливать квоты на трафик или полосу пропускания, фильтровать рекламу и вирусы.
- анонимизация доступа к различным ресурсам. Прокси-сервер может скрывать сведения об источнике запроса или пользователе. В таком случае целевой сервер видит лишь информацию о прокси-сервере, например, IP-адрес, но не имеет возможности определить истинный источник запроса. Существуют также искажающие прокси-серверы, которые передают целевому серверу ложную информацию об истинном пользователе.

Темы для подготовки

Атрощенко Александр Михайлович	Файловые серверы
Зиниатов Ильнур Мансурович	Принт-сервер
Карпов Антон Сергеевич	Характеристика сервера Apache
Колегов Владислав Андреевич	Почтовый сервер
Котин Егор Евгеньевич	Серверы приложений
Михайлов Олег Вячеславович	Роль сервера приложений
Наймушин Никита Александрович	Интерфейс сервера приложений
Петрунин Никита Александрович	Веб-сервер
Пигильцев Иван Сергеевич	Характеристика Internet Information Server от Microsoft
Ражукас Владлен Юрьевич	Хранимые процедуры
Рудковский Иван Русланович	Брандмауэр
Сабитова Надежда Максимовна	Прокси-серверы
Сидоров Никита Андреевич	Электронная почта
Силантьев Влад Борисович	Структура адреса электронной почты
Тепляков Тимур Николаевич	Что такое домен
Фаяршина Эльза Илнуровна	Распространённые серверы электронной почты
Фонарюк Светлана Александровна	Программное обеспечение электронной почты
Халанская Виктория Сергеевна	Mail.ru
Чашин Станислав Владимирович	Обзор веб-серверов
Чернавских Владимир Владимирович	Серверы Java-приложений
Шалапугина Екатерина Андреевна	Управляющий сервер
Шамсумухаметов Айдар Альфитович	Хранимые процедуры в среде MS SQL Server
Шредер Павел Юрьевич	Дисковый сервер