Оглавление

Классификация программного обеспечения

1.1 Что такое программное обеспечение

В компьютерном жаргоне часто используется слово «софт» от английского software, которое, в этом смысле впервые применил в статье American Mathematical Monthly математик из Принстонского университета Джон Тьюки (John W. Tukey) в 1958 г.

К программному обеспечению (ПО) относится также вся область деятельности по проектированию и разработке ПО:

- технология проектирования программ (например, нисходящее проектирование, структурное и объектно-ориентированное проектирование и др.);

- методы тестирования программ;

- методы доказательства правильности программ;

- анализ качества работы программ;

- документирование программ;

- разработка и использование программных средств, облегчающих процесс проектирования программного обеспечения, и многое другое.

Программное обеспечение – неотъемлемая часть компьютерной системы. Оно является логическим продолжением технических средств. Сфера применения конкретного компьютера определяется созданным для него ПО. Сам по себе компьютер не обладает знаниями ни в одной области применения. Все эти знания сосредоточены в выполняемых на компьютерах программах.

Программное обеспечение в настоящее время составляет сотни тысяч программ, которые предназначены для обработки самой разнообразной информации с самыми различными целями.

1.2 Виды программного обеспечения

Все программы, работающие на компьютере, можно условно разделить на три вида (рис. 1.):

- прикладные программы, непосредственно обеспечивающие выполнение необходимых пользователям работ;

- системные программы, предназначены для управления работой вычислительной системы, выполняют различные вспомогательные функции, например:

- управление ресурсами компьютера;

- создание копий используемой информации;

- проверка работоспособности устройств компьютера;

- выдача справочной информации о компьютере и др.;

- инструментальные программные системы, облегчающие процесс создания новых программ для компьютера.

Рисунок 1 Программы, работающие на компьютере.

При построении классификации ПО нужно учитывать тот факт, что стремительное развитие вычислительной техники и расширение сферы приложения компьютеров резко ускорили процесс эволюции программного

обеспечения. Если раньше можно было легко перечислить основные категории ПО — операционные системы, трансляторы, пакеты прикладных программ, то сейчас ситуация коренным образом изменилась. Развитие ПО пошло как вглубь (появились новые подходы к построению операционных систем, языков программирования и т.д.), так и вширь (прикладные программы перестали быть прикладными и приобрели самостоятельную ценность). Соотношение между требующимися программными продуктами и имеющимися на рынке меняется очень быстро. Даже классические программные продукты, такие, как операционные системы, непрерывно развиваются и наделяются интеллектуальными функциями, многие из которых ранее относились только к интеллектуальным возможностям человека.

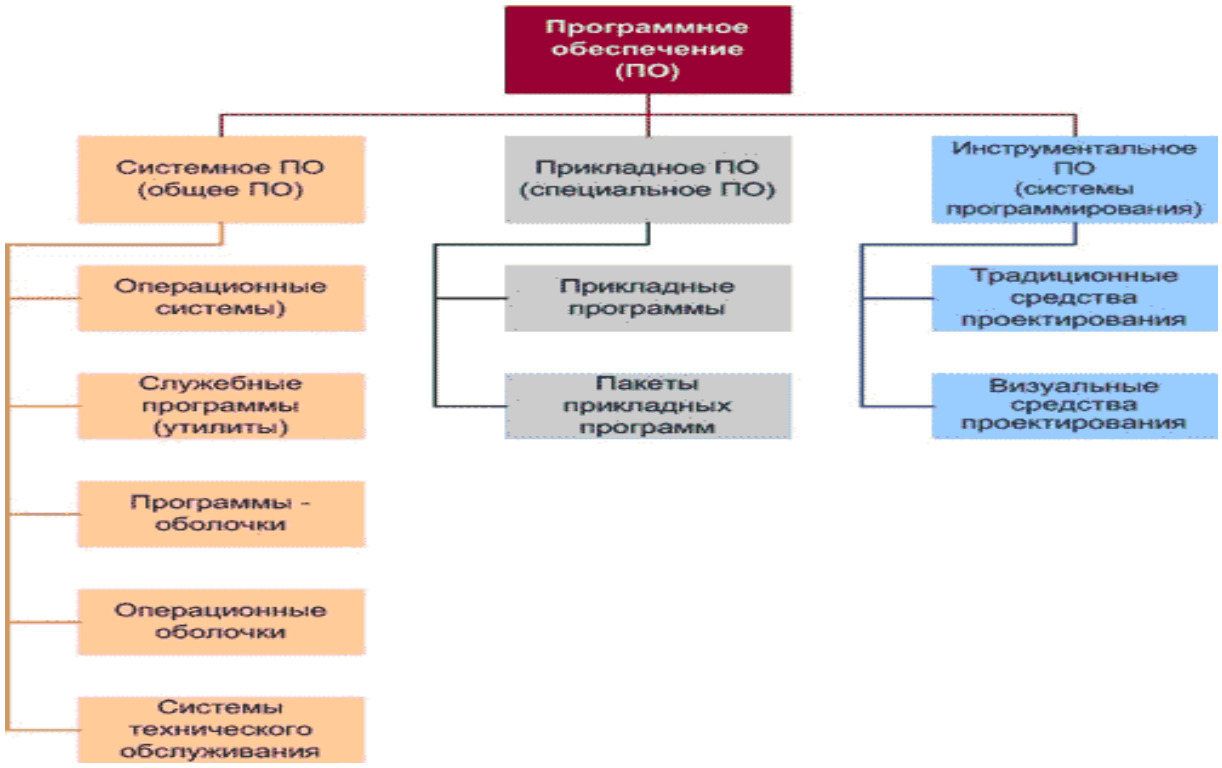


Рисунок 2 Основные категории ПО

1.3 Прикладное программное обеспечение

Какие программы называют прикладными.

Прикладная программа — это любая конкретная программа, способствующая решению какой-либо задачи в пределах данной проблемной области.

Прикладные программы могут носить и общий характер, например, обеспечивать составление и печатание документов и т.п.

В противоположность этому, операционная система или инструментальное ПО не вносят прямого вклада в удовлетворение конечных потребностей пользователя.

Прикладные программы могут использоваться либо автономно, то есть решать поставленную задачу без помощи других программ, либо в составе программных комплексов или пакетов.

Наиболее часто встречающееся прикладное ПО.

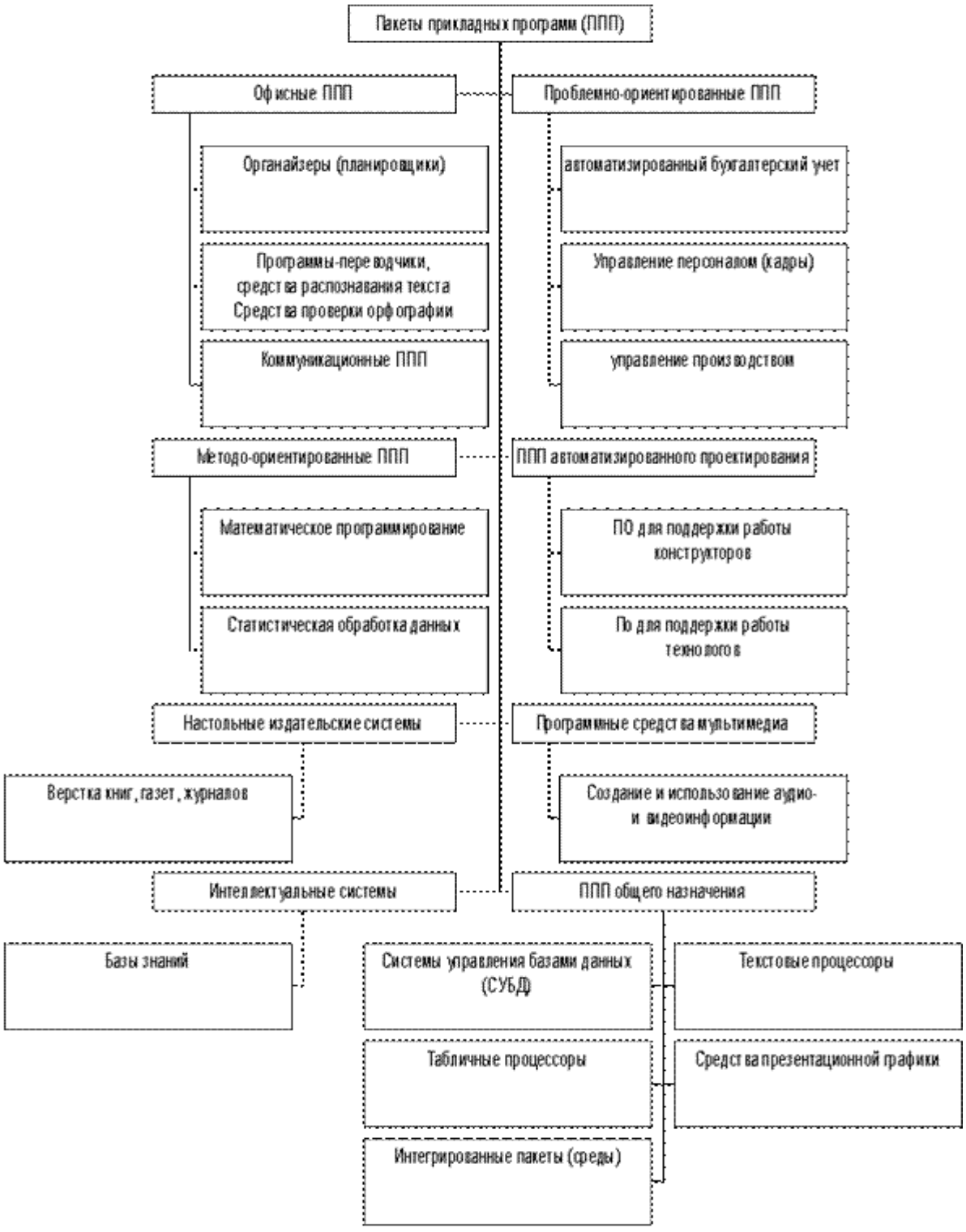


Рисунок 3 Пакеты прикладных программ.

Редакторы документов – это наиболее широко используемый вид прикладных программ. Они позволяют подготавливать документы гораздо быстрее и удобнее, чем с помощью пишущей машинки. Текстовые

редакторы могут обеспечивать выполнение разнообразных функций, а именно:

- редактирование строк текста;

- возможность использования различных шрифтов символов;

- копирование и перенос части текста с одного места на другое или из одного документа в другой;

- контекстный поиск и замена частей текста;

- задание произвольных межстрочных промежутков;

- автоматический перенос слов на новую строку;

- автоматическая нумерацию страниц;

- обработка и нумерация сносок;

- выравнивание краев абзаца;

- создание таблиц и построение диаграмм;

- проверка правописания слов и подбор синонимов;

- построение оглавлений и предметных указателей;

- распечатка подготовленного текста на принтере в нужном числе экземпляров и т.п.

Возможности текстовых редакторов различны — от программ, предназначенных для подготовки небольших документов простой структуры, до программ для набора, оформления и полной подготовки к типографскому изданию книг и журналов (издательские системы).

Представители редакторов документов – программы Microsoft Word, Wordpad, Microsoft Publisher, Corel Ventua и Adobe ageMaker.

Табличные процессоры. При работе с табличным процессором на экран выводится прямоугольная таблица, в клетках которой могут находиться числа, пояснительные тексты и формулы для расчета значения в клетке по именующимся данным. Все распространенные табличные процессоры позволяют вычислять значения элементов таблиц по заданным формулам, строить по данным в таблицах различные графики и т.д.

Табличные процессоры представляют собой удобное средство для проведения бухгалтерских и статистических расчетов. В каждом пакете имеются сотни встроенных математических функций и алгоритмов статистической обработки данных. Кроме того, имеются мощные средства для связи таблиц между собой, создания и редактирования электронных баз данных.

Специальные средства позволяют автоматически получать и распечатывать настраиваемые отчеты с использованием десятков различных типов таблиц, графиков, диаграмм, снабжать их комментариями и графическими иллюстрациями.

Табличные процессоры имеют встроенную справочную систему, предоставляющую пользователю информацию по конкретным командам меню и другие справочные данные. Многомерные таблицы позволяют быстро делать выборки в базе данных по любому критерию.

Представители семейства табличных процессоров: Microsoft Excel, Quatro Pro, Lotus 1-2-3.

Графические редакторы позволяют создавать и редактировать рисунки. В простейших редакторах предоставляются возможности рисования линий, кривых, раскраски областей экрана, создание надписей различными шрифтами и т.д. Большинство редакторов позволяют обрабатывать

изображения, полученные с помощью сканеров. Представители графических редакторов – программы Adobe Photoshop, Corel Draw.

Правовые базы данных содержат тексты нормативных документов и предоставляют возможности справки, контекстного поиска, распечатки и т.д. Представители правовых баз данных – пакеты Гарант и Консультант+.

Системы автоматизированного проектирования (САПР) или CAD (англ. Computer-Aided Design) — программный пакет, предназначенный для создания чертежей, конструкторской и/или технологической документации и/или 3D моделей. Среди систем малого и среднего класса в мире наиболее популярна система AutoCad фирмы AutoDesk. Отечественный пакет с аналогичными функциями – Компас.

Существуют остроумные способы визуализации наиболее простых многомерных объектов — множеств точек. Один из них носит название "лица Чернова" (Чернов — современный американский математик). {there must be pictures here}. Этим способом можно отображать 10-20-мерные множества. Суть способа такова: каждому из измерений сопоставляется один из параметров схематически изображённого человеческого лица, например, первое измерение дает отношение высоты лица к ширине, второе – размер носа, третье – расстояние между глазами и т.д. Таким образом, каждой точке исходного множества будет сопоставлено лицо. Рассматривая эти лица, можно отобрать похожие между собой или же выделить абсолютно непохожие и тем самым произвести некую классификацию исходного множества.

Системы управления базами данных (СУБД) позволяют управлять большими информационными массивами - базами данных. Программные системы этого вида позволяют обрабатывать на компьютере массивы информации,

обеспечивают ввод, поиск, сортировку выборку записей, составление отчетов и т.д. Представители данного класса программ – Microsoft Access, Clipper, Paradox, FoxPro.

Интегрированные системы сочетают в себе возможность системы управления базами данных, табличного процессора, текстового редактора, системы деловой графики, а иногда и другие возможности. Как правило, все компоненты интегрированной системы имеют схожий интерфейс, что облегчает обучение работе с ними. Представители интегрированных систем – пакет Microsoft Office и его бесплатный аналог Open Office.

1.4 Системные программы

Роль и назначение системных программ.

Системные программы выполняются вместе с прикладными и служат для управления ресурсами компьютера — центральным процессором, памятью, вводом-выводом.

Это программы общего пользования, которые предназначены для всех пользователей компьютера. Системное программное обеспечение разрабатывается так, чтобы компьютер мог эффективно выполнять прикладные программы.

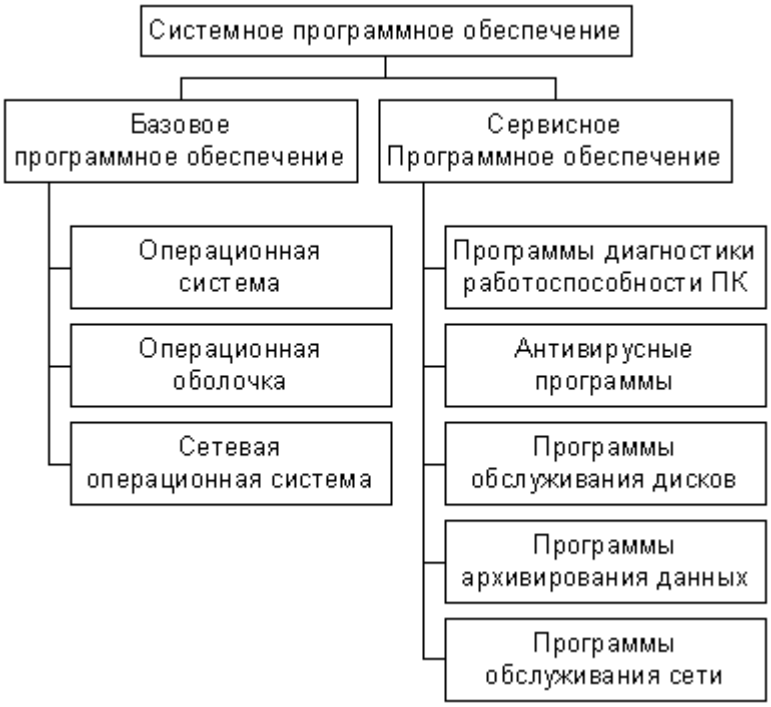


Рисунок 4 Пакеты системных программ

Системное программное обеспечение направлено:

- на создание операционной среды функционирования других программ;

- на обеспечение надежной и эффективной работы самого компьютера и вычислительной сети;

- на проведение диагностики и профилактики аппаратуры компьютера и вычислительных сетей;

- на выполнение вспомогательных технологических процессов (копирование, архивирование, восстановление файлов программ и баз данных и т.д.).

Данный класс программных продуктов тесно связан с типом компьютера и является его неотъемлемой частью. Программные продукты в основном ориентированы на квалифицированных пользователей — профессионалов в компьютерной области: системного программиста, администратора сети, прикладного программиста, оператора. Однако знание базовой технологии работы с этим классом программных продуктов требуется и конечным

пользователям персонального компьютера, которые самостоятельно не только работают со своими программами, но и выполняют обслуживание компьютера, программ и данных.

Программные продукты данного класса носят общий характер применения, независимо от специфики предметной области. К ним предъявляются высокие требования по надежности и технологичности работы, удобству и эффективности использования.

Структура системного программного обеспечения.

Системного программное обеспечение можно разделить на:

Базовое программное обеспечение (base software) — минимальный набор программных средств, обеспечивающих работу компьютера,(как правило, поставляется вместе с компьютером). В базовое программное обеспечение входят: операционная система; операционные оболочки (текстовые и графические); сетевая операционная система.

Сервисное программное обеспечение — программы и программные комплексы, которые расширяют возможности базового программного обеспечения и организуют более удобную среду работы пользователя – утилиты, (может быть приобретено дополнительно).

Глава II Комплексы программ

2.1 Что такое операционная система

Операционная система — это комплекс взаимосвязанных системных программ, назначение которого — организовать взаимодействие пользователя с компьютером и выполнение всех других программ.

Операционная система выполняет роль связующего звена между аппаратурой компьютера, с одной стороны, и выполняемыми программами, а также пользователем, с другой стороны.

Операционная система обычно хранится во внешней памяти компьютера — на диске. При включении компьютера она считывается с дисковой памяти и размещается в ОЗУ. Этот процесс называется загрузкой операционной системы.

В функции операционной системы входит:

- осуществление диалога с пользователем;

- ввод-вывод и управление данными;

- планирование и организация процесса обработки программ;

- распределение ресурсов (оперативной памяти и кэша, процессора, внешних устройств);

- запуск программ на выполнение;

- всевозможные вспомогательные операции обслуживания;

- передача информации между различными внутренними устройствами;

- программная поддержка работы периферийных устройств (дисплея, клавиатуры, дисковых накопителей, принтера и др.).

Анализ и исполнение команд пользователя, включая загрузку готовых программ из файлов в оперативную память и их запуск, осуществляет командный процессор операционной системы.

Операционную систему можно назвать программным продолжением устройства управления компьютера. Операционная система скрывает от пользователя сложные ненужные подробности взаимодействия с аппаратурой, образуя прослойку между ними. В результате этого люди освобождаются от очень трудоёмкой работы по организации взаимодействия с аппаратурой компьютера.

В Табл. 1. приведены некоторые характеристики некоторых наиболее распространенных ОС. Как видите, среди них нет ни одной операционной системы с чисто командным интерфейсом, и это неудивительно, так как эпоха интерфейсов, основанных на командной строке, медленно, но верно уходит в прошлое.

Название	Поддержка функций				Примеры
	Много-задачность	Много-поточность	GUI	SMD	
Windows 98				-	1. ПК, рабочие станции.
Windows 2000					2. ПК, рабочие станции, серверы, кластеры серверов.
System 7				-	3. ПК, Macintosh фирмы Apple.

Unix					4. ПК, раб. станции, кластеры серверов, миникомпьютеры
Linux					5. ПК, раб. станции, серверы.

Таблица 1

Кроме того, именно ОС обеспечивает возможность индивидуальной настройки компьютера: ОС определяет, из каких компонентов собран компьютер, на котором она установлена, и настраивает сама себя для работы именно с этими компонентами.

Ещё не так давно работы по настройке приходилось выполнять пользователю вручную, а сегодня производители компонентов компьютерной техники разработали протокол plug-and-play (включил - заработало). Этот протокол позволяет операционной системе в момент подключения нового компонента получить информацию о новом устройстве, достаточную для настройки ОС на работу с ним.

В зависимости от количества одновременно обрабатываемых задач и числа пользователей, которых могут обслуживать ОС, различают четыре основных класса операционных систем:

- однопользовательские однозадачные, которые поддерживают одну клавиатуру и могут работать только с одной (в данный момент) задачей;

- однопользовательские однозадачные с фоновой печатью, которые позволяют помимо основной задачи запускать одну дополнительную задачу, ориентированную, как правило, на вывод информации на печать. Это ускоряет работу при выдаче больших объёмов информации на печать;

- однопользовательские многозадачные, которые обеспечивают одному пользователю параллельную обработку нескольких задач. Например, к одному компьютеру можно подключить несколько принтеров, каждый из которых будет работать на "свою" задачу;

- многопользовательские многозадачные, позволяющие на одном компьютере запускать несколько задач нескольким пользователям. Эти ОС очень сложны и требуют значительных машинных ресурсов.

В различных моделях компьютеров используют операционные системы с разной архитектурой и возможностями. Для их работы требуются разные ресурсы. Они предоставляют разную степень сервиса для программирования и работы с готовыми программами.

2.2 Что такое программы-оболочки.

Оболочки — это программы, созданные для упрощения работы со сложными программными системами, такими, например, как DOS. Они преобразуют неудобный командный пользовательский интерфейс в дружественный графический интерфейс или интерфейс типа "меню". Оболочки предоставляют пользователю удобный доступ к файлам и обширные сервисные услуги.

Самая популярная у пользователей IBM-совместимого ПК оболочка — пакет программ Norton Commander. Он обеспечивает:

- создание, копирование, пересылку, переименование, удаление, поиск файлов, а также изменение их атрибутов;

- отображение дерева каталогов и характеристик входящих в них файлов в форме, удобной для восприятия человека;

- создание, обновление и распаковку архивов (групп сжатых файлов);

- просмотр текстовых файлов;

- редактирование текстовых файлов;

- выполнение из её среды практически всех команд DOS;

- запуск программ;

- выдачу информации о ресурсах компьютера;

- создание и удаление каталогов;

- поддержку межкомпьютерной связи;

- поддержку электронной почты через модем.

Что такое сетевые операционные системы.

Сетевые операционные системы — комплекс программ, обеспечивающий обработку, передачу и хранение данных в сети. Сетевая ОС предоставляет пользователям различные виды сетевых служб (управление файлами, электронная почта, процессы управления сетью и др.), поддерживает работу в абонентских системах. Сетевые операционные системы используют архитектуру клиент-сервер или одноранговую архитектуру. Они оцениваются по комплексу критериев: производительность, разнообразие возможностей связи пользователей, возможности администрирования.

2.3 Что такое утилиты

Важными классами системных программ являются также программы вспомогательного назначения — утилиты (лат. utilitas — польза). Они либо расширяют и дополняют соответствующие возможности операционной системы, либо решают самостоятельные важные задачи.

Кратко опишем некоторые разновидности утилит:

Программы контроля, тестирования и диагностики, которые используются для проверки правильности функционирования устройств компьютера и для обнаружения неисправностей в процессе эксплуатации; указывают причину и место неисправности;

Программы-драйверы, которые расширяют возможности операционной системы по управлению устройствами ввода-вывода, оперативной памятью и т.д.; с помощью драйверов возможно подключение к компьютеру новых устройств или нестандартное использование имеющихся;

Программы-упаковщики (архиваторы), которые позволяют за счет применения специальных алгоритмов упаковки информации сжимать информацию на дисках, т.е. создавать копии файлов меньшего размера, а также объединять копии нескольких файлов в один архивный файл. Применение программ-архиваторов очень полезно при создании архива файлов, так как в большинстве случаев значительно удобнее их хранить, предварительно сжав программами-архиваторами. Представители данных программ –WinRar и WinZip.

Антивирусные программы, предназначенные для предотвращения заражения компьютерными вирусами и ликвидации последствий заражения вирусами. Компьютерный вирус — это специально написанная небольшая по размерам программа, которая может "приписывать" себя к другим программам для выполнения каких-либо вредных действий — портит файлы, "засоряет оперативную память и т.д. Представители антивирусного семейства программ – Kaspersky Antivirus, DrWeb, Norton Antivirus.

Рисунок 5 Классификация вирусов

Согласно исследованию организации AVIEWS (Antivirus Information & Early Warning System), Sophos обнаруживает не менее 80 процентов неизвестных зловредных кодов, значительно опережая многие другие весьма популярные и именитые программы. Второе место занял «Антивирус Касперского», который обнаруживает 65 процентов угроз. Интересно, что третье место с 60 процентами занял Ikarus, не известная широким массам программа. А такая именитая программа, как Panda, показала всего 10%.

Программы для создания резервных копий информации позволяют периодически копировать важную информацию, находящуюся на жестком диске компьютера, на дополнительные носители. Представители программ резервного копирования – APBackUp, Acronis True Image.

Программы оптимизации и контроля качества дискового пространства;

Программы восстановления информации, форматирования, защиты данных;

Коммуникационные программы, предназначены для организации обмена информацией между компьютерами. Это программы позволяют удобно пересылать файлы с одного компьютера на другой при соединении кабелем их последовательных портов. Другой вид таких программ обеспечивает возможность связи компьютеров по телефонной сети (при наличии модема). Они дают возможность посылать и принимать телефаксные сообщения. Представители коммуникационных программ – Venta Fax, Cute FTP.

Программы для управления памятью, обеспечивающие более гибкое использование оперативной памяти;

Программы для печати экрана бывают весьма полезны при использовании графических программ для вывода на печать содержимого экрана, так как отнюдь не всегда это можно сделать с помощью самой графической программы. Представители программ для печати экрана – SnagIt, HyperSnap-DX.

Программы для записи CD-ROM, CD-R и многие другие.

Часть утилит входит в состав операционной системы, а другая часть функционирует независимо от нее, т.е. автономно.

2.4 Инструментальные системы

Какие программы называются инструментальными.

Инструментальные программные средства — это программы, которые используются в ходе разработки, корректировки или развития других прикладных или системных программ.

Инструментальные программные средства могут оказать помощь на всех стадиях разработки ПО. По своему назначению они близки системам программирования.

К инструментальным программам, например, относятся:

- редакторы;

- средства компоновки программ;

- отладочные программы, т.е. программы, помогающие находить и устранять ошибки в программе;

- вспомогательные программы, реализующие часто используемые системные действия;

- графические пакеты программ и т.п.

Система программирования.

Система программирования — это система для разработки новых программ на конкретном языке программирования.

Современные системы программирования обычно предоставляют пользователям мощные и удобные средства разработки программ. В них входят:

- компилятор или интерпретатор;

- интегрированная среда разработки;

- средства создания и редактирования текстов программ;

- обширные библиотеки стандартных программ и функций;

- отладочные программы, т.е. программы, помогающие находить и устранять ошибки в программе;

- "дружественная" к пользователю диалоговая среда;

- многооконный режим работы;

- мощные графические библиотеки; утилиты для работы с библиотеками;

- встроенный ассемблер;

- встроенная справочная служба;

- другие специфические особенности.

Транслятор (англ. translator — переводчик) — это программа-переводчик. Она преобразует программу, написанную на одном из языков высокого уровня, в программу, состоящую из машинных команд.

Трансляторы реализуются в виде компиляторов или интерпретаторов. С точки зрения выполнения работы компилятор и интерпретатор существенно различаются.

Компилятор (англ. compiler — составитель, собиратель) читает всю программу целиком, делает ее перевод и создает законченный вариант программы на машинном языке, который затем и выполняется.

Интерпретатор (англ. interpreter — истолкователь, устный переводчик) переводит и выполняет программу строка за строкой.

После того, как программа откомпилирована, ни сама исходная программа, ни компилятор более не нужны. В то же время программа, обрабатываемая интерпретатором, должна заново переводиться на машинный язык при каждом очередном запуске программы.

Откомпилированные программы работают быстрее, но интерпретируемые проще исправлять и изменять.

Популярные системы программирования – Turbo Basic, Quick Basic, Turbo Pascal, Turbo C. Borland C++, Borland Delphi и др.

Каждый конкретный язык ориентирован либо на компиляцию, либо на интерпретацию — в зависимости от того, для каких целей он создавался. Например, Pascal обычно используется для решения довольно сложных задач, в которых важна скорость работы программ. Поэтому данный язык обычно реализуется с помощью компилятора. С другой стороны, Basic

создавался как язык для начинающих программистов, для которых построчное выполнение программы имеет неоспоримые преимущества.

Иногда для одного языка имеется и компилятор, и интерпретатор. В этом случае для разработки и тестирования программы можно воспользоваться интерпретатором, а затем откомпилировать отлаженную программу, чтобы повысить скорость ее выполнения.

2.5 Тенденции развития программного обеспечения

Бурный рост и быстрые темпы развития рынка ПО.

Создание программного обеспечения для персональных компьютеров за последнее десятилетие превратилось из занятия отдельных программистов в важную и мощную сферу промышленности. Поэтому развитие программного обеспечения, предназначенного для широкого круга пользователей, происходит в процессе ожесточенной конкурентной борьбы между фирмами-производителями программного обеспечения. Доля некоммерческого программного обеспечения постоянно снижается и все более ограничивается программами, создаваемыми в процессе научных исследований или для собственного использования.

При разработке коммерческих программ основной задачей фирм-разработчиков является, естественно, обеспечение их успеха на рынке. Для этого необходимо, чтобы программы обладали следующими качествами:

- функциональность программы, т.е. полнота удовлетворения ею потребностей пользователя;

- наглядный, удобный, интуитивно понятный и привычный пользователю интерфейс (т.е. способ взаимодействия программы с пользователем);

- простота освоения программы даже начинающими пользователями, для чего используются информативные подсказки, встроенные справочники и подробная документация;

- надежность программы, т.е. устойчивость ее к ошибкам пользователя, отказам оборудования и т.д., и разумные ее действия в этих ситуациях.

Расширяется практика сдачи программного обеспечения в аренду.

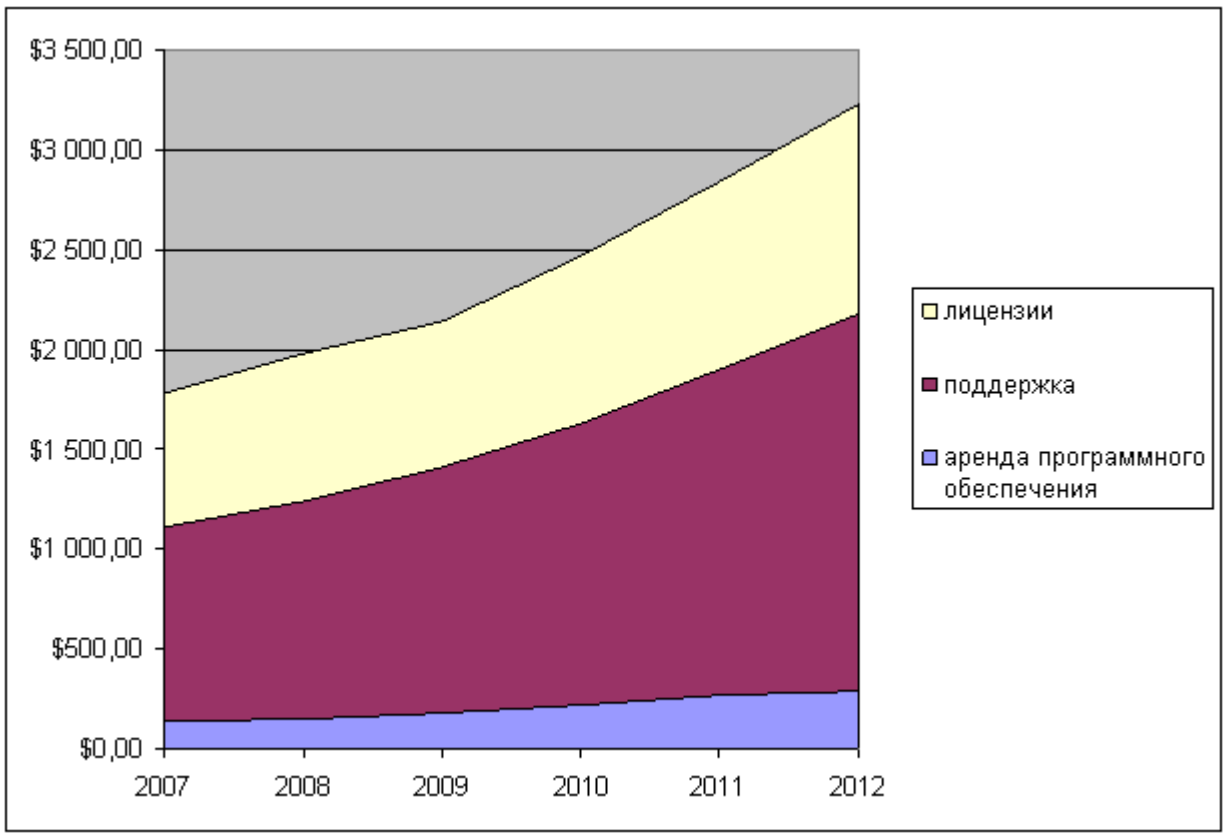


Диаграмма 1 Лицензии, поддержка и аренда ПО.

Стандартизация и интеграция продуктов ПО.

Во многих областях совместная работа различных производителей программного обеспечения приводит к стандартизации отдельных элементов интерфейса программ, форматов данных и т.д., что весьма удобно для пользователей. Это происходит прежде всего потому, что разработчики

программ перенимают друг у друга удачные находки и приемы и стремятся обеспечить совместимость с другими наиболее популярными программами.

Увеличение мощности программ.

Важнейшей тенденцией развития программного обеспечения является неуклонное увеличение их мощности – программы могут обрабатывать большие количества данных, делать это быстрее, предоставляют пользователю больше выполняемых функций и т.д. Таким образом, разработчики программного обеспечения используют возможности, появляющиеся из-за увеличения мощности компьютеров. Весьма заметно и стремление к интеграции функций программного обеспечения.

Возможность дистанционного запуска ПО через Web.

Сегодня большинство систем плавно перетекают в Web. Всемирная паутина затягивает все больше и больше приложений. Базы данных приобретают Web-интерфейсы пользователей, взамен имеющихся ранее настольных приложений. В конечном итоге, стоит ожидать, что конечному пользователю будет нужен лишь веб-браузер, чтобы иметь возможность удовлетворять все возможные потребности в программном обеспечении. В данном случае пользователю все равно, какая операционная система управляет локальным компьютером, главное - надежность и производительность сервера. (Например, пакет Microsoft Office может быть установлен на удаленных серверах, а не на системах конечных пользователей, но запуск приложений при этом будет происходить не менее быстро, чем на локальных ПК). Таким образом, все программы получат возможность как локального исполнения, так и дистанционного запуска через Web.

Удаленные рабочие столы. Специализированное программное обеспечение, применяемое в процессе тестирования.

Удаленный рабочий стол (Remote Desktop) — это термин, которым обозначается режим управления, когда один компьютер получает права администратора по отношению к другому (удаленному). Связь между устройствами происходит в реальном времени посредством сети Интернет или локальной сети.

Уровень доступа в режиме удаленного администрирования определяется конкретными задачами и может быть изменен по необходимости.

Например:

- в одном случае подключение к рабочей сессии дает возможность полного контроля и взаимодействия с удаленным компьютером, при котором допускается запуск на нем приложений и манипуляции с файлами;
- в другом - удаленный доступ к рабочему столу позволяет лишь вести наблюдения за процессами, без вмешательства в работу его системы.

Удаленное администрирование — предустановленная функция практически в каждой известной сегодня операционной системе, одновременно с этим, существует довольно большое количество программ, которые делают данный процесс более удобным, добавляя в стандартные версии новые функции.

Существуют несколько видов удаленного администрирования:

- Компьютер–сеть - позволяет контролировать работу локальной сети офиса или интернет-кафе.
- Терминал–компьютер - упрощает связь пользователя с системой, например, платежные терминалы банков.
- Компьютер–компьютер - чаще всего применяемый в быту, однако легко решающий и серьезные управленческие задачи.

- Сеть–сеть - отличный инструмент при необходимости взаимодействия между удаленными корпоративными сетями.

Стоит отметить и возможности перекрестного удаленного администрирования между различными операционными системам.

В основном, данный режим используется администраторами сетей для максимально быстрого выявления и устранения программных или аппаратных сбоев и мониторинга систем.

С другой стороны, развитие «облачных» технологий, способствующих централизованному хранению огромных массивов информации в удаленных сетях и серверах, как нельзя лучше соответствует основным принципам режима.

Тенденцией последнего времени, стала разработка и внедрение удаленного администрирования на основе беспроводных систем. Они уступают по функциональности привычным физическим сетям, но неплохо справляются с задачами мониторинга, ведения статистики и управлением несложными сетевыми процессами.

Что такое RDP подключение

RDP подключение позволяет управлять компьютером удалённо. RDP считается так называемым "прикладным протоколом", который основан на TCP. После того, как устанавливается соединение на транспортном уровне, начинается инициализация сессии этого протокола. В рамках такой сессии и происходит передача данных.

После того, как фаза инициализации будет завершена, сервер начинает передавать на компьютер (где установлен RDP-клиент) графический вывод. Затем сервер ожидает, когда к нему поступят входные данные от устройств ввода-вывода (мышь и клавиатура). Если говорить простым языком, то принцип работы RPD выглядит следующим образом:

пользователь при помощи мыши и клавиатуры управляет компьютером удалённо.

Сегодня эта технология используется повсеместно. Она нужна, в первую очередь, для того, чтобы управлять компьютерными системами. Также многие наработки этой технологии используются и в робототехнике. Даже простейшая радиоуправляемая игрушка основана на схожем принципе. Человеку, в руках которого находится пульт управления, нужно использовать его, чтобы управлять движением радиоуправляемой игрушки.

Обычно для графического вывода используется копия дисплея, которая может передаваться в качестве изображения. Передача вывода происходит при помощи примитивов.

Для чего используется RDP

Сегодня протокол RDP может применяться в нескольких случаях:

1. В целях администрирования.
2. В целях получения доступа к любому серверу приложений.

Этот вид соединения применяется любыми операционными системами, которые выпускала компания Microsoft. Обычно все серверные версии ОС от Microsoft могут поддерживать сразу несколько удалённых подключения, а также 1 локальный вход в систему.

Если же речь идёт о клиентской версии Windows, то она может поддерживать исключительно 1 вход. Он может быть, как удалённым, так и локальным. Для получения разрешения удалённых подключений нужно включить удалённый доступ к рабочему столу. Это можно сделать прямо в свойствах рабочей станции.

Важно отметить, этот удалённый доступ возможен далеко не всегда. Удалённый доступ функционирует исключительно в версии Windows, которая предназначена для сервера. Ещё одно преимущество в данном случае заключается в том, общее число удалённых подключений не ограничено. С другой стороны, потребуется настройка сервера лицензий, а

также его активация. Этот сервер можно установить на любой отдельно взятый сетевой узел, а также - на сервер терминалов.

Некоторые пользователи не понимают, что удалённый доступ к любому серверу может быть обеспечен только в случае установки все необходимых лицензий на сервер лицензий.

Протокол удаленного рабочего стола и его возможности

Данный протокол может использоваться для самых разнообразных задач и целей. В частности, он нужен для:

1. Звуковой подсистемы компьютера.
2. Поддержки функционирования буфера обмена.
3. Последовательного порта.
4. Принтера или других аналогичных устройств (сканера, копира и МФУ).
5. В целях перенаправления файловой системы.

Как защитить RDP

Настройка безопасности RDP - это один из ключевых вопросов. Именно правильная настройка протокола влияет на то, насколько будет безопасно передавать данные. Это особенно актуально для государственных компьютерных систем, а также различных частных коммерческих компаний, которые хотят иметь надёжную защиту от конкурентов. Сертификат RDP предусматривает использование любого доступного подхода для обеспечения безопасности. Для RDP можно использовать либо встроенную подсистему безопасности, либо же внешнюю подсистему безопасности.

В случае если пользователь выберет встроенную подсистему, все функции по обеспечению безопасности будут реализованы средствами, которые изначально имеются в RDP. Речь идёт о процессах шифрования и аутентификации.

Если же будет выбрана внешняя подсистема безопасности, то пользователю придётся полагаться на надёжность таких внешних модулей,

как CredSSP и TLS. Преимущества этого способа обеспечения безопасности заключаются в крайне надёжной аутентификации и эффективном шифровании.

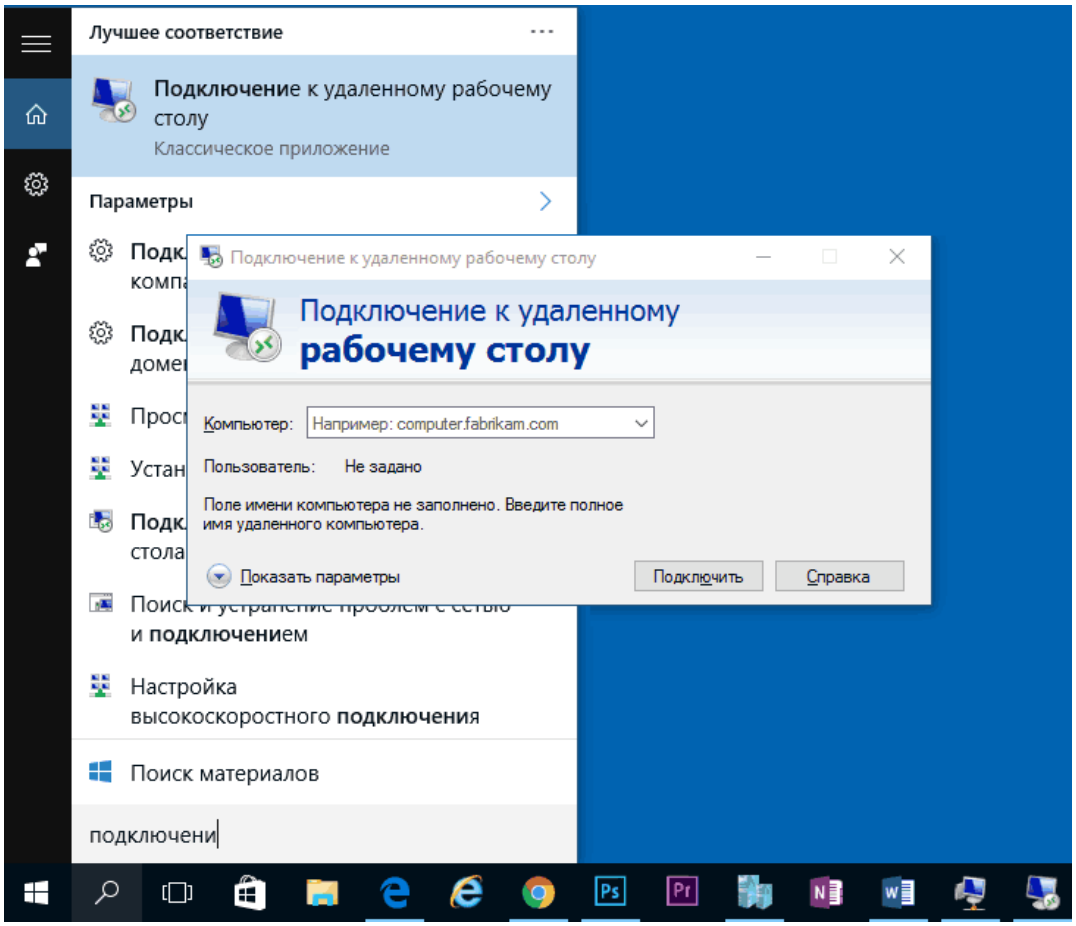
Именно качественное шифрование позволяет защитить свою систему от взлома злоумышленников. Сегодня существует масса вредоносных программ и алгоритмов, которые нужны для взлома. Более того, взлом может совершить не только квалифицированный специалист, но и рядовой пользователь, который приобрёл специальное вредоносное программное обеспечение.

Многих пользователей интересует вопрос о том, какой порт использует RDP (RDP сервер). Стандартный порт - это порт под номером 3389/TCP.

Программы для удаленного администрирования.

Для чего могут понадобиться такие программы? В большинстве случаев они используются для удаленного доступа к рабочему столу и действий для обслуживания компьютера системными администраторами и в сервисных целях. Однако, с точки зрения обычного пользователя, удаленное управление компьютером через Интернет или по локальной сети также может быть полезным: например, вместо установки виртуальной машины с Windows на ноутбуке Linux или Mac, можно подключаться к имеющемуся ПК с этой ОС (и это лишь один возможный сценарий).

Удаленный рабочий стол Microsoft (Microsoft Remote Desktop)

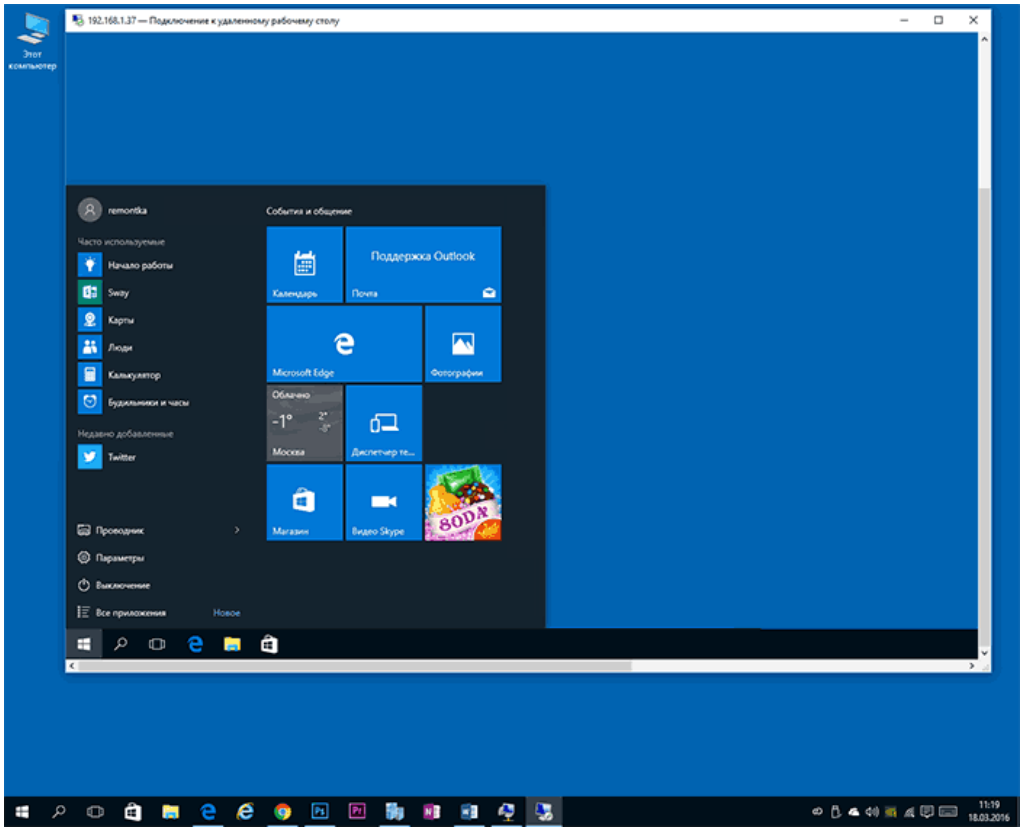


Удаленный рабочий стол Microsoft хорош тем, что для удаленного доступа компьютеру с его помощью не требуется установка какого-либо дополнительного программного обеспечения, при этом протокол RDP, который используется при доступе, в достаточной мере защищен и хорошо работает.

Но есть и недостатки. Прежде всего, в то время как подключиться к удаленному рабочему столу вы можете, не устанавливая дополнительных программ со всех версий Windows 7, 8 и Windows 10 (а также с других операционных систем, в том числе Android и iOS, загрузив бесплатный клиент Microsoft Remote Desktop), в качестве компьютера, к которому подключаются (сервера), может быть только компьютер или ноутбук с Windows Pro и выше.

Еще одно ограничение — без дополнительных настроек и изысканий подключение к удаленному рабочему столу Microsoft работает только, если

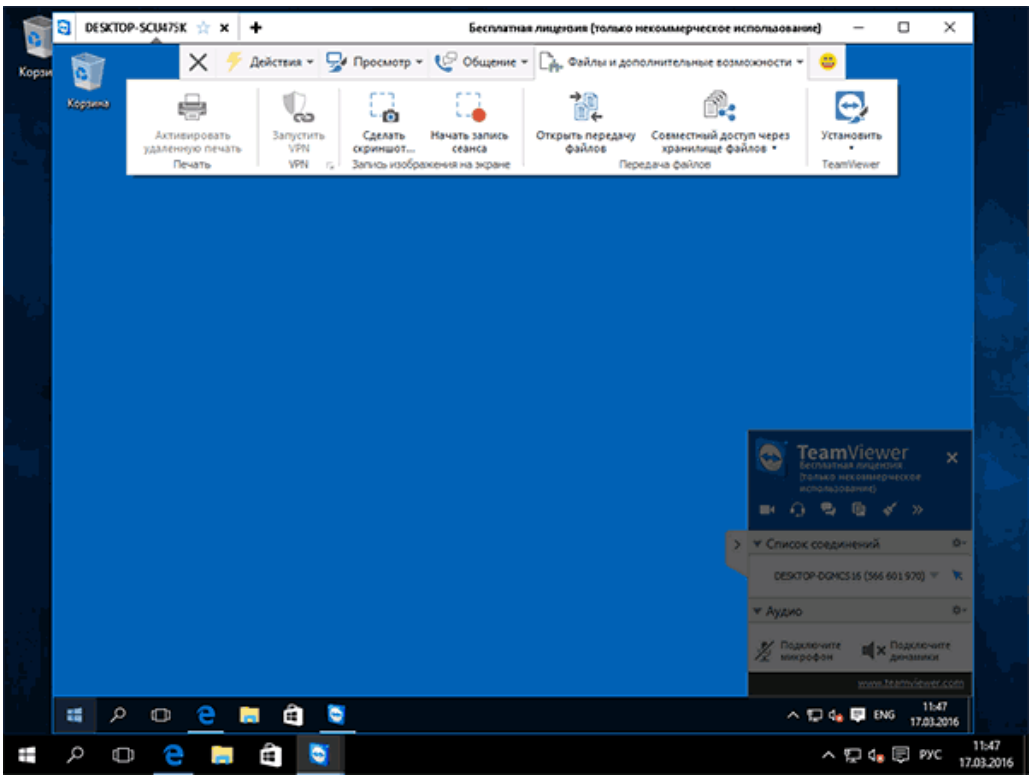
компьютеры и мобильные устройства находятся в одной локальной сети (например, подключены к одному роутеру, в случае домашнего использования) или же имеют статические IP в Интернете (при этом находятся не за маршрутизаторами).



Тем не менее, если у вас на компьютере установлена именно Windows 10 (8) Профессиональная, или Windows 7 Максимальная (как у многих), а доступ требуется только для домашнего использования, возможно, Microsoft Remote Desktop будет идеальным вариантом для вас.

TeamViewer

TeamViewer — наверное, самая известная программа для удаленного рабочего стола Windows и других ОС. Она на русском, проста в использовании, очень функциональна, отлично работает через Интернет и считается бесплатной для частного использования. Кроме этого, может работать без установки на компьютер, что полезно, если вам требуется лишь однократное подключение.

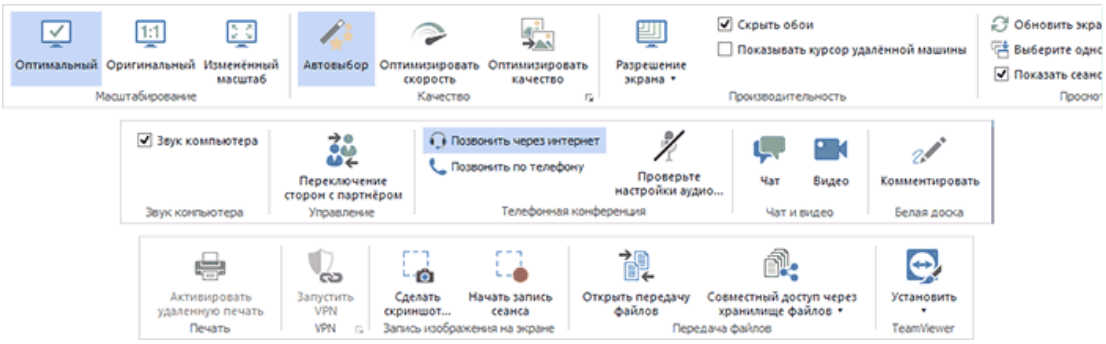


TeamViewer доступен в виде «большой» программы для Windows 7, 8 и Windows 10, Mac и Linux, совмещающей в себе функции сервера и клиента и позволяющей настроить постоянный удаленный доступ к компьютеру, в виде модуля TeamViewer QuickSupport, не требующего установки, который сразу после запуска выдает ID и пароль, которые требуется ввести на компьютере, с которого будет выполняться подключение. Существует дополнительный вариант - TeamViewer Host -для обеспечения возможности подключения к конкретному компьютеру в любое время. Также с недавних пор появился TeamViewer в виде приложения для Chrome, есть официальные приложения для iOS и Android.

Среди функций, доступных при сеансе удаленного управления компьютером в TeamViewer

- Запуск VPN соединения с удаленным компьютером.
- Удаленная печать.
- Создание скриншотов и запись удаленного рабочего стола.
- Общий доступ к файлам или просто передача файлов.
- Голосовой и текстовый чат, переписка, переключение сторон.

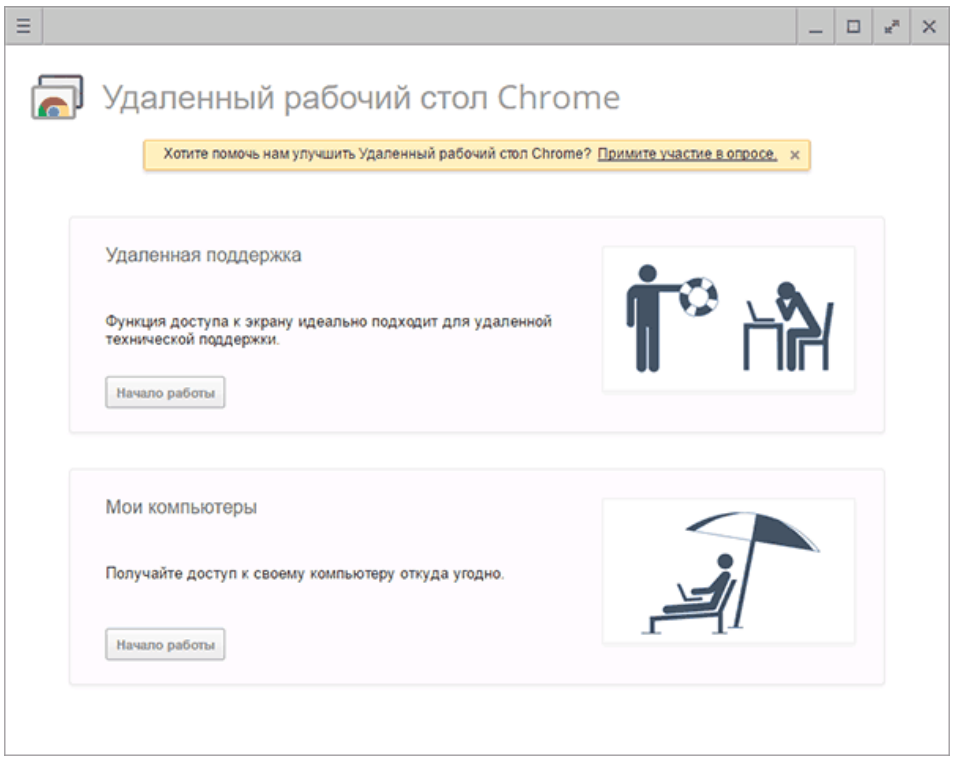
- Также TeamViewer поддерживает Wake-on-LAN, перезагрузку и автоматическое переподключение в безопасном режиме.



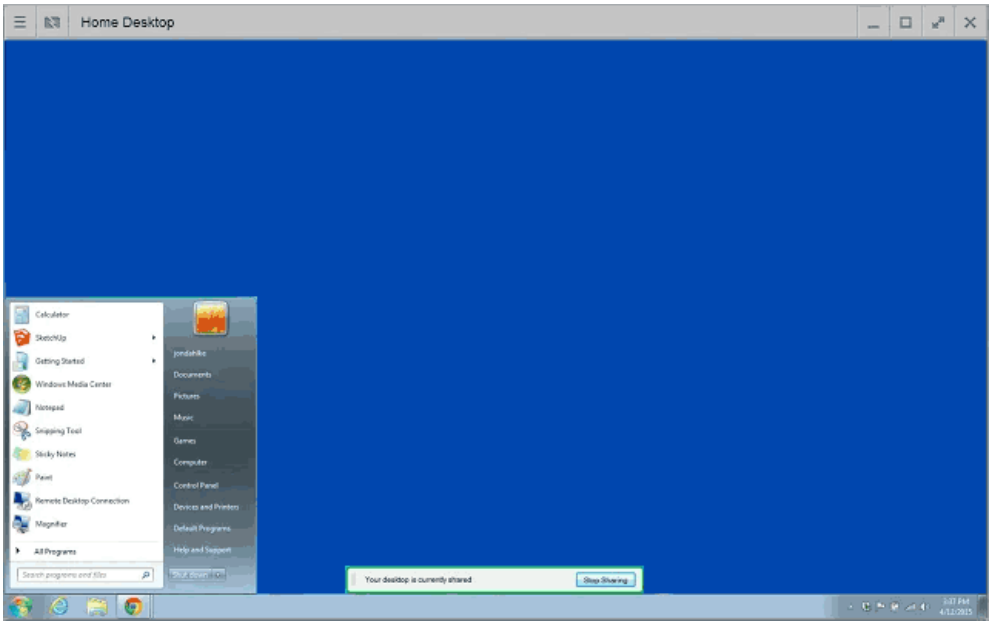
Подводя итог, TeamViewer — это тот вариант, который можно рекомендовать почти всем, кому потребовалась бесплатная программа для удаленного рабочего стола и управления компьютером в бытовых целях — в ней почти не придется разбираться, так как все интуитивно понятно и она проста в использовании. Для коммерческих целей придется покупать лицензию (в противном случае, Вы столкнетесь с тем, что сессии будут разрываться автоматически).

Удаленный рабочий стол Chrome (Chrome Remote Desktop)

Google имеет собственную реализацию удаленного рабочего стола, работающую как приложение для Google Chrome (при этом доступ будет не только к Chrome на удаленном компьютере, а ко всему рабочему столу). Поддерживаются все настольные операционные системы, на которые можно установить браузер Google Chrome. Для Android и iOS также имеются официальные клиенты в магазинах приложений.



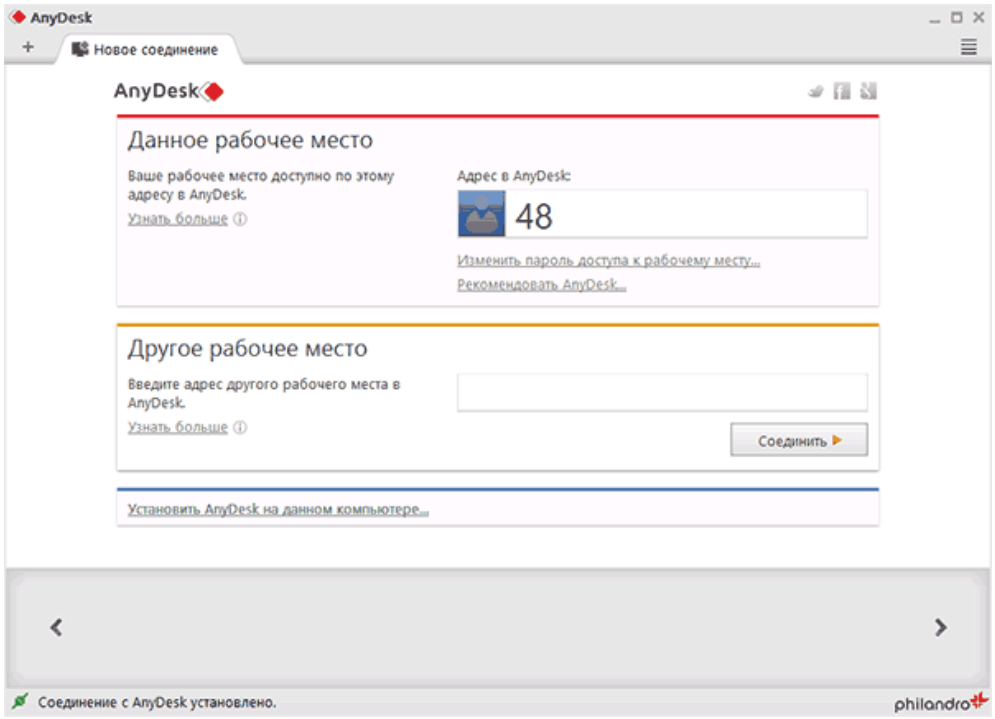
Для использования Chrome Remote Desktop потребуется загрузить расширение браузера из официального магазина, задать данные для доступа (пин-код), а на другом компьютере — подключиться с использованием этого же расширения и указанного пин-кода. При этом для использования удаленного рабочего стола Chrome обязательно требуется войти в свой аккаунт Google (не обязательно один и тот же аккаунт на разных компьютерах).



Среди преимуществ способа — безопасность и отсутствия необходимости установки дополнительного ПО, если вы и так пользуетесь браузером Chrome. Из недостатков — ограниченная функциональность.

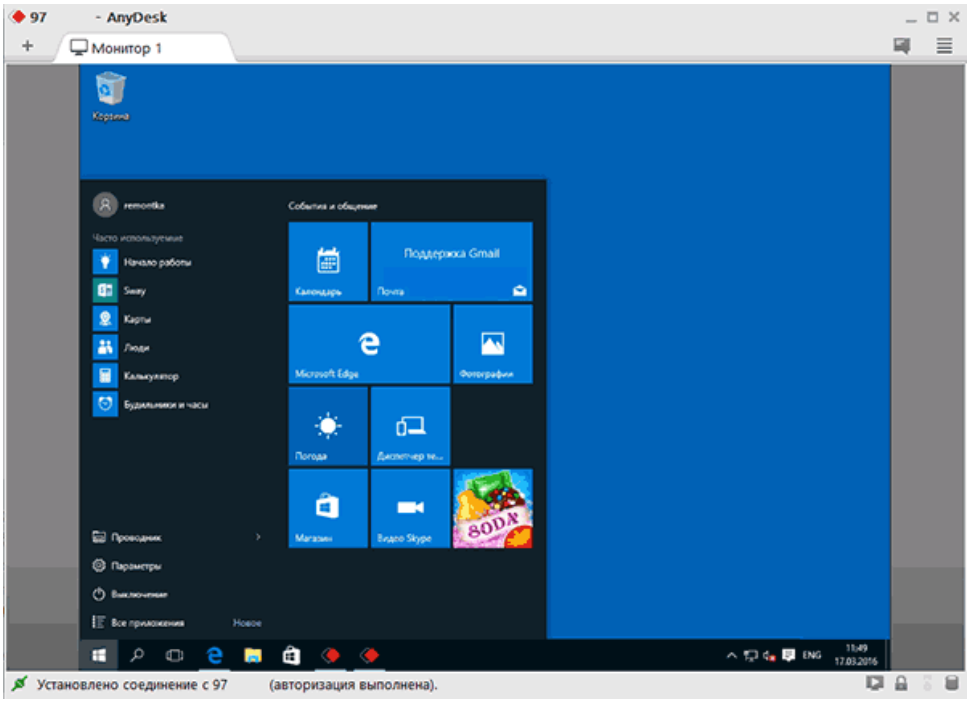
AnyDesk

AnyDesk — еще одна бесплатная программа для удаленного доступа к компьютеру, причем, создана она бывшими разработчиками TeamViewer. Среди преимуществ, которые заявляют создатели — высокая скорость работы (передачи графики рабочего стола) по сравнению с другими такими же утилитами.



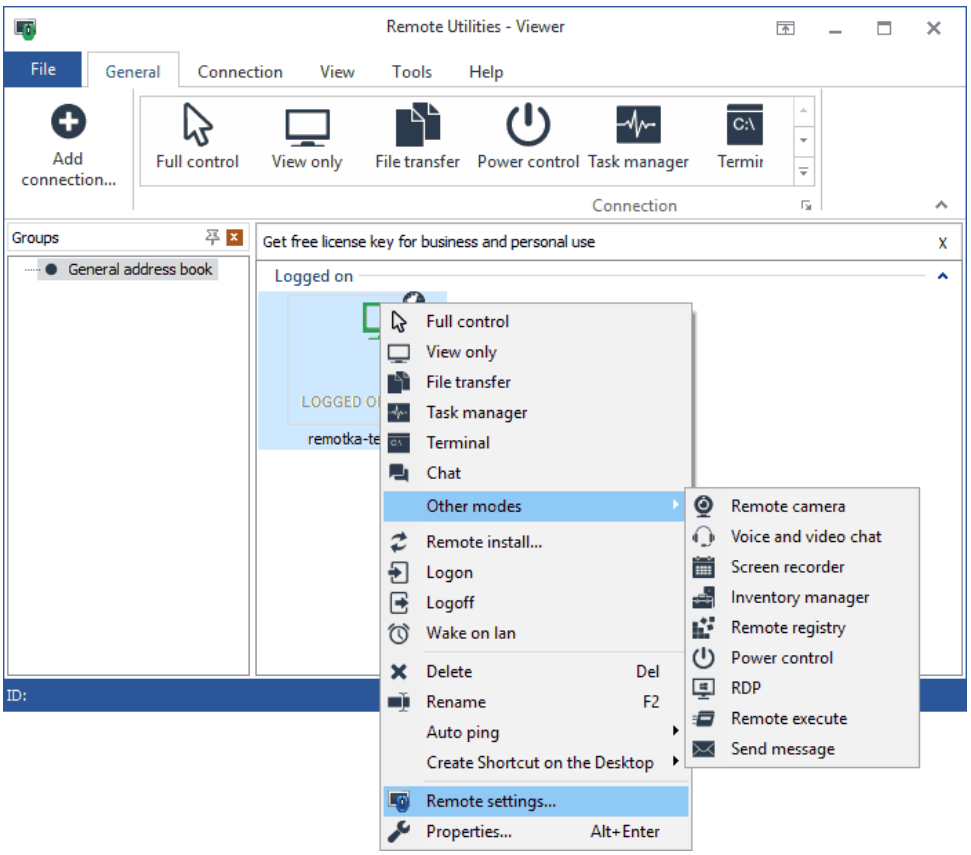
AnyDesk поддерживает русский язык и все необходимые функции, включая передачу файлов, шифрование соединения, возможность работы без установки на компьютер. Впрочем, функций несколько меньше, чем в некоторых других решениях удаленного администрирования, но именно для использования подключения к удаленному рабочему столу «для работы» тут есть всё. Имеются версии AnyDesk для Windows и для всех популярных дистрибутивов Linux. Версии для Mac OS X, iOS и Android обещаются (уже давно).

Из интересных особенностей — работа с несколькими удаленными рабочими столами на отдельных вкладках.



Удаленный доступ RMS или Remote Utilities

Remote Utilities, представленная на российском рынке как Удаленный доступ RMS (на русском языке) — одна из самых мощных программ для удаленного доступа к компьютеру. При этом бесплатна для управления до 10 компьютеров даже для коммерческих целей.



Список функций включает все то, что может понадобиться, а может и не потребоваться, включая, но не ограничиваясь:

- Несколько режимов подключения, включая поддержку подключения RDP через интернет.
- Удаленная установка и развертывание ПО.
- Доступ к видеокамере, удаленному реестру и командной строке, поддержка Wake-On-Lan, функции чата (видео, аудио, текстового), запись удаленного экрана.
- Поддержка Drag-n-Drop для передачи файлов.
- Поддержка нескольких мониторов.

Это далеко не все возможности RMS (Remote Utilities), если Вам требуется что-то действительно функциональное для удаленного администрирования компьютеров и бесплатно, то стоит попробовать этот вариант.

Типы серверного оборудования и их классификация

Что такое сервер

В отличие от персонального компьютера, сервер - это специализированное оборудование, предназначенное для выполнения сервисных задач без участия человека. Представив интернет как огромный организм, сервер будет являться главным органом всей его структуры. Сервер - это вычислительное устройство многопользовательского формата, обеспечивающее выполнение какой-либо задачи (программы) или ряда программ для некоторого количества персональных компьютеров, которых в парке более одного. Следовательно, от сервера зависит работа пользователей, чьи персональные компьютеры к нему подключены.
В зависимости от задач и условий использования, сервер имеет следующие основные свойства:

- Производительность
- Надежность
- Масштабируемость
- Управляемость

Производительность - это количественная характеристика скорости выполнения определённых операций на компьютере или сервере.

Надежность/обеспечение бесперебойной работы. Серверам бесперебойность не обязательна, но она нужна сервисам, которые предоставляют эти серверы. Надежность, в данном случае, достигается путем резервирования и возможности горячей замены компонентов.

Масштабируемость - способность системы, позволяющая справляться с увеличением рабочей нагрузки (увеличивать свою производительность) при добавлении аппаратных(программных) ресурсов.

Управляемость - возможность удаленного мониторинга, удаленного включения и перезагрузки, а также способность проводить диагностику при выключенном состоянии (при условии наличия электропитания).

Типы серверов и их предназначение

- Серверы типа **Tower**, внешне похожи на системный блок персонального компьютера, но имеющие большую вычислительную мощность. Материнская плата таких серверов поддерживает серверные процессоры и буферизированную оперативную память с ECC, что положительно сказывается на надежности сервера. Модуль ECC предназначен для коррекции ошибок, который, в момент некорректной записи информации в память, защищает сервер от перегрузки.
Данный тип серверов подойдет для маленьких компаний из-за малой масштабируемости аппаратной части, но к достоинствам можно отнести более тихую работу и легкость в администрировании. Также, данный сервер не нуждается в специализированной телекоммуникационной стойке, что немаловажно, при отсутствии серверной.
Главная задача данного сервера - это сервис удаленного доступа для небольшого количества пользователей.
Также, есть возможность установки дополнительных контроллеров, типа RAID, для подключения систем хранения данных для организации файлового хранилища или FTP-сервера.

- Серверы типа **Rack**, предназначены для установки в телекоммуникационную стойку, откуда и получили своё название. Данный тип серверов предназначен для того же сервиса удаленного доступа, однако, уже в более крупных масштабах. Это, также, могут быть доменные серверы, DHCP, и другие. Такие серверы имеют больше слотов под установку оперативной памяти, более одного процессора, больше слотов под модули расширения. Ввиду большой нагрузки, данные серверы оборудованы

турбинной системой охлаждения, что сопутствует высокому уровню шума. Серверы можно объединять в кластеры для обеспечения большей вычислительной мощности, развернуть виртуализацию для единой системы или разбить один сервер на несколько для выполнения различных задач.

- **Модульные серверы** или **Blade-серверы**, имеют те же характеристики, что и серверы типа Rack, однако, из-за особенности компактной установки и объединения серверов в одну рабочую машину, данный тип серверов используется для организации вычислительной мощности с целью выполнения сложных вычислений или специализированных операций, а также организации баз данных на крупных предприятиях. Как правило, управление данными серверами организуется через единую систему, что обеспечивает удобство в администрировании. С помощью инструментов виртуализации можно добиться хорошей отказоустойчивости за счет резервирования целых групп серверов подобно RAID-массивам.

Совместимость оборудования

Сервер включает в себя совокупность множества компонентов, которые можно заменять на более новые для достижения лучшей производительности или модификации его под определенные задачи. Однако, не все компоненты взаимозаменяемы и далеко не все совместимы между собой. Производителей на рынке много, каждый имеет как преимущества, так и недостатки. Большую роль играет программное обеспечение компонентов, установленных в сервер. Вместе с этим, существуют и универсальные устройства, ровно как и предназначенные для определенных моделей определенных вендоров.
Так, например, компоненты серверного оборудования HP предназначены только для серверов HP, и с вендором DELL обстоит такая же ситуация. По большей части, процессоры и оперативная память ограничиваются только свойствами материнской платы, и в их совместимости нет трудностей. Разве

что возникают трудности с поколениями данных устройств. Процессоры здесь ограничены сокетом, а память - стандартами скорости передачи данных: DDR2, DDR3, DDR4.
Но, к примеру, с сетевыми картами, RAID-контроллерами и т.д ситуация гораздо сложнее. Для удобства их подбора и использования существуют определенные инструменты - в виде технической документации и таблиц поддержки различных компонентов.

Инструменты

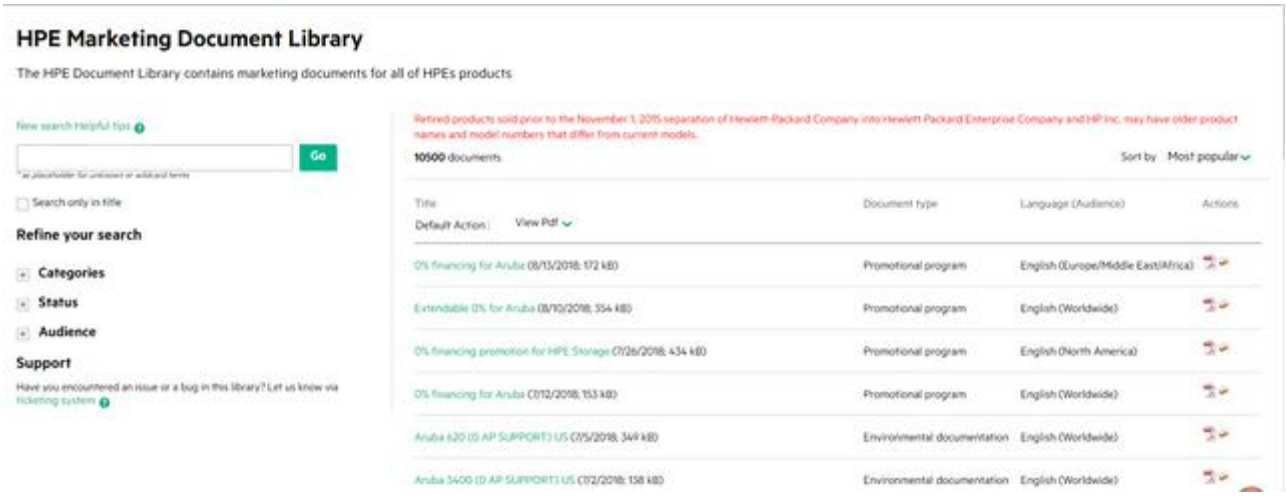
У вендора HP существует определенная база для каждого устройства. Это, пожалуй, самый удобный инструмент в сравнении с другими вендорами, благодаря консолидации данных в едином месте и удобному поиску:

- Сайт-инструмент по нахождению характеристик и совместимости каждого оборудования вендора HP.
- У вендора DELL существует похожая система, но, в отличие от HP, структурированность данных здесь несколько хуже.
- Сайт-инструмент по нахождению характеристик и совместимости каждого оборудования вендора DELL.

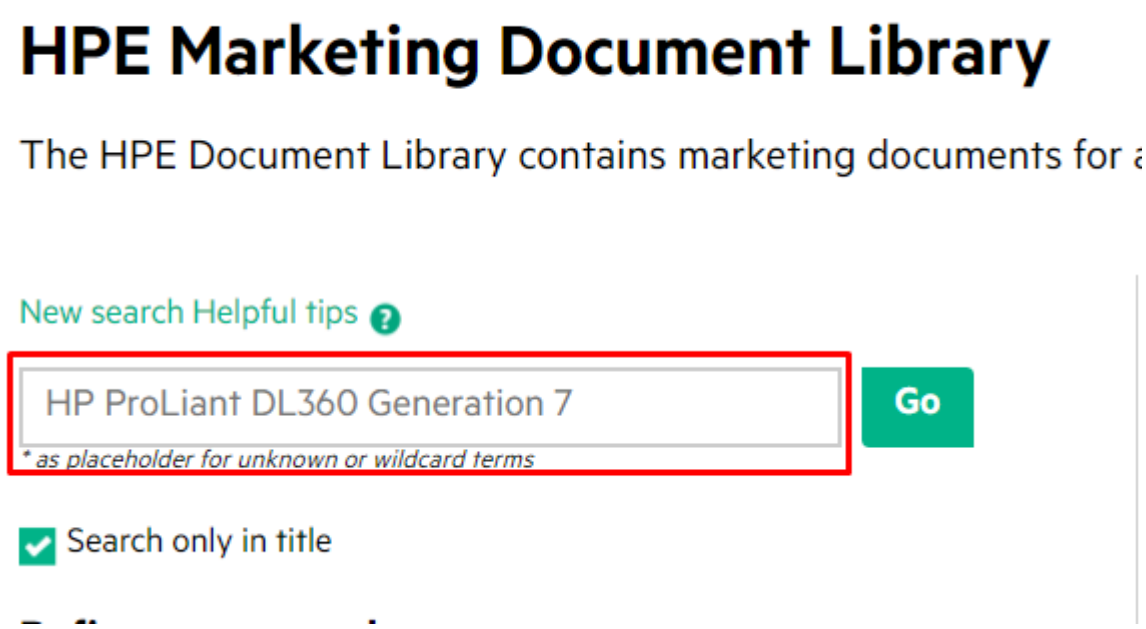
Инструкция по использованию инструментов для поиска совместимости

HP

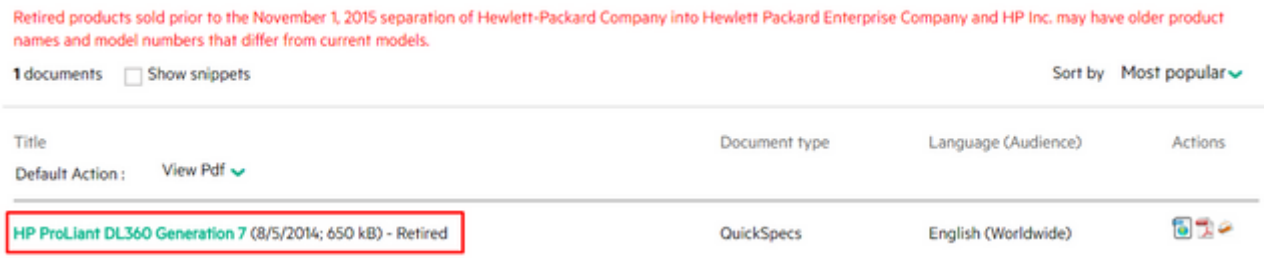
1. Переходим по ссылке, открывается инструмент, представленный на скриншоте ниже:



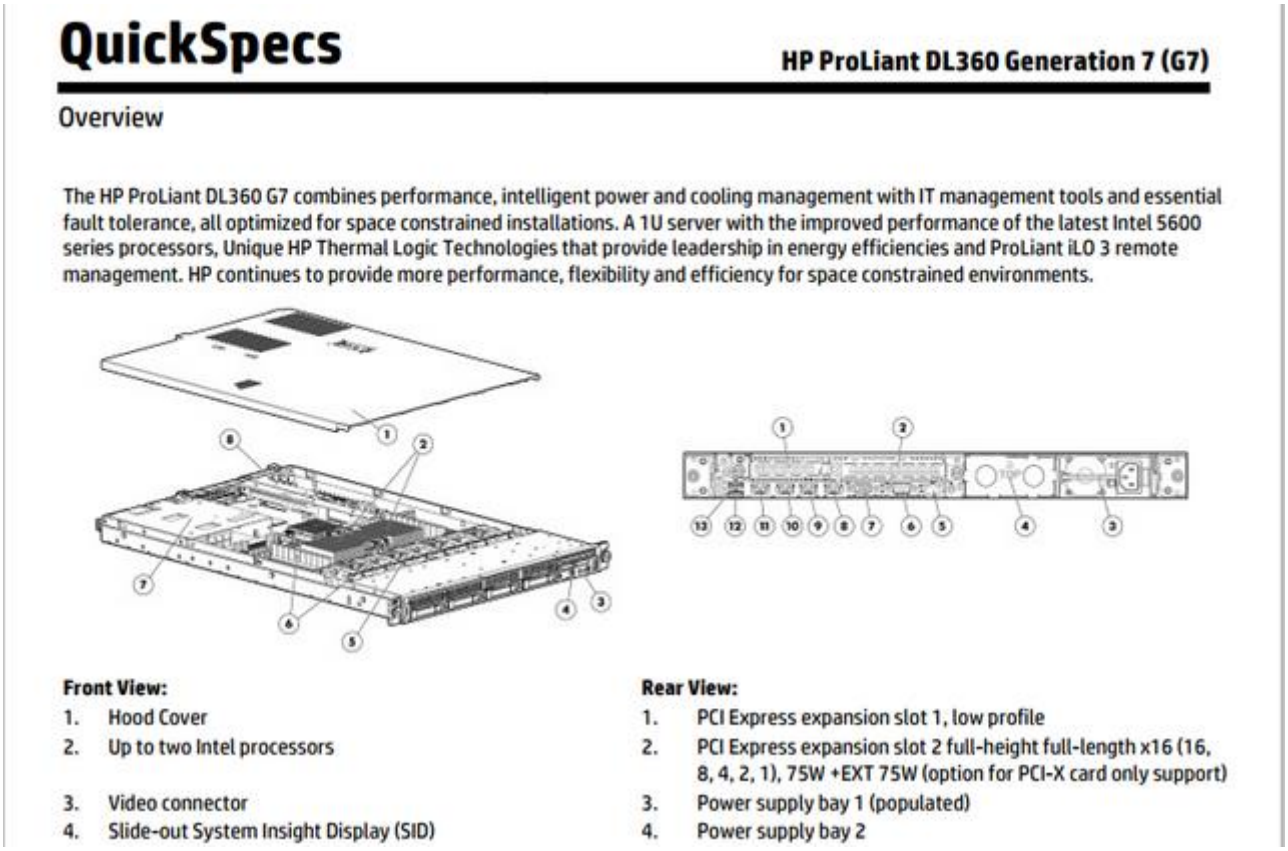
2. Для поиска нужного нам оборудования воспользуемся указанным на скриншоте поиском:



3. Нужный нам объект расположен в обведенной области представленного ниже скриншота:



4. Кликнув на объект откроется подробный документ с описанием, стандартной конфигурацией и всеми подходящими по совместимости компонентами:



QuickSpecs **HP ProLiant DL360 Generation 7 (G7)**

Overview

The HP ProLiant DL360 G7 combines performance, intelligent power and cooling management with IT management tools and essential fault tolerance, all optimized for space constrained installations. A 1U server with the improved performance of the latest Intel 5600 series processors, Unique HP Thermal Logic Technologies that provide leadership in energy efficiencies and ProLiant iLO 3 remote management. HP continues to provide more performance, flexibility and efficiency for space constrained environments.

Front View:

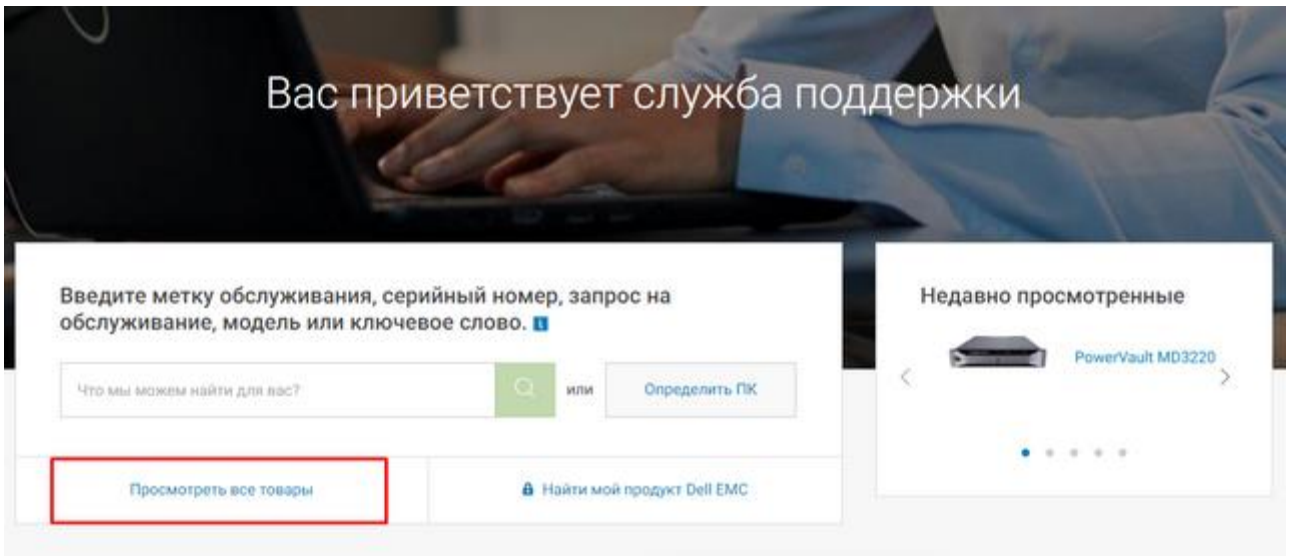
1. Hood Cover
2. Up to two Intel processors
3. Video connector
4. Slide-out System Insight Display (SID)

Rear View:

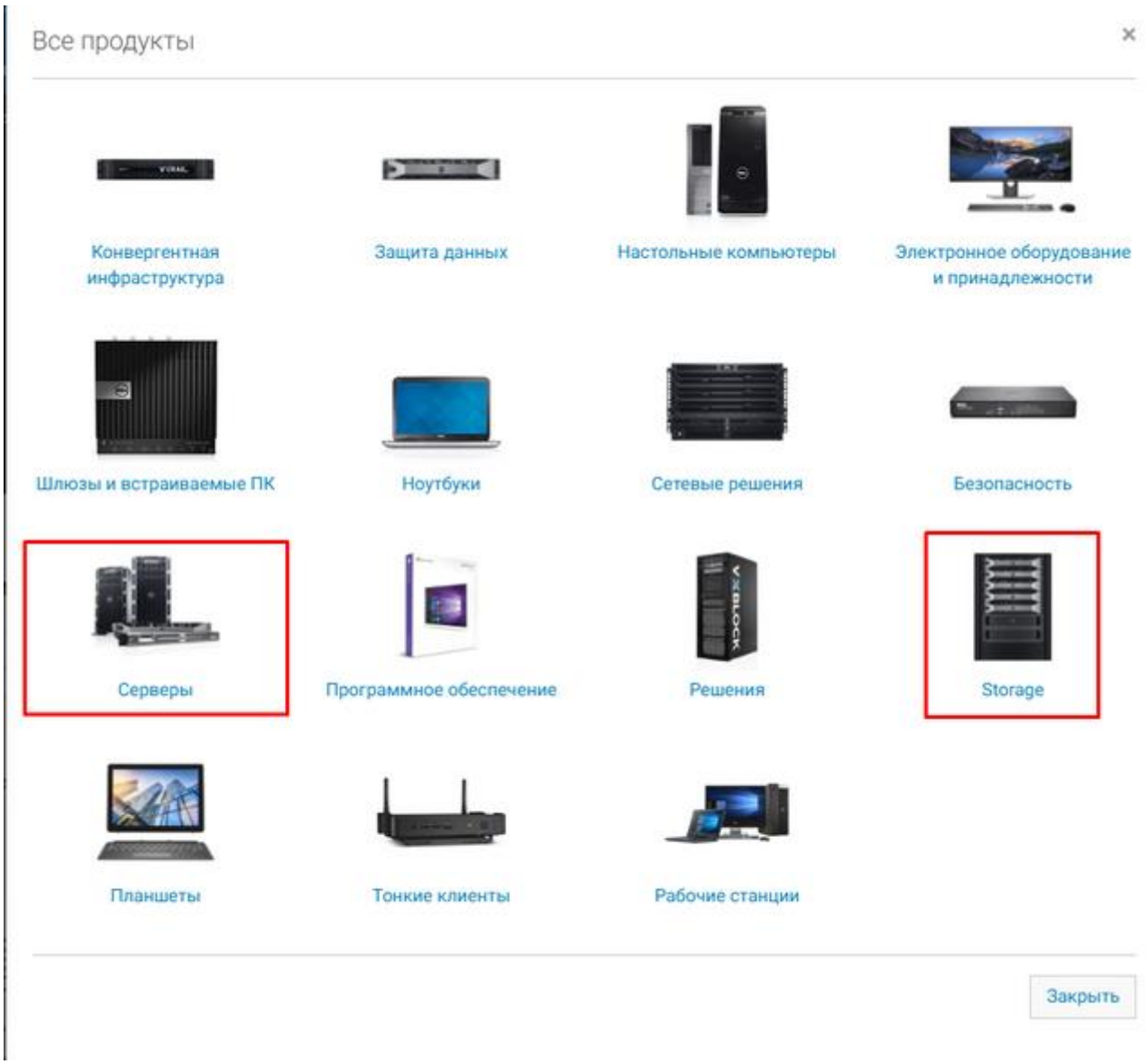
1. PCI Express expansion slot 1, low profile
2. PCI Express expansion slot 2 full-height full-length x16 (16, 8, 4, 2, 1), 75W +EXT 75W (option for PCI-X card only support)
3. Power supply bay 1 (populated)
4. Power supply bay 2

DELL

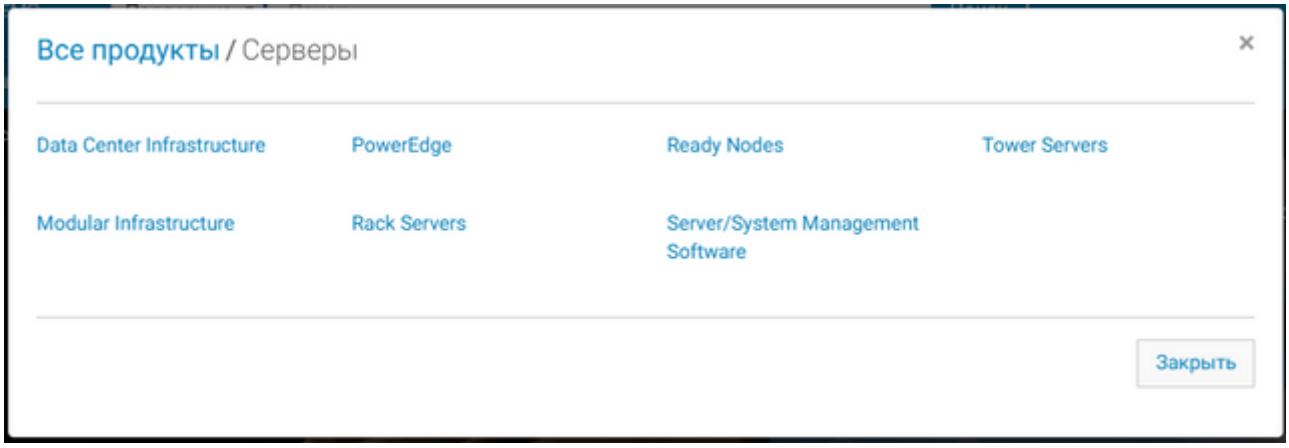
1. Перейдя по ссылке, мы попадаем в инструмент поиска оборудования DELL. На представленном ниже скрине выбираем указанный раздел:



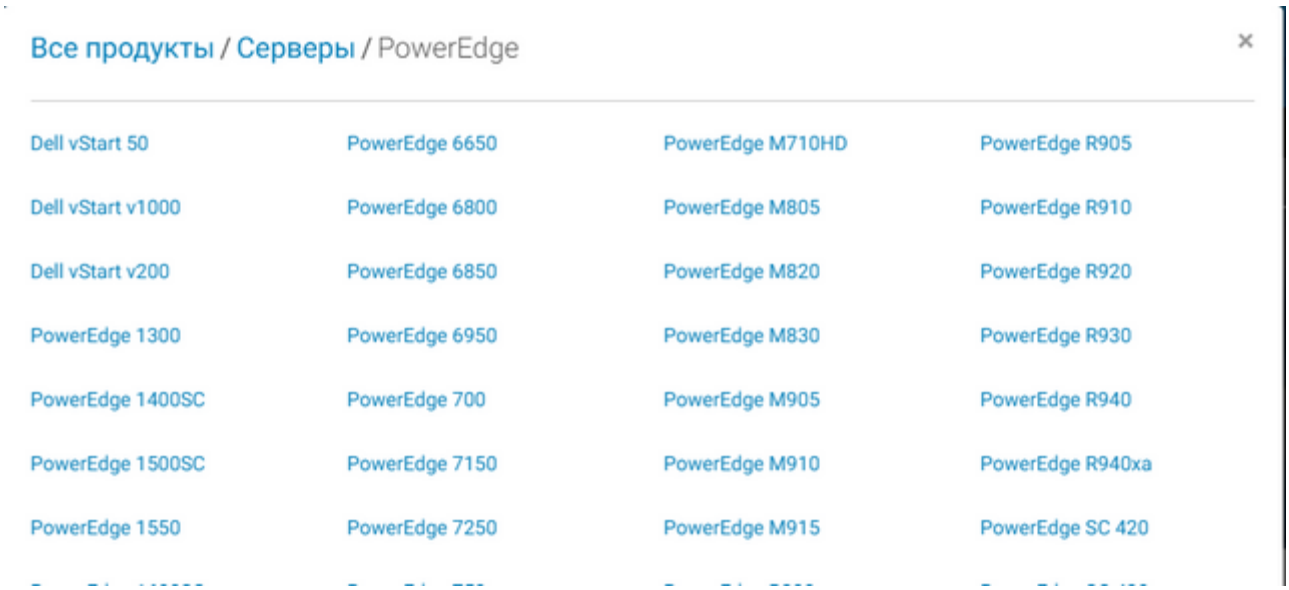
2. Выбираем нужную категорию, в нашем случае - “Серверы и системы хранения данных”



3. Выбираем интересующий класс оборудования



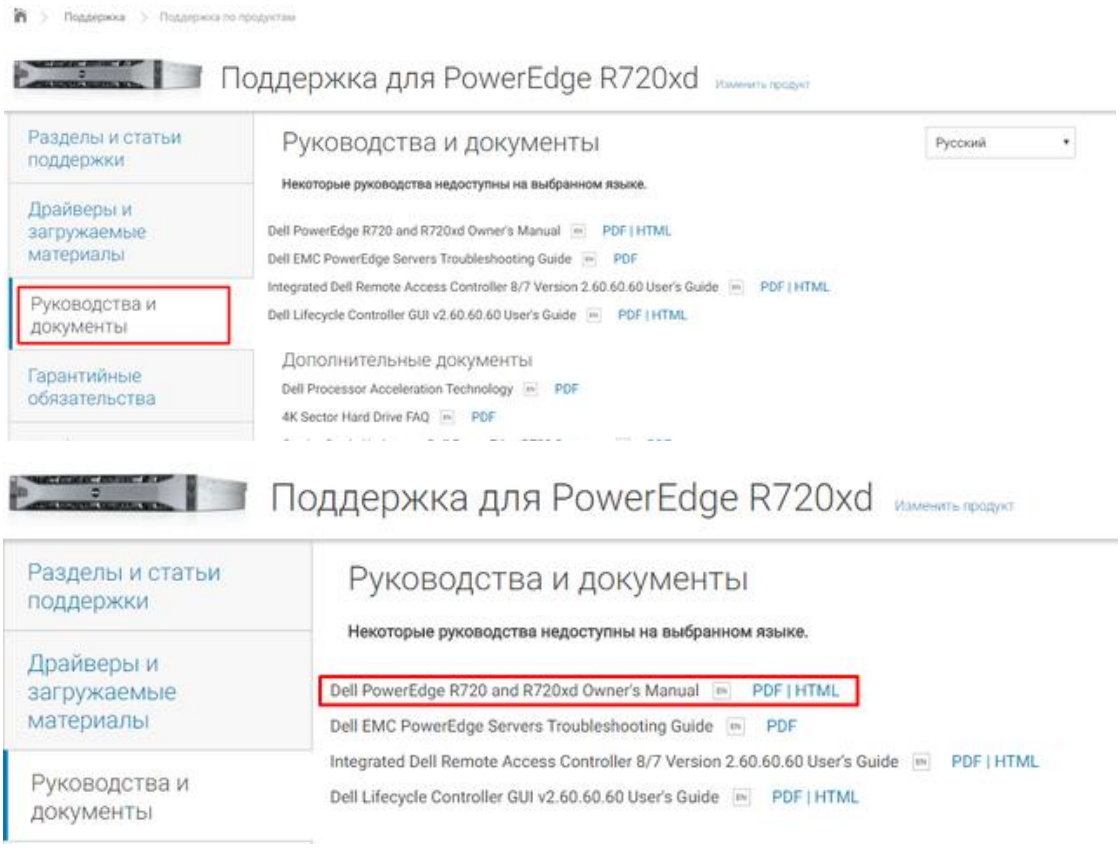
4. Осуществляем поиск модели



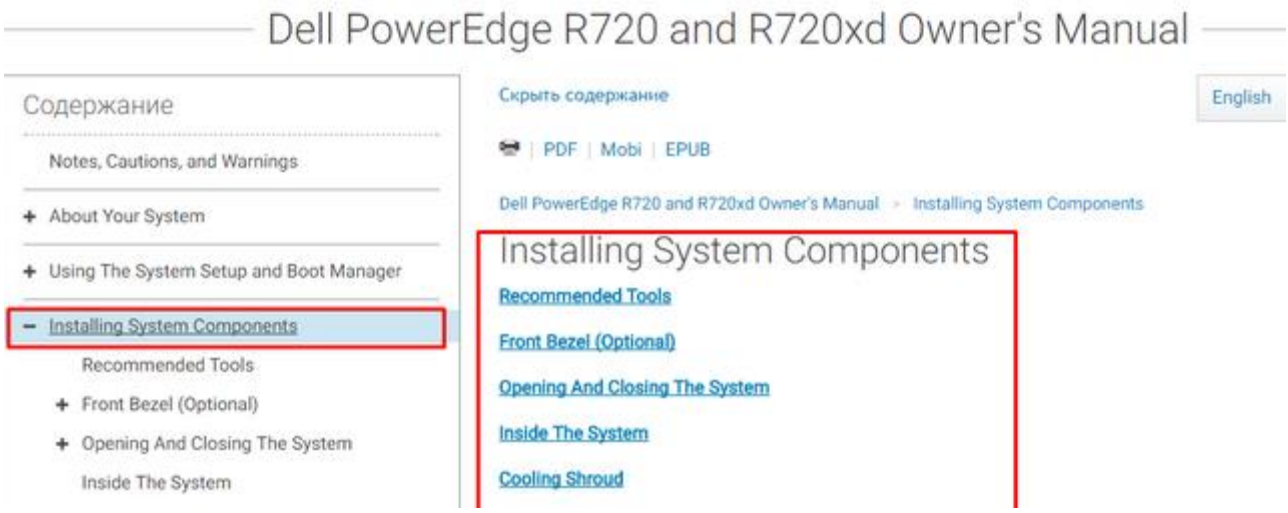
У разных классов устройств документ по характеристикам и совместимости называется разными способами, всего их два:

- Hardware Owner’s Manual
- Series Support Matrix.

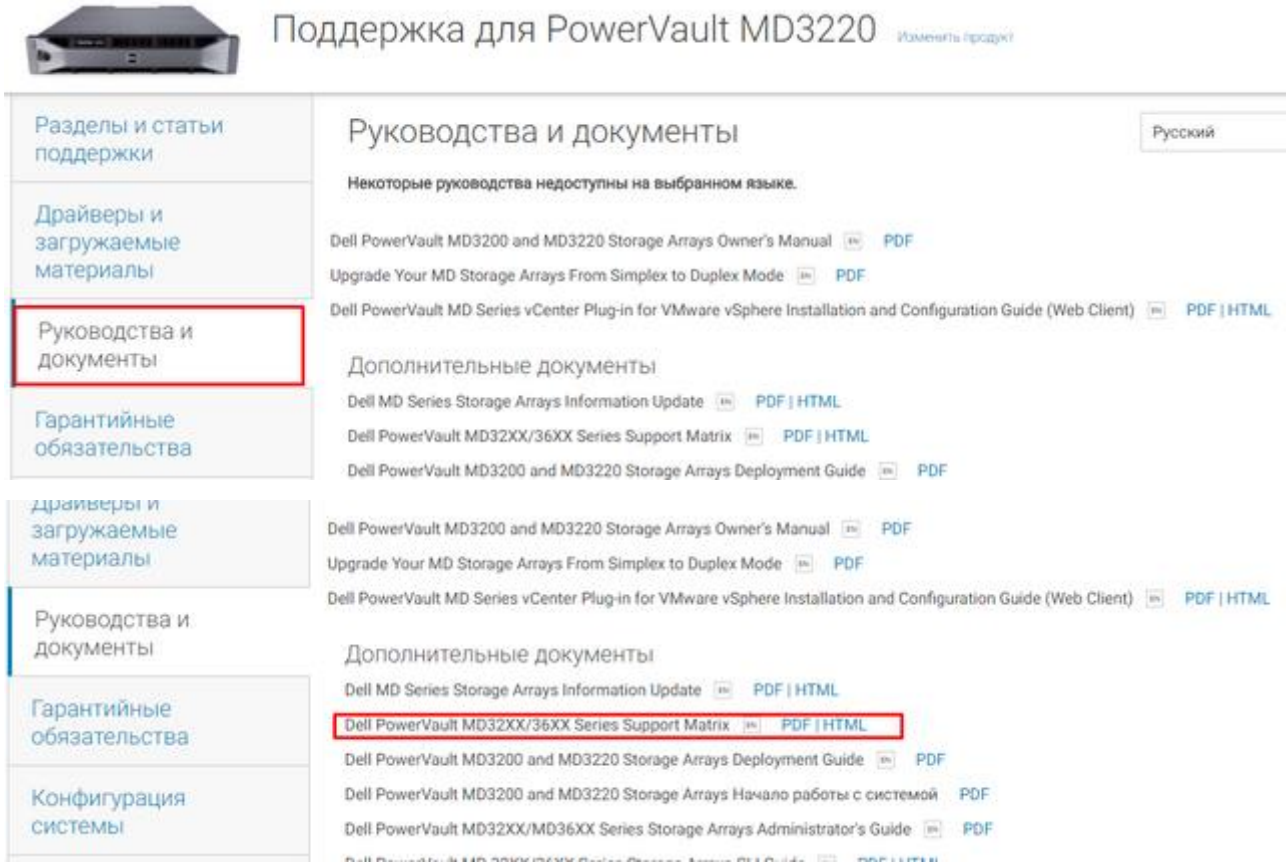
5. Поиск Hardware Owner’s Manual



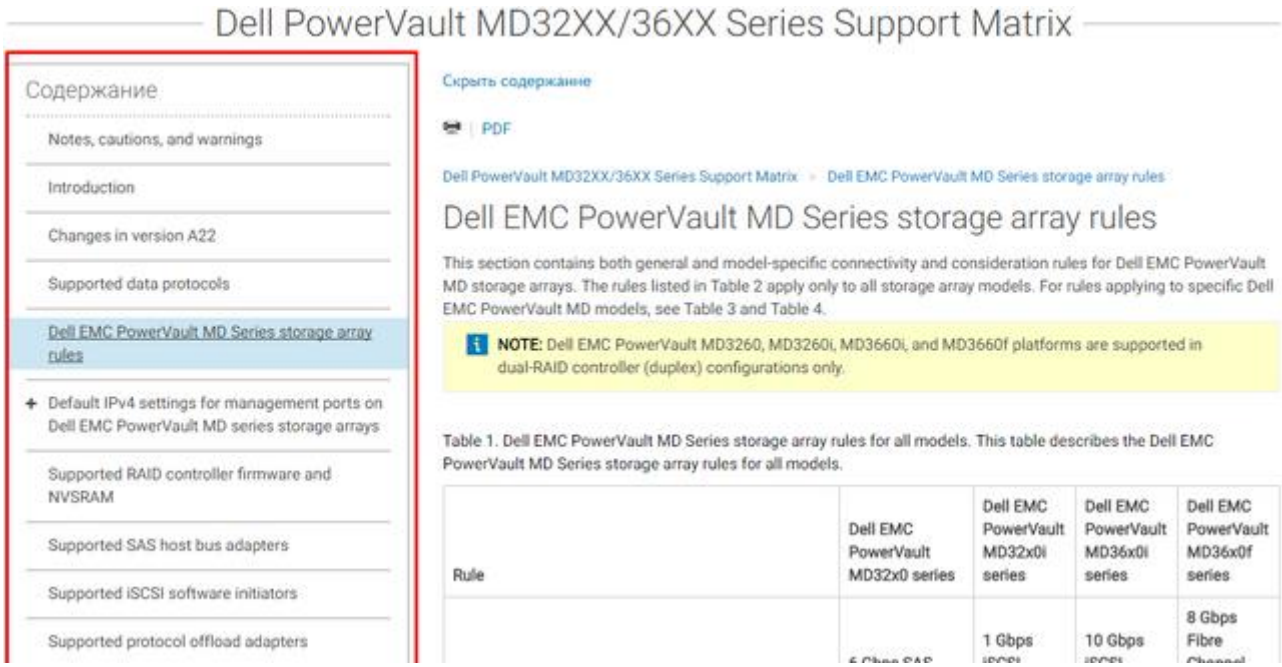
6. Рабочая область по поиску совместимости и характеристикам интересующего оборудования по Hardware Owner’s Manual



7. Поиск Series Support Matrix



8. Рабочая область по поиску совместимости и характеристикам интересующего оборудования по Series Support Matrix



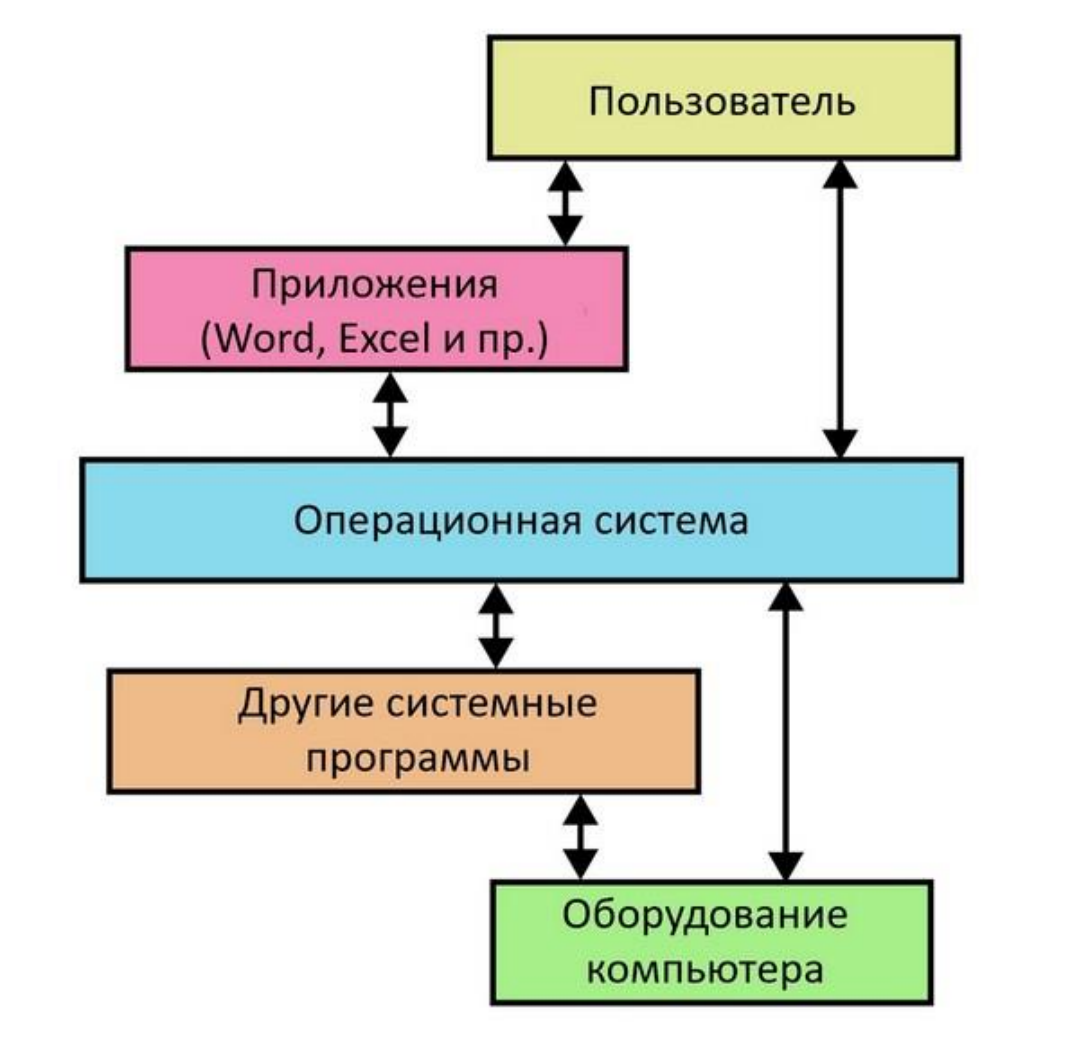
Заключение

Для выбора сервера нужно определиться с целью его использования. Задав цель и воспользовавшись данной статьей, можно выбрать класс устройства, подходящий для ваших задач. Подобрать совместимые компоненты помогут официальные мануалы или документы о совместимости, которые можно найти на сайте производителя, выбрав модель, которая вам нужна.

Серверные операционные системы

Операционная система ОС (Operating system, OS) – это комплекс программ, который выполняет роль интерфейса (панели взаимодействия) между пользователем и оборудованием компьютера. Чтобы компьютер мог работать, на нем должна быть установлена хотя бы одна ОС. Все приложения компьютера, такие как текстовые и графические редакторы, электронные таблицы, базы данных, интернет-браузеры и пр., и пр., не могут работать и

выполнять свои задачи без программной среды операционной системы, которая предоставляет для них необходимые сервисы.



Схематичное изображение функций ОС

Важно понимать отличие серверной операционной системы от операционной системы обычного компьютера.

В обычной ОС работают такие программы как MS Word, Excel, PowerPoint, Visio, Adobe Photoshop и многие другие, которые используются для повседневной работы, а также игры и прочие развлекательные приложения для отдыха. Обычная ОС отвечает за подключение пользователя компьютера к локальной сети LAN и к сети Интернет, а также к различным устройствам через протокол Bluetooth. Стоит добавить, что обычная ОС стоит гораздо меньше, чем ОС сервера.

Серверная ОС использует гораздо больший объем памяти для вычислений, а также может выполнять функции веб-сервера, сервера приложений и сервера электронной почты и многих других серверов, необходимых для работы ИТ-системы предприятия. Серверная ОС может подключать к локальной сети и к Интернет многих пользователей, а не одного, как обычная ОС. Поэтому серверная ОС и более дорогая.

Обзор наиболее популярных серверных ОС

Компания Microsoft предлагает ОС Windows Server – серверную операционную систему корпоративного класса с широкими возможностями управления хранением данных, приложениями и сетями.

Компания Apple также имеет ОС для серверов macOS Server, содержащую возможности программирования сервера, а также управления и администрирования пользователей ОС macOS для персональных компьютеров.

Кроме того, альтернативами являются ОС на базе Linux: Red Hat Enterprise Linux, Ubuntu Server и CentOS. Существуют также серверные ОС UNIX. Кратко рассмотрим основные из них.

Microsoft Windows Server

Windows Server является частью семейства сетевого программного обеспечения Windows Network, которое разрабатывалось совместно с ОС Windows 10.

Последняя версия ОС Windows Server 2019 может работать как на серверах предприятия, так и на арендованных серверах в облаке (Microsoft Azure), создавая гибридные вычислительные среды. Облачные ресурсы могут задействоваться в случае необходимости при возрастании нагрузки на собственную ИТ-систему и использоваться по модели оплаты по мере использования (pay-as-you-go). Windows Server 2019 также имеет расширенные функции безопасности.

Новые функции в Windows Server 2019:

- **Поддержка гибридного облака**. ИТ-систему предприятия на Windows Server можно расширить в облачную среду Azure и получить там дополнительные функции и сервисы, а также увеличить емкость хранения и вычислительную мощность.
- **Безопасность**. Возможность защиты от атак злонамеренных сторонних программ и предотвращения несанкционированного проникновения в виртуальные машины.
- **Разработка приложений**. Поддержка технологий контейнеров Kubernetes для разработки новых сервисов и приложений, а также новые возможности развертывания и масштабирования приложений в гибридном облаке на базе Azure.
- **Поддержка гиперконвергентной инфраструктуры HCI**. Windows Server 2019 облегчает развертывание HCI (Hyper Converged Infrastructure), и таким образом значительно снижает расходы на развертывание ИТ-системы предприятия.
- **Поддержка Linux**. Windows Server 2019 содержит усовершенствованную версию подсистемы для поддержки Windows

Subsystem for Linux (WSL). Поэтому разработчики на базе Windows Server 2019 имеют возможность разрабатывать приложения для ОС Linux непосредственно в среде Windows, в которой могут работать виртуальные машины Linux. Кроме того, разработчики могут писать программы на популярном языке команд Bash, а также Ruby и Python.

- **Поддержка системы управления контейнерами Kubernetes**. Контейнерные технологии приобретают все большую популярность, поскольку они позволяют вместо виртуальных машин, которым нужна нижележащая ОС, запускать контейнеры, в которых сервисы и приложения работают на ОС, которая встроена непосредственно в контейнер. Эти контейнеры можно сохранять в виде образов, и при необходимости повторно использовать при разработке приложений вместо того, чтобы писать код заново. Платформы оркестрации контейнеров, такие как Kubernetes, автоматизируют создание, развертывание и управление контейнерами, а также их масштабирование и другие текущие задачи. Таким образом, создается база цифровой трансформации предприятия на основе модели DevOps, когда разработчики (developers) работают параллельно с операционным службами (operation). В Windows Server 2019 имеется встроенная поддержка Kubernetes с улучшенными функциями, по сравнению с предыдущей версией Windows Server 2016.

В Windows Server 2019 имеется также центр администрирования серверов WAC (Windows Admin Center). Он устанавливается в ИТ-системе заказчика и позволяет администрировать локальные и облачные экземпляры Windows Server 2019, компьютеры под управлением ОС Windows 10, кластеры и гиперконвергентную инфраструктуру. WAC также может администрировать серверы за пределами ИТ-системы организации за счет средств повышенной безопасности и мобильным решениям Enterprise Mobility + Security (EMS), которые позволяют предоставлять или отказывать в доступе в зависимости от

соответствия устройства политикам, рискам, местоположению и другим факторам.

Windows Server 2019 обладает новыми интеллектуальными возможностями, в частности, System Insights, для прогнозной аналитики, которые позволяют предотвращать проблемы в парке серверов предприятия до их возникновения. Модель машинного обучения учитывает нагрузку и события в системе, а также может спрогнозировать недостаток свободного места в системах хранения данных. Кроме того, машинное обучение предоставляет аналитические сведения о работе серверов и помогает сократить эксплуатационные затраты.

Red Hat Enterprise Linux

Red Hat Enterprise Linux (RHEL) – довольно популярный дистрибутив распределенной серверной ОС, разработанной на базе ОС Linux компанией Red Hat, первая версия которого была выпущена на рынок в 2003 году.

Хотя RHEL относится к классу «открытого ПО» и его исходный программный код бесплатен и доступен всем желающим, однако двоичный (исполняемый компьютером) код RHEL приобретается за плату. Раз в два года Red Hat выпускает версии RHEL в двоичном коде с поддержкой в течение десяти лет, причем Red Hat отслеживает критические исправления Linux и обновляет уже выпущенные версии серверной ОС. Кроме того, Red Hat является крупнейшим вкладчиком проекта ОС Linux по объему программного кода.

Последняя версия RHEL 8.2 была представлена 24 апреля 2020 года. Она ориентирована для работы в гибридном облаке и располагает эффективными инструментами для обеспечения надежности, стабильности и доступности.

Она содержит следующие обновления:

- Система умного мониторинга и управления в рамках сервиса Red Hat Insights.

- Расширенный набор инструментов для работы с контейнерами.
- Улучшенный интерфейс, подходящий как для новичков, так и экспертов.

29 июля 2020 года компания Red Hat открыла общий доступ к «бета-версии» Red Hat Enterprise Linux 8.3 beta. В корпоративных ИТ-системах могут работать сотни и тысячи серверов, каждый из которых требует почти ежедневного обслуживания и управления. Нерегулярное выполнение этих операций может вызвать простои и уязвимости безопасности. Для решения этой проблемы в RHEL 8.3 beta включен сервис System Roles – уже настроенные и входящие в состав ОС сценарии выполнения типовых процессов администрирования.

В RHEL 8.3 beta также появились дополнительные роли, в частности System Role for System Logging, System Role for System Metrics и ряд других. Каждая из этих ролей предоставляет повторяемые наборы операций для всех поддерживаемых версий RHEL, что позволяет автоматизировать работу системного администратора.

RHEL 8.3 beta поддерживает протокол автоматизации управления данными безопасности SCAP, с помощью которого администратор может сконфигурировать систему в строгом соответствии требованиям безопасности коммерческого сектора или сферы здравоохранения. RHEL 8.3 beta также получила роль System Role for Network-Bound Disk Encryption (NBDE), обеспечивающую согласованность и повторяемость при настройке шифрования дисков и ведении журналов.

Многие другие дистрибутивы на базе Linux, такие как CentOS, Scientific Linux и Oracle Linux, построены на исходных кодах RHEL.

macOS Server

macOS Server (ранее Mac OS X Server, OS X Server) – это серверная операционная система компании Apple. Начиная с версии OS X 10.7 в 2011

году, серверная редакция была встроена в обычную версию OS X. Однако для управления сервером было необходимо приобрести приложение OS X Server.

macOS Server включает программы администрирования рабочих групп, которые обеспечивают упрощенный доступ к сетевым сервисам: почтовому серверу, серверу DNS и другим. Также включает в себя многочисленные дополнительные сервисы и программы управления, например, веб-сервер, вики-сервер, чат-сервер, календарь-сервер и другие. OS X Server поставлялась до 2014 года с компьютерами Mac mini Server и Mac Pro Server. В настоящее время macOS Server распространяется через App Store для использования на компьютерах компании Apple.

В 2018 году компания Apple прекратила поддержку многих сервисов в macOS Server: DHCP, DNS, почты, сообщений, NetInstall, VPN, Web server, Wiki, а также календаря и контактов. Для тех, кто продолжает работать с macOS Server, Apple привела список доступных альтернатив с открытым кодом.

Ubuntu Server

Ubuntu OS – популярная ОС с открытым кодом на базе Linux для компьютеров, ноутбуков, планшетов, телефонов и виртуальных машин, а также для серверов. Основным разработчиком и спонсором является компания Canonical, однако эта ОС развивается и поддерживается свободным сообществом разработчиков. Ubuntu распространяется абсолютно бесплатно и содержит полный набор всех необходимых для работы приложений, а всё недостающее в стандартной поставке вы можно скачать из Интернета. Пользовательская документация по операционной системе Ubuntu Linux создается сообществом forum.ubuntu.ru.

По утверждениям Canonical, Ubuntu используется примерно 20 миллионами пользователей и занимает 1-е место в списке самых популярных дистрибутивов Linux для веб-серверов.

Ubuntu поставляется с подборкой программного обеспечения для разных типов серверов и рабочих станций, а также имеет версии для разных процессорных архитектур: x86, AMD, ARM. Кроме того, с 2013 года начата разработка специальной версии Ubuntu для смартфонов на архитектуре ARM и x86. Существует также редакция Ubuntu Core, предназначенная для устройств интернета вещей (IoT) и роботов.

CentOS Server

CentOS – это еще один дистрибутив Linux, представляющий собой свободную ОС корпоративного класса, поддерживаемую сообществом разработчиков.

Разработка CentOS началась в 2006 году под наименованием Tao Linux (клона Red Hat Enterprise Linux), затем было принято название CentOS. В январе 2014 года компания Red Hat объявила, что она будет спонсировать этот проект. В результате владение товарными знаками CentOS было передано Red Hat; при этом большинство разработчиков CentOS работают в обособленном подразделении компании Red Hat параллельно с командой разработчиков Red Hat Enterprise Linux.

CentOS функционально совместима с Red Hat Enterprise Linux и построена из программных блоков RHEL. Согласно жизненному циклу Red Hat Enterprise Linux, CentOS версий 5, 6 и 7 будет поддерживаться до 10 лет.

На базе CentOS были созданы следующие программные продукты:

- Scientific Linux — дистрибутив, развиваемый компанией FermiLab для научных работников.
- SME Server для малых и средних предприятий;
- Boston University’s Linux 4.5 Server Edition (Zodiac) – версия Linux для сотрудников и студентов университета Бостона (США);
- Elastix — дистрибутив для организации сервера коммуникаций, основанный на CentOS 7;

- ClearOS —маршрутизатор для дома и небольших организаций, предоставляется на основе ежемесячной подписки;
- AstraLinux — дистрибутив, ориентированный на использование в государственных учреждениях России.
- Янукс — российский дистрибутив, ориентированный на использование в информационных системах с повышенными требованиями к безопасности обрабатываемых данных.

SUSE Enterprise Linux Server

SUSE Linux Enterprise Server (SLES) — еще один дистрибутив Linux, созданный германской компанией SUSE (Software und System Entwicklung), с 2003 года принадлежащей американской корпорации Novell. Новые версии SLES выходят каждые 2–3 года. Предоставляется коммерческая поддержка на время жизни версии – 10 лет.

С 2006-го года SUSE и Microsoft работают над улучшением взаимодействия между Linux и Windows. В настоящее время SLES показывает самую лучшую производительность среди других дистрибутивов Linux на гипервизоре Microsoft Hyper-V. SLES 11 в настоящее время поддерживается в облачной платформе Microsoft Azure и на виртуальных серверах Microsoft Hyper-V.

В 2010-ом году компания VMware приняла SUSE Linux Enterprise в качестве своего внутреннего стандарта. Компании VMware и Microsoft предлагают поддержку для виртуальных серверов, работающих на SLES.

Партнером SUSE, активно поддерживающим SLES, является также компания SAP. Более 70 % клиентов SAP, работающих на Linux, используют SLES.

Кроме того, SLES используется в суперкомпьютерных системах, например Titan, IBM Watson, операционной среде CLE суперкомпьютеров

корпорации Cray. С 2018 года существует версия SLES для микрокомпьютеров Raspberry Pi для Интернета Вещей.

Oracle Linux Server

Oracle Linux (Oracle Enterprise Linux, Unbreakable Linux) — открытый дистрибутив ОС Linux, который после регистрации можно свободно скачать с сайта компании Oracle. Oracle также предоставляет услуги по платной технической поддержке организаций, использующих дистрибутив.

Oracle Linux создан в 2006 году на основе Red Hat Enterprise Linux. Oracle Linux на 100 % совместим с Red Hat Enterprise Linux на уровне двоичного кода. Предлагая копию дистрибутива от Red Hat, Oracle установила стоимость услуг по технической поддержке на Oracle Linux ниже, чем у Red Hat.

31 марта 2020 г. было выпущено обновление ядра ОС Unbreakable Enterprise Kernel 6 для Oracle Linux, которое предоставляет новые функции с открытым исходным кодом и критически важные для бизнеса решения по оптимизации производительности и безопасности для корпоративных ИТ-систем как на площадке предприятия, так и в облаке.

ClearOS Server

ClearOS (ранее ClarkConnect) – серверная ОС, выпускаемая компанией ClearCenter и представляющая собой дистрибутив Linux на базе Red Hat Enterprise Linux. Предназначена для малых и средних предприятий, где использование Linux в качестве серверной ОС затруднено сложностью развертывания. Для решения этой проблемы была разработана ClearOS, интерфейс которой значительно упрощает процесс установки и развертывания.

Существует 3 редакции этой ОС – от бесплатной версии с ограниченным функционалом до платной корпоративной версии с профессиональной поддержкой.

Начиная с версии ClearOS 6.1 дистрибутив является полнофункциональной операционной системой для серверов и рабочих станций, собранной из модулей исходного кода Red Hat Enterprise Linux.

ClearOS предоставляет следующие возможности:

- файрвол ИТ-системы с динамической фильтрацией пакетов по заданным правилам;
- система обнаружения и предупреждения вторжений;
- защищенные каналы VPN в с поддержкой IPSEC, PPTP;
- прокси-сервер с фильтром контента и антивирусом;
- сервисы электронной почты (Webmail, Postfix, SMTP, POP3/s, IMAP/s);
- программное обеспечение совместной работы (Kolab);
- сервер баз данных и веб-сервер;
- файловый сервер и принт-сервер;
- Flexshares: технология управления приоритетами в обслуживании томов данных в СХД;
- MultiWAN: технология подключения к нескольким глобальным сетям;
- встроенная система отчетов для ведения статистики системы и услуг (MRTG и др.).

Linux Debian

Linux Debian— еще один дистрибутив Linux, который можно использовать в качестве операционной системы как для серверов, так и для рабочих станций.

Среди всех Linux-дистрибутивов Debian имеет самое большое хранилище готовых к использованию программ и библиотек, в том числе по количеству поддерживаемых архитектур процессоров – начиная с ARM, используемой во встраиваемых устройствах, x86-64 и PowerPC, и заканчивая IBM S/390, используемой в больших компьютерах (мейнфреймах).

FreeBSD

FreeBSD — свободная операционная система семейства UNIX, первая версия которой была разработана в университете Беркли в Калифорнии в 1993 году. FreeBSD разрабатывалась как целостная операционная система. Это означает, что исходный код ее ядра, драйверов устройств и базовых пользовательских программ, таких как командные оболочки и т. п., содержится в одном дереве системы управления версиями. Это отличает FreeBSD от Linux — другой свободной UNIX-подобной операционной системы, в которой ядро разрабатывается одной группой разработчиков, а набор пользовательских программ — другой (например, проект GNU). Затем третьи группы разработчиков комбинируют все это и выпускают в виде различных дистрибутивов Linux.

FreeBSD хорошо зарекомендовала себя как ОС для сетей Интранет и Интернет, а также для серверов.

Помимо своей стабильности, FreeBSD популярна благодаря своей лицензии, которая существенно отличается от известной лицензии GNU GPL, требующей раскрытия исходных кодов. FreeBSD требует лишь упомянуть заимствование, указать авторство и отказаться от навязывания ответственности (нельзя отсылать пользователей своего продукта к авторам заимствованного кода FreeBSD).

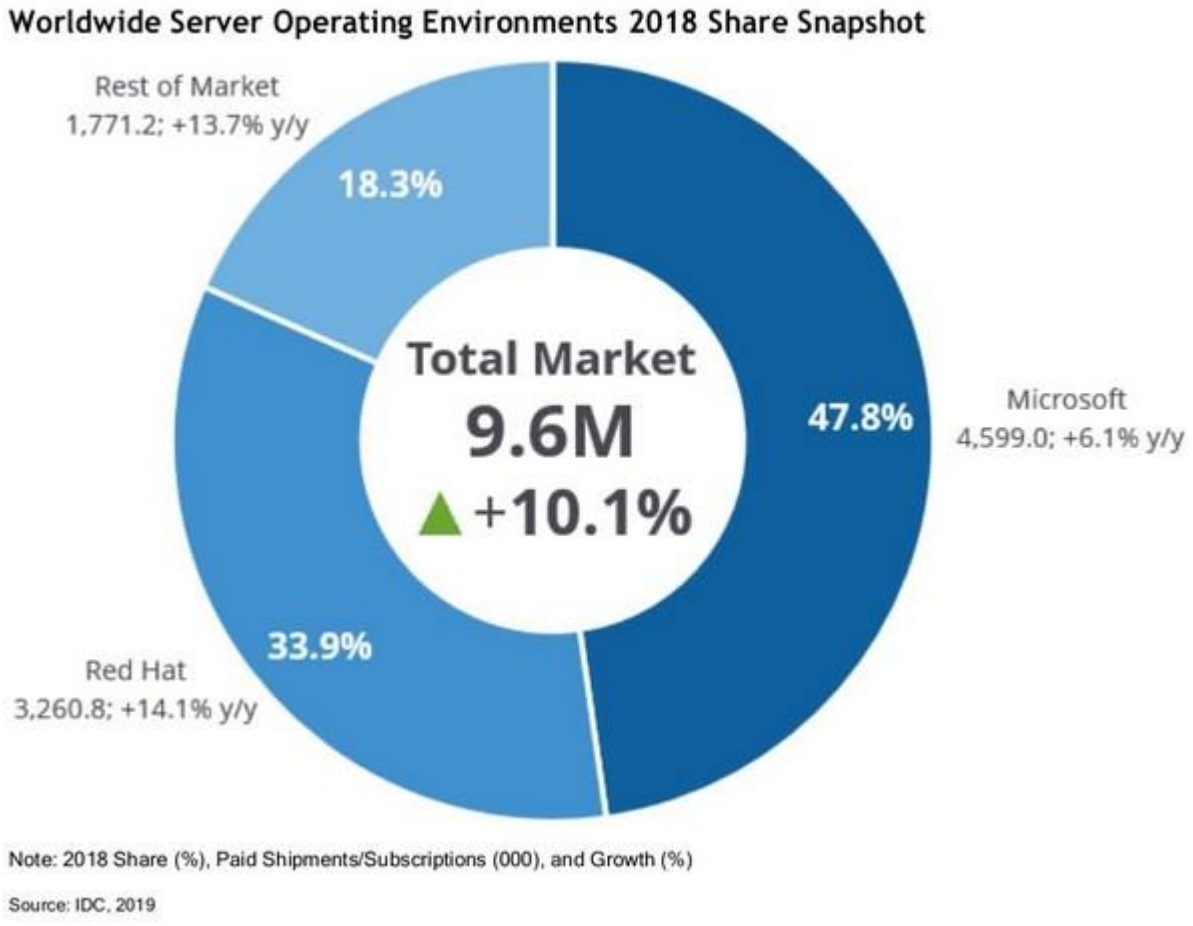
Последней версией является FreeBSD 12, выпущенная 11 декабря 2018 года, в которой, среди прочего, значительно улучшена поддержка виртуализации.

Так какая серверная ОС лучше?

Здесь кратко описаны лишь основные типы серверных ОС, не считая их взаимных клонов, версий и комбинаций. На вопрос «какую серверную ОС выбрать для предприятия» невозможно дать определенного ответа: слишком много факторов будут влиять на выбор. Здесь многое зависит от личных

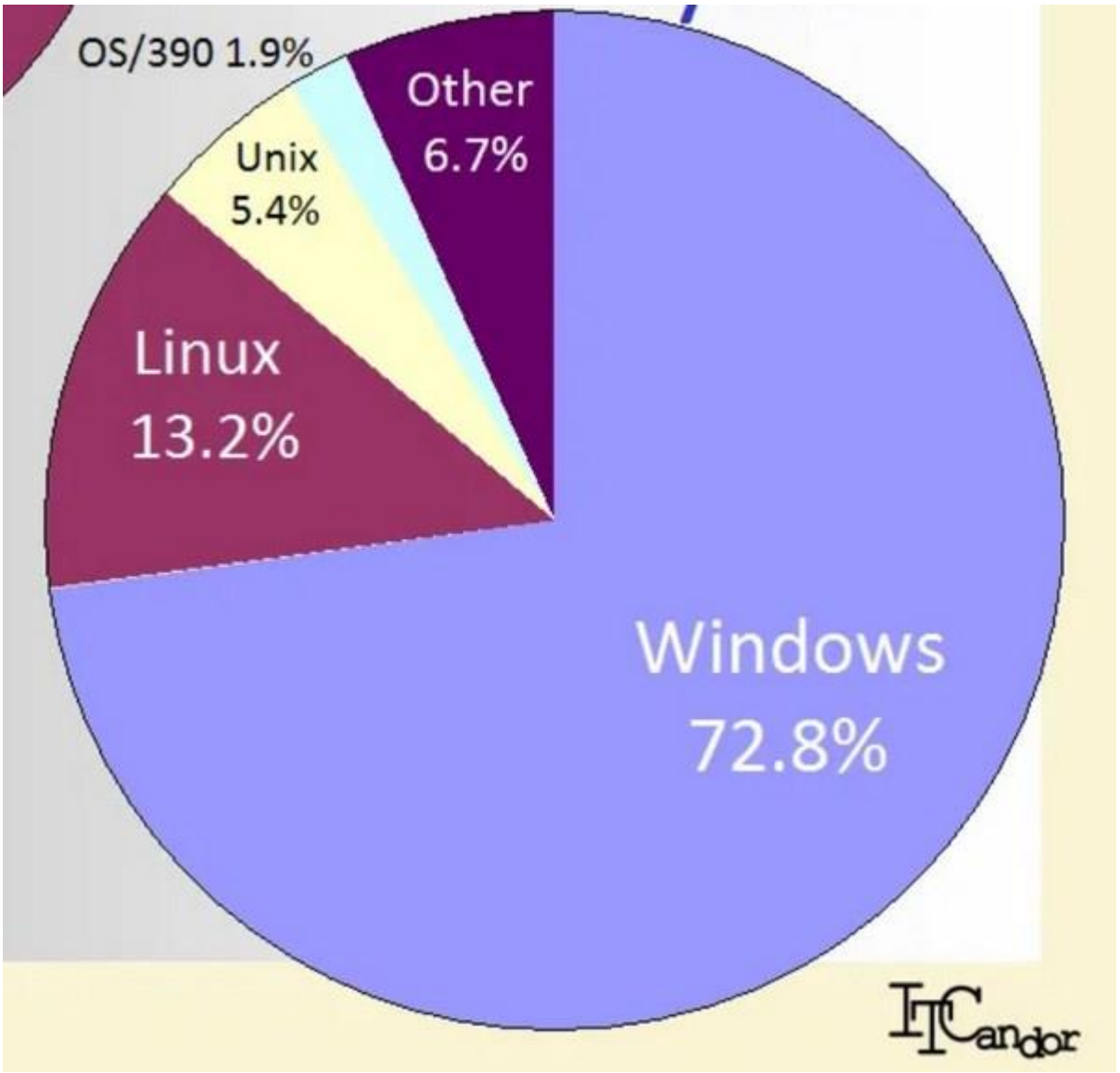
предпочтений как руководителя предприятия, так и его ИТ-директора, а также системного администратора.

Можно лишь привести данные о популярности серверных ОС. По данным компании IDC за 2018 год лидерами в разработке и поставках серверных ОС являются компании Microsoft (47.8%) и Red Hat (33.9%). Остальные компании занимают 18.35% рынка, но этот сектор растет быстрее – темп среднегодового прироста составляет 13.7%. Среднегодовой темп роста рынка серверных ОС, несмотря на кризис, составляет внушительную цифру – более 10% в год!



Что касается числа серверов под той или иной ОС, то компания ITCandor приводит такие данные: в 2019 году 72.8% серверов поставлялись с ОС Windows, 13.2% – с ОС Linux (в виде различных дистрибутивов), 5.4% – с ОС Unix и 1.9% занимают большие компьютеры с

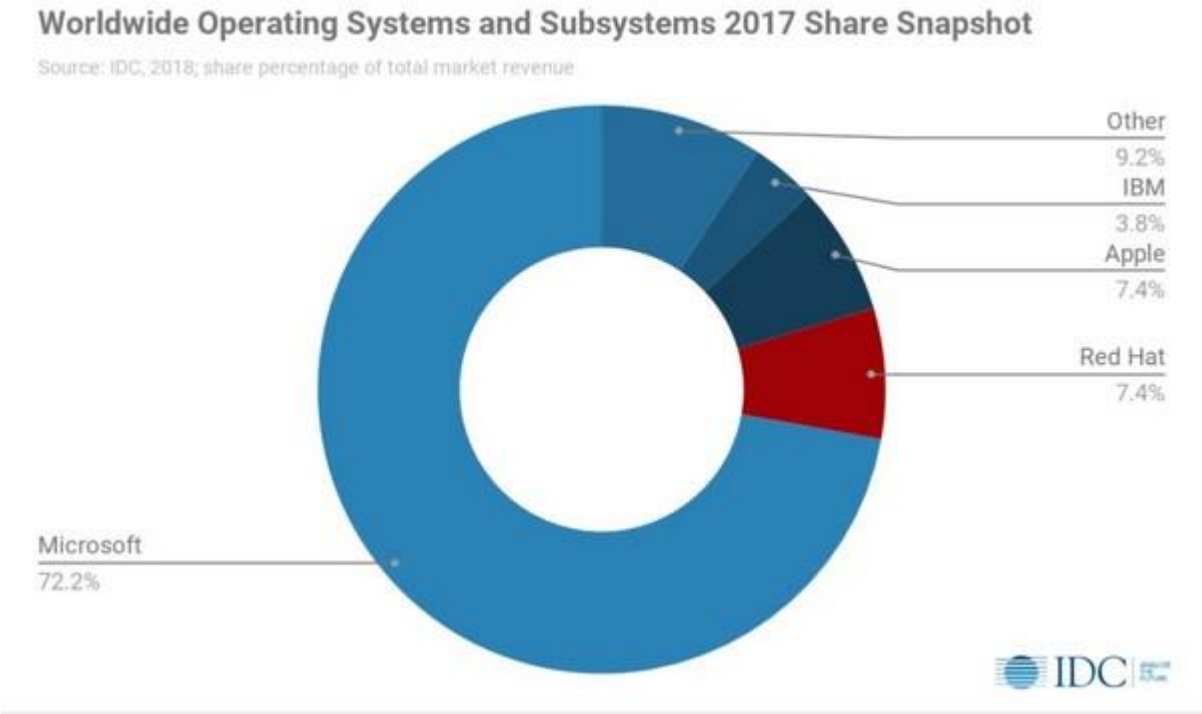
OS/390, которая в статье не рассматривалась. Другие серверные ОС занимают 6.7%.

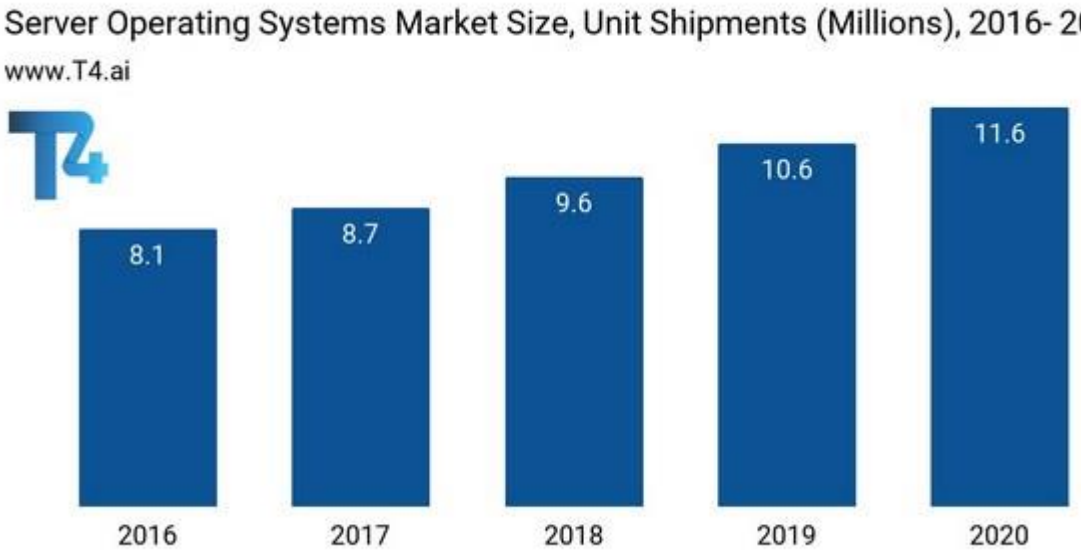


А вот как выглядит распределение доходов компаний, разрабатывающих и поддерживающих серверные ОС, а также различные подсистемы для них (данные IDC за 2107 год.)

Стоит также добавить: несмотря на экономический кризис, рынок серверных ОС растет стабильно увеличивающимися темпами. Вот какие

данные по числу поставок серверных ОС за 2016–2020 годы приводит аналитическая компания T4.ai:





LINUX

Linux — это семейство Unix-подобных операционных систем использующих ядро Linux, которое разработал финно-американский программист Линус Торвальдс. ОС, использующие ядро Linux, называются дистрибутивами Linux исходные коды которых являются открытыми, так как они

распространяются под лицензией GNU GPL, которая подразумевает создание бесплатного и открытого программного обеспечения (open-source software). Это означает, что у любого пользователя есть право изучать и изменять исходный код.

Появившись как решения вокруг созданного в начале 1990-х годов ядра, уже с начала 2000-х годов системы Linux являются основными для суперкомпьютеров и серверов, расширяется применение их для встраиваемых систем и мобильных устройств, некоторое распространение системы получили и для персональных компьютеров.

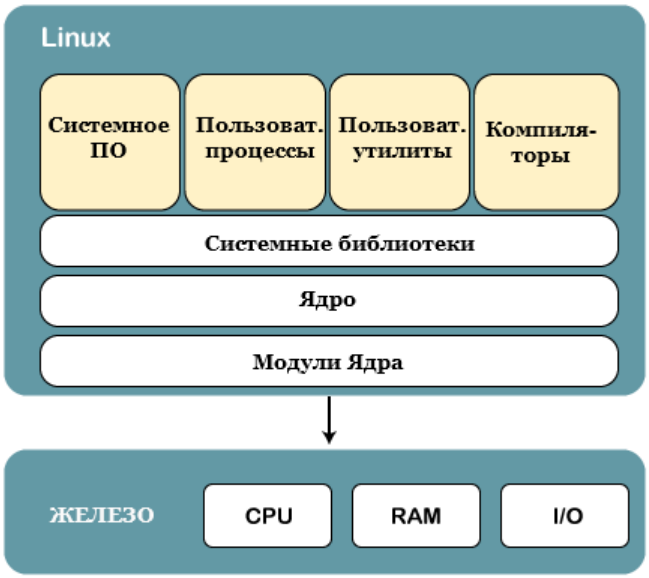
Семейство систем, включающих в качестве компонентов основные программы проекта GNU, такие как bash, gcc, glibc, coreutils, GNOME и ряд других, иногда идентифицируется как GNU/Linux. Так как традиционно большинство систем было именно таким, под «Linux» обычно подразумеваются именно они; притом существует спор об именовании GNU/Linux. Существует проект стандартизации внутренней структуры Linux-систем — Linux Standard Base, часть документов которого зарегистрирована в качестве стандартов ISO; но далеко не все системы сертифицируются по нему, и в целом для Linux-систем не существует какой-либо общепризнанной стандартной комплектации или формальных условий включения в семейство. Однако есть ряд систем на базе ядра Linux, но не имеющих в основе зависимости от программ GNU, которые поэтому GNU/Linux не называют, в частности, таковы мобильные системы Android и FirefoxOS.

Официальным логотипом и талисманом Linux является пингвин Tux, созданный в 1996 году Ларри Юингом[18]. Торговая марка «Linux» принадлежит создателю и основному разработчику ядра Линусу Торвальдсу. При этом проект Linux в широком смысле не принадлежит какой-либо организации или частному лицу, вклад в его развитие и распространение осуществляют тысячи независимых разработчиков и компаний, одним из инструментов взаимодействия которых являются группы пользователей Linux. Существует ряд некоммерческих объединений, ставящих основной целью развитие и продвижение Linux, наиболее крупное и влиятельное из них — основанный в 2007 году The Linux Foundation. Существует значительный рынок коммерческой технической поддержки Linux-систем, на котором с долей свыше 70 % (2017) доминирует корпорация Red Hat (поглощена IBM в 2019 году).

Ядро Linux было разработано в 1991 году программистом Линусом Торвальдсом. Об этом снят документальный фильм «Revolution OS» (2001 г.).

Структура Linux

Linux состоит из следующих компонентов:



Ядро

Ядро — это своего рода главная программа, являющаяся основной частью операционной системы. Оно выступает в роли посредника между устройствами компьютера (процессором, видеокартой, оперативной памятью и т.д.) и его программным обеспечением, абстрагируя от обычных программ и пользователей сложную, низкоуровневую работу с «железом» компьютера, предоставляя взамен простой, понятный и удобный в использовании интерфейс. Для этого в код ядра были включены драйверы устройств, которые могут, как загружаться в память вместе с ядром ОС, так и подключаться по мере возникновения потребности в ресурсах необходимого устройства.

Как вы наверняка знаете, на компьютере может быть запущено сразу несколько программ: какие-то из них работают в фоновом режиме, другие могут ожидать определенных действий от пользователя, а третьим необходимо получать информацию из другой запущенной программы. В такой ситуации именно ядро берет на себя функцию оптимального распределения ресурсов компьютера между запущенными программами и организацию параллельной работы множества различных процессов. Оно первым загружается в оперативную память компьютера и всегда находится в запущенном состоянии, постоянно взаимодействуя с его аппаратным обеспечением и установленными программами.

Как правило, большинство ядер делятся на три типа:

- микроядра;
- монолитные;
- гибридные.

Микроядро — это ядро, состоящее из нескольких подгружаемых в память по мере надобности независимых модулей, выполняющихся в отдельных адресных пространствах. По сути, грубо говоря, в таком варианте исполнения оно не сильно отличается от обычных прикладных программ. К достоинствам данного ядра можно отнести теоретически большую надежность в сравнении с другими архитектурами (в действительности же — не всё так радужно и гладко) и его модульность (легкость в подключении дополнительных частей ядра). К минусам же микроядерной архитектуры относится то, что ядро, построенное по такой схеме, получается очень медленным (ведь ему нужно постоянно переключаться между отдельными частями).

Монолитное ядро — это полная противоположность микроядра, т.к. в памяти компьютера всегда находится весь (или почти весь) код ядра. Вследствие чего, скорость его работы выше в сравнении с микроядром.

Гибридное ядро — это ядро, сочетающее в себе элементы как монолитной, так и микроядерной архитектур.

Ядро Linux хоть и относится к монолитным ядрам, но оно также заимствует и некоторые идеи из микроядерной архитектуры, что означает, что вся операционная система работает в пространстве ядра, а драйвера устройств (в виде модулей) могут быть легко загружены (или выгружены) прямо во время работы операционной системы.

Архитектура системы Linux.

Рассмотрим детально:

«Железо» — аппаратное обеспечение компьютера (процессор, видеокарта, оперативная память и т.д.) со всеми его периферийными устройствами.

Ядро — является основным компонентом операционной системы, взаимодействует непосредственно с аппаратным обеспечением, играя роль посредника между низкоуровневым «железом» и компонентами верхнего уровня.

Оболочка (или «командный интерпретатор») — интерфейс для взаимодействия между пользователями системы и ядром ОС, абстрагирующий внутреннее устройство системы. Принимает команды от пользователей и запускает на выполнение соответствующие функции.

Утилиты (vi, cat, sed, date и т.д.) — служебные программы, которые предоставляют пользователю большую часть функциональных возможностей операционной системы.

У ядра есть четыре обязанности:

- Управление устройствами. Система имеет множество подключенных к ней устройств, таких как: процессор, устройство памяти, звуковые карты, графические карты и т.д. Ядро хранит все данные, относящиеся ко всем устройствам, в драйвере устройства (без этого ядро не сможет управлять устройствами). Таким образом, ядро знает, что может сделать каждое устройство и как им манипулировать, чтобы добиться наилучшей производительности. Оно также управляет связью между всеми устройствами. Ядро имеет определенный набор правил, которым должны следовать все устройства.
- Управление памятью. Ядро отслеживает используемую и неиспользуемую память и с помощью механизма виртуальной памяти следит за тем, чтобы процессы не манипулировали данными друг друга.
- Управление процессами. Ядро управляет процессами, выделяя для их работы требуемое количество времени и наделяя их необходимым приоритетом выполнения. Оно также имеет дело с информацией о правах доступа, ограничениях безопасности и владельце каждого процесса.
- Обработка системных вызовов. Обработка системных вызовов означает, что программист может написать запрос к ядру или попросить ядро выполнить задачу.

Системные библиотеки

Системные библиотеки — это специальные программы, которые помогают получить доступ к функциям ядра. Для выполнения какой-либо задачи ядро сначала должно получить системный вызов, и этот вызов исходит от приложений. Но приложения должны знать, как выполнить данный вызов, потому что каждое ядро имеет свой набор системных вызовов. Для этого программистами была разработана стандартная библиотека процедур для взаимодействия с ядром, описывающая набор системных вызовов для каждой конкретной операционной системы.

Самой известной системной библиотекой для Linux является библиотека glibc (GNU C library).

Системные утилиты

В Linux имеется набор утилит, представляющий собой простые команды. Используя эти команды, вы можете, например, получить доступ к своим файлам и каталогам: редактировать и манипулировать данными, хранящимися в них, изменять расположение файлов и пр.

Утилиты разработки ПО

С помощью вышеперечисленных трех компонентов ваша ОС может запускаться и работать. Но для обновления системы и создания для нее новых программ у вас должны быть дополнительные инструменты и библиотеки. Данный набор инструментов называется toolchain. Toolchain — это жизненно-важный набор программ, утилит и инструментов, используемый разработчиками для создания рабочего приложения из исходных кодов.

Пользовательские программы

Подобные программы не относятся к обязательным компонентам операционной системы, и часто их пишут сами пользователи. Они нужны для того, чтобы пользователь мог задать компьютеру какую-то конкретную работу. К таким утилитам относятся: инструменты графического дизайна, офисные пакеты, браузеры, различные плееры и т.д.

Почему именно Linux?

Это один из самых часто задаваемых вопросов о Linux-системах. Почему мы используем другую и сложную операционную систему, если у нас есть более простая, такая как Windows? Система Linux имеет несколько отличительных особенностей. Linux может стать для вас идеальным вариантом в вопросе выбора операционной системы, если вы хотите избавиться от вирусов, вредоносных программ, различным сбоев и многого другого. Кроме того, Linux предоставляет различные преимущества по сравнению с другими операционными системами, и нам не нужно платить за это. Давайте рассмотрим некоторые из этих преимуществ:

Свободная и открытая операционная система

Большинство операционных систем поставляются в скомпилированном виде, что означает, что их основной исходный код был пропущен через компилятор, переводящий код системы на машинный язык, понятный компьютеру. Изменение данного скомпилированного кода — сложная и тяжелая работа. Вдобавок, лицензионное соглашение используемой программы может явно запрещать декомпиляцию и изменение своих файлов.

Но с open-source всё совсем иначе. Исходный код системы (или программы) поставляется вместе с её скомпилированной версией и предоставляет возможность изменять его любому, кто обладает достаточными знаниями. Это дает нам свободу запускать программу, свободу изменять код в соответствии с нашим потребностями, свободу распространять копии кода и свободу распространять копии, которые были подвержены нашим изменениям.

Безопасность

Linux поддерживает различные параметры безопасности, которые спасут вас от вирусов, вредоносных программ, замедлений, сбоев и пр. Кроме того, он будет защищать ваши данные. Именно это и составляет основную причину того, что он является наиболее выгодным вариантом для разработчиков. Конечно, он не является абсолютно безопасной ОС, но он менее уязвим, чем другие ОС. Действие каждого приложения должно быть авторизовано администратором системы. Вирус не будет запущен до тех пор, пока администратор сам не предоставит пароль доступа.

Выгодный выбор для разработчиков

Linux подходит для разработчиков, так как поддерживает почти все наиболее часто используемые языки программирования: C/C++, Java, Python, Ruby и другие. Кроме того, он облегчает работу с широким спектром полезных приложений для разработки ПО.

Разработчики считают, что терминал в Linux намного лучше командной строки в Windows. Менеджер пакетов в системе Linux поможет выполнить установку и обновление, как целых компонентов программного обеспечения, так и его отдельных частей. Bash-скрипты также будут очень полезны для программистов. Кроме того, поддержка SSH позволит быстро управлять серверами.

Гибкость

Linux — это гибкая ОС, которая может применяться где угодно: в настольных приложениях и серверах, во встроенных системах, в наручных часах, в суперкомпьютерах, в наших телефонах, ноутбуках, автомобилях и пр. Кроме того, Linux поддерживает различные варианты кастомизации.

Дистрибутивы

Многие фирмы и организации модифицировали операционную систему Linux, выпустив свои собственные дистрибутивы. На рынке существуют десятки (а может уже даже и сотни) различных Linux-дистрибутивов, предлагающие пользователям ознакомиться со своими уникальными разработками и преимуществами. На сегодняшний день, среди всех дистрибутивов Linux можно выделить несколько самых популярных, а именно: Ubuntu, Debian, Linux Mint, MX Linux, Arch Linux, Manjaro, Fedora, CentOS.

Команды Linux семейства Debian

Команда adduser/addgroup

Команды **adduser** и **addgroup** используются для добавления пользователя и группы в систему в соответствии с конфигурацией по умолчанию, указанной в файле **/etc/adduser.conf**.

```
$ sudo adduser sedicomm
$ sudo addgroup sedicomm
```

Команда agetty

Agetty — это команда, которая управляет физическими или виртуальными терминалами и вызывается **init.** Как только она обнаруживает соединение, сразу открывает порт **tty,** запрашивает имя пользователя для входа и вызывает команду **/bin/login**. **Agetty** — это замена **Linux getty**:

```
$ agetty -L 9600 ttyS1 vt100
```

Команда alias

Alias - полезная встроенная командная оболочка для создания псевдонимов (ярлыков) для команд **Linux**. Это полезно для создания новых/пользовательских команд из существующих команд оболочки **Linux** (включая опции):

```
$ alias home='cd /home/sedicomm/'
```

Вышеупомянутая команда создаст псевдоним, называемый **home** для каталога **/home/sedicomm/**, поэтому всякий раз, когда вы вводите **home** в командной строке терминала, она помещает вас в каталог **/home/sedicomm/**.

Команда anacron

Anacron — это команда Linux, используемая для периодического запуска команд с частотой, определенной в днях, неделях и месяцах.

В отличие от **cron,** она предполагает, что система не будет работать непрерывно, поэтому, если запланированное задание выполняется, когда система выключена, она запускается после включения устройства.

Команда apropos

Команда **apropos** используется для поиска и отображения краткой справочной страницы команды/программы следующим образом:

```
$ apropos adduser
```

Команда apt

Инструмент **apt** — относительно новый менеджер пакетов высокого уровня для систем **Debian/Ubuntu:**

```
$ sudo apt update
```

Команда apt-get

Apt-get — мощный и бесплатный менеджер пакетов интерфейса для систем **Debian/Ubuntu.** Он используется для установки новых, удаления доступных и обновления существующих пакетов программного обеспечения, а также обновления всей операционной системы.

```
$ sudo apt-get update
```

Команда aptitude

Aptitude — это мощный текстовый интерфейс для системы управления пакетами **Debian GNU/Linux**. Такими как **apt-get** и **apt.** Его можно использовать для установки, удаления или обновления пакетов программного обеспечения в системе.

```
$ sudo aptitude update
```

Команда arch

Arch — простая команда для отображения архитектуры машины или имени оборудования (аналогично uname -m):

```
$ arch
```

Команда arp

ARP (протокол разрешения адресов) — это протокол, который отображает **IP**-адреса сети с адресами аппаратного обеспечения (MAC) в сети **IPv4**.

Вы можете использовать его, как показано ниже, чтобы найти все «живые» хосты в сети:

```
$ sudo arp-scan --interface=enp2s0 –localnet
```

Команда at

Команда **at** используется для планирования задач в будущем. Это альтернатива **cron** и **anacron,** однако она запускает задачу один раз в будущем без редактирования любых файлов конфигурации:

Например, чтобы отключить систему сегодня в 23:55, запустите:

```
$ sudo echo "shutdown -h now" | at -m 23:55
```

Команда atq

Команда **atq** используется для просмотра заданий в командной очереди:

```
$ atq
```

Команда atrm

Команда **atrm** используется для удаления заданий (обозначенных их номером) из очереди команд:

```
$ atrm 2
```

Команда awk

Awk — это мощный язык программирования, созданный для обработки текста и обычно используемый в качестве инструмента для извлечения данных и создания отчетов.

```
$ awk '//{print}' /etc/hosts
```

Командная batch

Команда также используется для планирования задач в будущем, аналогично команде **at.**

Команда basename

Команда **basename** выводит имя файла, удаляя каталоги в абсолютном пути:

```
$ basename bin /findhosts.sh
```

Команда bc

Bc — простой, но мощный и произвольный язык калькулятора **CLI**, который можно использовать следующим образом:

```
$ echo 20.05 + 15.00 | bc
```

Команда bg

Bg — это команда, используемая для отправки процесса в фоновый режим.

```
$ tar -czf home.tar.gz
$ bg
$ jobs
```

Команда bzip2

Команда **bzip2** используется для сжатия или распаковки файлов.

```
$ bzip2 -z filename #Compress
$ bzip2 -d filename.bz2 #Decompress
```

Команда cal

Команда **cal** выводит календарь.

```
$ cal
```

Команда cat

Cat используется для просмотра содержимого файла или данных, представленных и отображенных в терминале.

```
$ cat file.txt
```

Команда chgrp

Команда **chgrp** используется для изменения правила группового владения файлом. Укажите новое имя группы в качестве первого аргумента, а имя файла — как второй аргумент:

```
$ chgrp sedicomm users.txt
```

Команда chmod

Команда **chmod** используется для изменения/обновления прав доступа к файлу:

```
$ chmod +x sysinfo.sh
```

Команда chown

Команда **chown** изменяет/обновляет права доступа пользователей и групп к файлу/каталогу:

```
$ chmod -R www-data:www-data /var/www/html
```

Команда cksum

Команда **cksum** используется для отображения контрольной суммы **CRC** и количества байт входного файла.

```
$ cksum README.txt
```

Команда clear

Команда **clear** позволяет очистить экран терминала, для этого просто введите её в терминал:

```
$ clear
```

Команда cmp

Cmp выполняет побайтное сравнение двух файлов:

```
$ cmp file1 file2
```

Команда comm

Команда **comm** используется для сравнения двух отсортированных по очереди файлов. Это возможно с помощью команды:

```
$ comm file1 file2
```

Команда cp

Команда **cp** используется для копирования файлов и каталогов из одного места в другое.

```
$ cp /home/sedicomm/file1 / home/sedicomm/personal/
```

Команда date

Команда **date** отображает/устанавливает системную дату и время следующим образом.

```
$ date
$ date --set = "8 JUN 2017 13:00:00"
```

Команда dd

Команда **dd** используется для копирования файлов, преобразования и форматирования в соответствии с флагами, указанными в командной строке. Она может разбивать заголовки, извлекать части двоичных файлов и так далее.

В приведенном ниже примере показано создание загрузочного **USB**-устройства:

```
$ dd if=/home/sedicomm/kali-linux-1.0.4-i386.iso of=/dev/sdc1 bs=512M; sync\
```

Команда df

Команда **df** используется для демонстрации использования дискового пространства файловой системы:

```
$ df –h
```

Команда diff

Команда **diff** используется для сравнения двух файлов по строкам. Её также можно использовать, чтобы найти разницу между двумя каталогами в Linux:

```
$ diff file1 file2
```

Команда dir

Команда **dir** работает как команда **ls,** она перечисляет содержимое каталога.

```
$ dir
```

Команда dmidecode

Команда **dmidecode** — это инструмент для извлечения информации об оборудовании любой системы Linux. Он преобразовывает содержимое таблицы **DMI** компьютера в удобный для чтения формат.

Чтобы просмотреть информацию о системном оборудовании, вы можете ввести:

```
$ sudo dmidecode --type system
```

Команда du

Du используется для отображения дискового пространства файлов, присутствующего в каталоге, а также его подкаталогах следующим образом:

```
$ du /home/aaronkilik
```

Команда echo

Команда **echo** выводит текст в строку выделенную для неё:

```
$ echo “This is Sedicomm - Linux How Tos”
```

Команда eject

Команда **eject** используется для извлечения съемных носителей, таких как **DVD** или **CD ROM** из системы.

```
$ eject /dev/cdrom
$ eject /mnt/cdrom/
$ eject /dev/sda
```

Команда env

Команда **env** перечисляет все текущие переменные среды и используется для их установки.

```
$ env
```

Команда exit

Команда **exit** используется для выхода из оболочки.

```
$ exit
```

Команда expr

Команда **expr** используется для вычисления выражений, как показано ниже:

```
$ expr 20 + 30
```

Команда factor

Factor используется для отображения простых коэффициентов числа.

```
$ factor 10
```

Команда Find

Find позволяет искать файлы в каталоге, а также в его подкаталогах. Она ищет файлы по таким атрибутам: разрешения, пользователи, группы, тип файла, дата, размер и т.д..

```
$ find /home/sedicomm/ -name trolo-lo.txt
```

Команда Free

Free, показывает использование системной памяти в системе, включая пространство подкачки. Используйте параметр **-h** для отображения вывода в удобном пользовательском формате.

```
$ free -h
```

Команда grep

Команда **grep** выполняет поиск заданного шаблона в файле (или файлах) и выводит результат в выходных строках, содержащих этот шаблон, следующим образом:

```
$ grep 'sedicomm' domain-list.txt
```

Команда groups

Команда **groups** отображает все имена групп, в которых задействован пользователь.

```
$ groups
$ group sedicomm
```

Команда gzip

Gzip помогает сжать файл, меняя его расширение на «**.gz**«, как показано ниже:

```
$ gzip passwds.txt
$ cat file1 file2 | gzip > foo.gz
```

Команда gunzip

Gunzip восстанавливает файлы, сжатые командой **gzip**.

```
$ gunzip foo.gz
```

Команда head

Head используется для отображения первых строк (10 строк по умолчанию) указанного файла или **stdin** на экран:

```
# ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem | head
```

Команда History

History используется для отображения ранее использованных команд или для получения информации о команде, выполняемой пользователем.

```
$ history
```

Команда hostname

Hostname используется для вывода или установки имени системного хоста в **Linux**.

```
$ hostname
$ hostname NEW_HOSTNAME
```

Команда hostnamectl

Команда **hostnamectl** управляет именем системного хоста под **systemd.** Она используется для вывода или изменения имени системного хоста и любых связанных настроек:

```
$ hostnamectl
$ sudo hostnamectl set-hostname NEW_HOSTNAME
```

Команда Hwclock

Hwclock — это инструмент для управления аппаратными часами системы.

```
$ sudo hwclock
$ Sudo hwclock --set --date 8/06/2017
```

Команда hwinfo

Hwinfo используется для проверки оборудования, присутствующего в системе Linux.

```
$ hwinfo
```

Команда id

Id показывает пользовательскую и групповую информацию для текущего пользователя или указанного имени пользователя, как показано в примере ниже:

```
$ id sedicomm
```

Команда ifconfig

Команда **ifconfig** используется для настройки, просмотра и управления сетевыми интерфейсами Linux.

```
$ Ifconfig
$ sudo ifconfig eth0 up
$ sudo ifconfig eth0 down
$ sudo ifconfig eth0 172.16.25.125
```

Команда ionice

Ionice используется для установки или просмотра класса планирования ввода-вывода процесса и приоритета указанного процесса.

Если она вызывается без каких-либо параметров, то будет запрашивать текущий класс планирования и приоритет ввода-вывода для текущего процесса:

```
$ ionice -c 3 rm /var/logs/syslog
```

Команда iostat

Iostat используется для отображения статистики **CPU** и ввода/вывода для устройств и разделов. Она создает полезные отчеты для обновления конфигураций системы, чтобы помочь сбалансировать нагрузку ввода/вывода между физическими дисками.

```
$ iostat
```

Команда ip

ip — утилита командной строки в **Linux** из пакета **iproute2**. Позволяет выполнять настройку сетевой подсистемы и является заменой таких утилит, как **ifconfig**, **route**, **arp**.

Эта команда назначит **IP**-адрес определенному интерфейсу (eth1 в этом случае).

```
$ sudo ip addr add 192.168.56.10 dev eth1
```

Команда iptables

Iptables — это брандмауэр на основе терминалов для управления входящим и исходящим трафиком через набор настраиваемых правил таблиц.

Приведенная ниже команда используется для проверки существующих правил в системе (для этого могут потребоваться привилегии root).

```
$ sudo iptables -L -n –v
```

Команда iw

Команда **iw** используется для управления беспроводными устройствами и их конфигурацией.

```
$ iw list
```

Команда iwlist

Команда **iwlist** отображает подробную беспроводную информацию с беспроводного интерфейса. Приведенная ниже команда позволяет получить подробную информацию о интерфейсе **wlp1s0**.

```
$ iwlist wlp1s0 scanning
```

Команда kill

Kill используется для завершения процесса с использованием его **PID**, посылая ему сигнал (сигнал по умолчанию для kill — TERM).

```
$ kill -p 2300
$ kill -SIGTERM -p 2300
```

Команда killall

Команда **killall** используется для завершения процесса с использованием его имени.

```
$ killall firefox
```

Команда kmod

Команда **kmod** используется для управления модулями ядра Linux. Чтобы просмотреть список всех загруженных модулей, введите:

```
$ kmod list
```

Команда Last

Команда last отображает список последних зарегистрированных пользователей.

```
$ last
```

Команда ln

Команда **ln** используется для создания символьной ссылки между файлами с использованием флага **-s**, вот таким образом.

```
$ ln -s /usr/bin/lscpu cpuinfo
```

Команда locate

Команда **locate** используется для поиска файла по имени. Приведенная ниже команда будет искать файл по его точному имени:

```
$ locate -b '\ domain-list.txt
```

Команда login

Команда **login** используется для создания нового сеанса в системе. Вам будет предложено указать имя пользователя и пароль для входа в систему, это можно выполнить с помощью команды показанной ниже:

```
$ sudo login
```

Команда ls

Команда **ls** используется для отображения содержимого каталога. Она работает аналогично команде **dir**.

Параметр **-l** позволяет использовать длинный формат списка.

```
$ ls -l file1
```

Команда lshw

Команда **lshw** является самым простым инструментом для получения подробной информации об аппаратной конфигурации машины, вызывайте её с привилегиями суперпользователя, для получения более полной информации:

```
$ sudo lshw
```

Команда lscpu

Команда **lscpu** отображает информацию о архитектуре ЦП (количество процессоров, потоков, ядер, сокетов и т.д.).

```
$ lscpu
```

Команда lsof

Команда **lsof** отображает информацию, связанную с файлами, открытыми процессами. Файлы могут быть любого типа, включая обычные файлы, каталоги, специальные файлы, специальные файлы символов, библиотеки и потоковые/сетевые файлы.

Чтобы просмотреть файлы, открытые процессами конкретного пользователя, введите команду приведенную ниже:

```
$ lsof -u sedicomm
```

Команда lsusb

Команда **lsusb** показывает информацию о шинах **USB** как в системе так и на устройствах, подключенных к ним.

```
$ lsusb
```

Команда Man

Man используется для просмотра справочных страниц для команд и программ.

```
$ man du
$ man df
```

Команда md5sum

Команда **md5sum** используется для вычисления и вывода дайджеста сообщения **MD5** файла. Если запустить без аргументов, **debsums** проверяет каждый файл в вашей системе на наличие файлов **md5sum:**

```
$ sudo debsums
```

Команда mkdir

Команда **mkdir** используется для создания одного или нескольких каталогов, если они еще не существуют (их можно переопределить с помощью опции **-p**).

```
$ mkdir sedicomm-files
$ mkdir -p sedicomm-files
```

Команда more

Команда **more** позволяет просматривать относительно длинные текстовые файлы на одном экране.

```
$ more file.txt
```

Команда mv

Команда **mv** используется для переименования файлов или каталогов. Она также перемещает файл или каталог в другое место в структуре каталогов.

```
$ mv test.sh sysinfo.sh
```

Команда nano

Nano — популярный небольшой, бесплатный и удобный текстовый редактор для Linux. Чтобы открыть файл с помощью **nano**, введите:

```
$ nano file.txt
```

Команда nc/netcat

Nc (или netcat) используется для выполнения любой операции, связанной с сокетами **TCP**, **UDP** или **UNIX**. Она может обрабатывать как **IPv4**, так и **IPv6** для открытия **TCP**-соединений, отправки **UDP**-пакетов, прослушивания на произвольных портах **TCP** и **UDP**, выполнения сканирования портов.

Приведенная ниже команда поможет нам узнать, открыт ли **22** порт на хосте **192.168.56.5**.

```
$ nc -zv 192.168.1.5 22
```

Команда netstat

Команда **netstat** отображает полезную информацию о сетевой подсистеме Linux (сетевые подключения, таблицы маршрутизации, статистику интерфейсов, соединения маскарадов и членства в многоадресной рассылке).

Эта команда отобразит все открытые порты в локальной системе:

```
$ netstat -a | more
```

Команда nice

Команда **nice** используется для показа или изменения значения **nice** запущенной программы. При запуске без какого-либо параметра, она выводит текущее значение **nice**.

Следующая команда запускает процесс «tar command», устанавливая значение «nice» равным 12.

```
$ nice -12 tar -czf backup.tar.bz2 /home/*
```

Команда nmap

Nmap — популярный и мощный инструмент для открытого сканирования и проверки безопасности. Он предназначен для быстрого сканирования больших сетей, но он также отлично работает с одиночными хостами. Приведенная ниже команда будет проверять открытые порты на всех хостах в указанной сети.

```
$ nmap -sV 192.168.56.0/24
```

Команда nproc

Команда **nproc** показывает количество обрабатывающих блоков, присутствующих в текущем процессе.

```
$ nproc
```

Команда openssl

Openssl — это инструмент командной строки для использования различных криптографических операций библиотеки **OpenSSL**. Приведенная ниже команда создаст архив всех файлов в текущем каталоге и зашифрует содержимое архивного файла:

```
$ tar -czf - * | openssl enc -e -aes256 -out backup.tar.gz
```

Команда passwd

Команда **passwd** используется для создания/обновления паролей для учетных записей пользователей, а также изменения учетной записи или связанного с ней срока действия пароля. Обратите внимание, что обычные пользователи системы могут изменять пароль только своей учетной записи, а **root** может изменять пароль для любой учетной записи.

```
$ passwd sedicomm
```

Команда pidof

Pidof отображает идентификатор процесса запущенной программы/команды.

```
$ pidof init
$ pidof cinnamon
```

Команда ping

Команда **ping** используется для определения отклика подключения между хостами в сети (или в Интернете):

```
$ ping sedicomm.com
```

Команда ps

Ps показывает полезную информацию об активных процессах, запущенных в системе. В приведенном ниже примере показаны запущенные процессы с максимальной загрузкой памяти и использованием ЦП.

```
# ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem | head
```

Команда pstree

Pstree показывает запущенные процессы, которое берут истоки либо в PID, либо в init.

```
$ pstree
```

Команда pwd

Команда **pwd** отображает имя текущего/рабочего каталога, как показано в примере ниже:

```
$ pwd
```

Команда rdiff-backup

Rdiff-backup — мощный локальный/удаленный инкрементный сценарий резервного копирования, написанный на **Python**. Он работает в любой операционной системе **POSIX**, такой как **Linux**, **Mac OS X**.

Обратите внимание, что для удаленных резервных копий вы должны установить ту же версию **rdiff—backup** как на локальном, так и на удаленном компьютере. Ниже приведен пример локальной команды резервного копирования:

```
$ sudo rdiff-backup /etc /media/sedicomm/Backup/server_etc.backup
```

Команда reboot

Команда **reboot** может использоваться для остановки, выключения или перезагрузки системы следующим образом:

```
$ reboot
```

Команда rename

Команда **rename** используется для переименования нескольких файлов одновременно. Если у вас есть набор файлов с расширением «.html», и вы хотите поменять их расширение на «.php», это можно выполнить введя команду приведенную ниже:

```
$ rename '/ / .html $ / \. Php /' * .html
```

Команда rm

Команда **rm** используется для удаления файлов или каталогов, как показано в примере ниже:

```
$ rm file1
$ rm -rf my-files
```

Команда rmdir

Команда **rmdir** помогает удалить пустые каталоги следующим образом:

```
$ rmdir /backup /all
```

Команда scp

Команда **scp** позволяет безопасно копировать файлы между хостами в сети.

```
$ scp ~/names.txt root@192.168.56.10:/root/names.txt
```

Команда shutdown

Команда **shutdown** устанавливает время, в течение которого система будет выключена. Команда может использоваться для остановки, отключения питания или перезагрузки машины.

```
$ shutdown –poweroff
```

Команда sleep

Команда **sleep** используется для задержки или приостановки (в частности, выполнения команды) в течение определенного периода времени.

```
$ check.sh; sleep 5; sudo apt update
```

Команда Sort

Sort используется для сортировки строк текста в указанных файлах или из **stdin,** как показано ниже:

```
$ cat words.txt
```

Команда split

Split, как следует из названия, используется для разделения большого файла на мелкие части.

```
$ tar -cvjf backup.tar.bz2 /home/sedicomm/Documents/*
```

Команда ssh

Ssh (SSH client) — это команда для удаленного доступа и запуска команд на удаленном компьютере. Она предназначен для обеспечения защищенной зашифрованной связи между двумя ненадежными хостами по небезопасной сети, такой как Интернет.

```
$ ssh sedicomm@192.168.56.10
```

Команда stat

Stat используется для отображения состояния файловой системы (опция -f используется для указания файловой системы).

```
$ stat file1
```

Команда su

Команда **su** используется для переключения на другой идентификатор пользователя или включения пользователя **root** во время сеанса входа в систему. Обратите внимание, что когда **su** вызывается без имени пользователя, по умолчанию пользователь — **root**.

```
$ su
$ su sedicomm
```

Команда sudo

Sudo позволяет разрешенному пользователю системы запускать команду как **root** пользователь, по правилам определенным политикой безопасности, такой как **sudoers.**

```
$ Sudo apt update
$ Sudo useradd sedicomm
$ Sudo passwd sedicomm
```

Команда sum

Sum используется для отображения контрольной суммы и количества блоков для каждого указанного файла в командной строке.

```
$ sum output file.txt
```

Команда tac

Команда **tac** объединяет и отображает файлы в обратном порядке. Она просто выводит каждый файл в терминал, сначала показывая последнюю строку.

```
$ tac file.txt
```

Команда tail

Команда **tail** используется для отображения последних строк (по 10 строк по умолчанию) каждого файла для стандартного вывода.

Если имеется более одного файла, перед каждым заголовком указывается имя файла. Используйте её следующим образом (укажите больше строк для отображения с использованием опции -n).

```
$ tail long-file
```

```
$ tail -n 15 long-file
```

Командная talk

Команда **talk** используется для общения с другим пользователем системы/сети. Чтобы поговорить с пользователем используйте его имя для входа, однако, чтобы поговорить с пользователем на другом компьютере, используйте «user @ host».

```
$ talk person [ttyname]
$ talk 'user @ host '[ttyname]
```

Команда tar

Команда **tar** — самая мощная утилита для архивирования файлов в Linux.

```
$ tar -czf home.tar.gz
```

Команда tee

Команда **tee** используется для чтения файлов с терминала, как показано в примере ниже.

```
$ echo "Testing how tee command works" | tee file1
```

Команда Time

Time запускает программы и суммирует использование ресурсов системы.

```
$ time wc /etc/host
```

Команда top

Команда **top** отображает все процессы в системе **Linux** в отношении использования памяти **ЦП** и обеспечивает динамическое представление текущей системы в режиме реального времени.

```
$ top
```

Команда Touch

Touch команда изменяет временные метки файла, ее также можно использовать для создания файла следующим образом.

```
$ touch file.txt
```

Команда tr

Tr — полезная утилита, используемая для перевода (изменения) или удаления символов из **stdin** и записи результата в **stdout**.

```
$ cat domain-list.txt | Tr [: lower:] [: upper:]
```

Команда uname

Команда **uname** отображает системную информацию, такую как операционная система, имя ядра узла хоста, версия, дата релиза и т.д.

Используйте опцию **-a**, чтобы отобразить всю системную информацию:

```
$ uname
```

Команда uniq

Команда **uniq** отображает или пропускает повторяющиеся строки терминала. Чтобы указать количество вхождений строки, используйте параметр **-c**.

```
$ cat domain-list.txt
```

Команда uptime

Uptime показывает, сколько времени система работает, количество зарегистрированных пользователей и среднюю загрузку системы.

```
$ uptime
```

Команда User

User показывает имена пользователей, которые вошли в настоящее время.

Команда vim/vi

Vim (улучшенный Vi) популярный текстовый редактор для **Unix**-подобных операционных систем. Он может использоваться для редактирования всех видов текстовых и программных файлов.

```
$ vim file
```

Команда w

Команда **w** отображает время безотказной работы системы, загружает средние значения и информацию о пользователях, находящихся в данный момент на машине, и о том, что они делают (их процессы).

```
$ w
```

Команда Wall

Wall используется для отправки сообщения всем пользователям системы.

```
$ wall “This is Sedicomm – Linux How Tos”
```

Команда watch

Команда **watch** запускает программу повторно, пока она отображается в полноэкранном режиме. Её также можно использовать для просмотра изменений в файле/каталоге. В приведенном ниже примере показано, как просматривать содержимое каталога.

```
$ watch -d ls -l
```

Команда wc

Команда **wc** используется для отображения значений строк, слов и байт для каждого указанного файла в очереди.

```
$ wc filename
```

Команда wget

Команда **wget** — это простая утилита, используемая для загрузки файлов из Интернета неинтерактивным (может работать в фоновом режиме) способом.

```
$ wget -c http://ftp.gnu.org/gnu/wget/wget-<span< a="" style="box-sizing: border-box;"> class="hljs-number">1.5.3.tar.gz</span<>
```

Команда whatis

Команда **whatis** выполняет поиск и показывает краткие или однострочные описания страниц с указанными именами команд.

```
$ whatis wget
```

Команда which

Команда **which** отображает абсолютный путь (пути) файлов (или, возможно, ссылок), которые будут выполняться в текущей среде.

```
$ which who
```

Команда who

Команда **who** показывает информацию о пользователях, которые в настоящее время вошли в систему.

```
$ who
```

Команда whereis

Команда **whereis** помогает нам находить двоичные файлы, исходные файлы и т.д..

```
$ whereis cat
```

Команда xargs

Команда **xargs** — полезная утилита для чтения элементов с терминала, разделенных пробелами или иными символами (двойными или одинарными кавычками или обратной косой чертой).

В приведенном ниже примере показано, что **xargs** используются для копирования файла в несколько каталогов **Linux**.

```
$ echo /home/aaronkilik/test/ /home/aaronkilik/tmp | xargs -n 1 cp -v
/home/aaronkilik/bin/sys_info.sh
```

Команда Yes

Команда **yes** используется для отображения строки несколько раз, пока она не будет завершена с помощью [Ctrl + C].

```
$ yes "This is Sedicomm - Linux HowTos"
```

Команда youtube-dl

Youtube-dl — это легкая команда командной строки для загрузки видео, а также для извлечения треков **MP3** с сайта **YouTube.com** и еще некоторых сайтов.

Приведенная ниже команде перечислит доступные форматы видео для выбранной ссылки.

```
$ youtube-dl --list-formats https://www.youtube.com/watch?v=iR
```

Команда zcmp / zdiff

Zcmp и **zdiff** утилиты, используемые для сравнения сжатых файлов, как показано в приведенном ниже примере.

```
$ zcmp domain-list.txt.zip basic_passwords.txt.zip
$ zdiff domain-list.txt.zip basic_passwords.txt.zip
```

Команда zip

Zip — простая в использовании команда, используемая для сжатия (архивирования) файлов.

```
$ tar cf -. | Zip | dd of =/dev/nrst0 obs=16k
$ zip inarchive.zip foo.c bar.c --out outarchive.zip
$ tar cf -. | zip backup –
```

Команда zz

Команда **zz** является псевдонимом инструмента командной строки **fasd,** который обеспечивает быстрый доступ к файлам и каталогам в Linux. Она используется для быстрого и интерактивного ввода **cd** в ранее доступный каталог, выбирая номер каталога из первого поля следующим образом.

Windows Server

Windows Server – наиболее распространенная операционная система для серверов. Компьютер, на котором установлена такая операционная система, может выполнять роли файлового сервера, сервера службы веб-приложений, сервера терминалов, почтового сервера, сервера удаленного доступа, службы DNS (доменных имен), службы каталогов, сервера потоков мультимедиа и другие.

Windows Server является достаточно быстрой и надежной ОС. Надежность обеспечивает платформа приложений, в которую встроены функции сервера приложений, а также интегрированная среда обеспечивающая доступность и безопасность информации.

Эту ОС достаточно просто развернуть и проводить ее администрирование. Она дает возможность управлять сетью с помощью политик и автоматизации выполнения каких-либо задач.

Для организаций, работающих с важными сетевыми бизнес-приложениями очень важна служба кластеров, позволяющая объединить несколько серверов. Узлы в кластере взаимно заменяемы. То есть, если один из серверов становится недоступным, обслуживание переходит на другой сервер.

Сегодня многие локальные сети объединены с интрасетями, экстрасетями и веб-узлами. В связи с этим требования к безопасности системы резко возросли. Современные ОС Windows Server тщательно проверяются на наличие слабых мест и точек ошибок. Например, безопасность вычислительной среды обеспечивает общая языковая среда исполнения.

Очень многие особенности Windows Server дают возможность сотрудникам организации работать более производительно. Файловые службы и службы печати позволяют управлять файловыми ресурсами. В то же время сохраняется безопасность и доступ для пользователей. Служба каталогов в Windows Server называется Active Directory. Благодаря этой службе сохраняются сведения о сетевых объектах. Кроме того, Active Directory позволяет администратору быстро найти данную информацию. Эта служба также упрощает работу по проектированию, развертыванию и управлению каталогами сети. В оптимальной работе Active Directory поможет консоль управления групповыми политиками.

До выпуска Windows Server 2012 на выделенных серверах для управления доступом к файлам и папкам использовалась система Access Control List. Она позволяла давать доступ только на основании учетных записей пользователей или членства пользователя в группе.

Модель управления доступом выглядела приблизительно так. Пусть есть общая папка, расшаренная или на уровне NTFS. Для доступа к ней составлялся список групп или пользователей, имеющих право доступа. Для того, чтобы у пользователя был доступ к какой-то папке, он должен быть включен в этот список или в группу, у которой есть право доступа.

У такой модели присутствуют некоторые недостатки. Во-первых, Если есть достаточно много общих ресурсов, то придется создать много групп. Во-вторых, нельзя проконтролировать доступ с учетом каких-либо атрибутов пользователя либо характеристик устройства, с которого осуществляется подключение. Достаточно членства пользователя в определенной группе. В-

третьих, достаточно проблематично осуществить сложные сценарии доступа. Пользователь может случайно выложить закрытую информацию в общую папку, где каждый из членов группы может ее прочитать.

Поэтому, Windows Server 2012 появилась концепция Dynamic Access Control. Она позволяет управлять доступом к файлам и папкам во всей сети. DAC делает возможным управление доступом на основании любого атрибута или критерия. С помощью этой концепции можно создавать правила доступа к данным, проводящие проверку членства пользователя в тех или иных группах. Эти правила применимы к любому из серверов сети в форме политик. Тем самым создается общая система безопасности.

Вредоносное ПО

Термин «вредоносное ПО» используется для описания любой вредоносной программы на компьютере или мобильном устройстве. Эти программы устанавливаются без согласия пользователей и могут вызывать ряд неприятных последствий, таких как снижение производительности компьютера, извлечение из системы персональных данных пользователя, удаление данных или даже воздействие на работу аппаратных средств компьютера. Поскольку киберпреступники придумывают все более сложные способы проникновения в системы пользователей, рынок вредоносных программ существенно расширился. Давайте рассмотрим некоторые из наиболее распространенных типов вредоносных программ, которые можно встретить в интернете.

1. Вирусы

Компьютерные вирусы получили свое название за способность «заражать» множество файлов на компьютере. Они распространяются и на другие машины, когда зараженные файлы отправляются по электронной почте или переносятся пользователями на физических носителях, например, на USB-накопителях или (раньше) на дискетах. По данным Национального института

стандартов и технологий (NIST) , первый компьютерный вирус под названием «Brain» был написан в 1986 году двумя братьями с целью наказать пиратов, ворующих ПО у компании. Вирус заражал загрузочный сектор дискет и передавался на другие компьютеры через скопированные зараженные дискеты.

2. Черви

В отличие от вирусов, червям для распространения не требуются вмешательства человека: они заражают один компьютер, а затем через компьютерные сети распространяются на другие машины без участия их владельцев. Используя уязвимости сети, например, недостатки в почтовых программах, черви могут отправлять тысячи своих копий и заражать все новые системы, и затем процесс начинается снова. Помимо того, что многие черви просто «съедают» системные ресурсы, снижая тем самым производительность компьютера, большинство из них теперь содержит вредоносные «составляющие», предназначенные для кражи или удаления файлов.

3. Рекламное ПО

Одним из наиболее распространенных типов вредоносных программ является рекламное ПО. Программы автоматически доставляют рекламные объявления на хост-компьютеры. Среди разновидностей Adware - всплывающие рекламные объявления на веб-страницах и реклама, входящая в состав «бесплатного» ПО. Некоторые рекламные программы относительно безвредны, в других используются инструменты отслеживания для сбора информации о вашем местонахождении или истории посещения сайтов и вывода целевых объявлений на экран вашего компьютера. BetaNews сообщил об обнаружении нового типа рекламного ПО, который может отключить антивирусную защиту. Поскольку Adware устанавливается с согласия пользователя, такие программы нельзя назвать вредоносными: обычно они идентифицируются как «потенциально нежелательные программы».

4. Шпионское ПО

Шпионское ПО делает то, что предполагает его название - следит за вашими действиями на компьютере. Оно собирает информацию (например, регистрирует нажатия клавиш на клавиатуре вашего компьютера, отслеживает, какие сайты вы посещаете и даже перехватывает ваши регистрационные данные), которая затем отправляется третьим лицам, как правило, киберпреступникам. Оно также может изменять определенные параметры защиты на вашем компьютере или препятствовать сетевым соединениям. Как пишет TechEye, новые типы шпионских программ позволяют злоумышленникам отслеживать поведение пользователей (естественно, без их согласия) на разных устройствах.

5. Программы-вымогатели

Программы-вымогатели заражают ваш компьютер, затем шифруют конфиденциальные данные, например, личные документы или фотографии, и требуют выкуп за их расшифровку. Если вы отказываетесь платить, данные удаляются. Некоторые типы программ-вымогателей могут полностью заблокировать доступ к вашему компьютеру. Они могут выдавать свои действия за работу правоохранительных органов и обвинить вас в каких-либо противоправных поступках. В июне 2015 года в Центр приёма жалоб на мошенничество в Интернете при ФБР обратились пользователи, сообщившие о финансовых потерях на общую сумму 18 000 000 долларов в результате деятельности вируса-вымогателя CryptoWall.

6. Боты

Боты - это программы, предназначенные для автоматического выполнения определенных операций. Они могут использоваться для легитимных целей, но злоумышленники приспособили их для своих вредоносных целей. Проникнув в компьютер, боты могут заставить его выполнять определенные команды без одобрения или вообще без ведома пользователя. Хакеры могут также пытаться заразить несколько компьютеров одним и тем же ботом, чтобы создать бот-сеть, которая затем будет

использоваться для удаленного управления взломанными машинами - красть конфиденциальные данные, следить за действиями жертвы, автоматически распространять спам или запускать разрушительные DDoS-атаки в компьютерных сетях.

7. Руткиты

Руткиты позволяют третьей стороне получать удаленный доступ к компьютеру и управлять им. Эти программы используются IT-специалистами для дистанционного устранения сетевых проблем. Но в руках злоумышленников они превращаются в инструмент мошенничества: проникнув в ваш компьютер, руткиты обеспечивают киберпреступникам возможность получить контроль над ним и похитить ваши данные или установить другие вредоносные программы. Руткиты умеют качественно маскировать свое присутствие в системе, чтобы оставаться незамеченными как можно дольше. Обнаружение такого вредоносного кода требует ручного мониторинга необычного поведения, а также регулярного внесения корректировок в программное обеспечение и операционную систему для исключения потенциальных маршрутов заражения.

8. Троянские программы

Более известные как троянцы, эти программы маскируются под легитимные файлы или ПО. После скачивания и установки они вносят изменения в систему и осуществляют вредоносную деятельность без ведома или согласия жертвы.

9. Баги

Баги - ошибки в фрагментах программного кода - это не тип вредоносного ПО, а именно ошибки, допущенные программистом. Они могут иметь пагубные последствия для вашего компьютера, такие как остановка, сбой или снижение производительности. В то же время баги в системе безопасности - это легкий способ для злоумышленников обойти защиту и заразить вашу машину. Обеспечение более эффективного контроля

безопасности на стороне разработчика помогает устранить ошибки, но важно также регулярного проводить программные корректировки, направленные на устранение конкретных багов.

Мифы и факты

Существует ряд распространенных мифов, связанных с компьютерными вирусами:

Любое сообщение об ошибке компьютера указывает на заражение вирусом. Это неверно: сообщения об ошибках также могут быть вызваны ошибками аппаратного или программного обеспечения.

Вирусам и червям всегда требуется взаимодействие с пользователем. Это не так. Для того чтобы вирус заразил компьютер, должен быть исполнен код, но это не требует участия пользователя. Например, сетевой червь может заражать компьютеры пользователей автоматически, если на них имеются определенные уязвимости.

Вложения к электронным письмам от известных отправителей являются безопасными. Это не так, потому что эти вложения могут быть заражены вирусом и использоваться для распространения заражения. Даже если вы знаете отправителя, не открывайте ничего, что в чем вы не уверены.

Антивирусные программы могут предотвратить заражение. Со своей стороны, поставщики антивирусного ПО делают все возможное, чтобы не отставать от разработчиков вредоносных программ, но пользователям обязательно следует установить на своем компьютере комплексное защитное решение класса Internet security, который включает в себя технологии, специально предназначенные для активного блокирования угроз. Даже при том, что 100-процентной защиты не существует. Нужно просто осознанно подходить к обеспечению собственной онлайн-безопасности, чтобы уменьшить риск подвергнуться атаке.

Вирусы могут нанести физический ущерб вашему компьютеру. Что если вредоносный код приведет к перегреву компьютера или уничтожит

критически важные микрочипы? Поставщики защитных решений неоднократно развенчивали этот миф - такие повреждения просто невозможны.

Между тем, рост количества устройств, взаимодействующих друг с другом в Интернете Вещей (IoT), открывает дополнительные интересные возможности: что если зараженный автомобиль съедет с дороги, или зараженная «умная» печь продолжит нагреваться, пока не случится превышение нормальной нагрузки? Вредоносного ПО будущего может сделать такой физический ущерб реальностью.

У пользователей есть ряд неправильных представлений о вредоносных программах: например, многие считают, что признаки заражения всегда заметны и поэтому они смогут определить, что их компьютер заражен. Однако, как правило, вредоносное ПО не оставляет следов, и ваша система не будет показывать каких-либо признаков заражения.

Так же не стоит верить, что все сайты с хорошей репутацией безопасны. Они также могут быть взломаны киберпреступниками. А посещение зараженного вредоносным кодом легитимного сайта – еще большая вероятность для пользователя расстаться со своей личной информацией. Именно это, как пишет SecurityWeek, произошло с Всемирным банком. Также многие пользователи считают, что их личные данные - фотографии, документы и файлы - не представляют интереса для создателей вредоносных программ. Киберпреступники же используют общедоступные данные для того, чтобы атаковать отдельных пользователей, или собрать информацию, которая поможет им создать фишинговые письма, чтобы проникнуть во внутренние сети организаций.

Стандартные методы заражения

Итак, как же происходит заражение компьютерными вирусами или вредоносными программами? Существует несколько стандартных способов. Это ссылки на вредоносные сайты в электронной почте или сообщениях в

социальных сетях, посещение зараженного сайта (известного как drive-by загрузка) и использование зараженного USB-накопителя на вашем компьютере. Уязвимости операционной системы и приложений позволяют злоумышленникам устанавливать вредоносное ПО на компьютеры. Поэтому для снижения риска заражения очень важно устанавливать обновления для систем безопасности, как только они становятся доступными.

Киберпреступники часто используют методы социальной инженерии, чтобы обманом заставить вас делать что-то, что угрожает вашей безопасности или безопасности вашей компании. Фишинговые сообщения являются одним из наиболее распространенных методов. Вы получаете на вид абсолютно легитимное электронное сообщение, в котором вас убеждают загрузить зараженный файл или посетить вредоносный веб-сайт. Цель хакеров - написать сообщение так, чтобы вы нашли его убедительным. Это может быть, например, предупреждение о возможном вирусном заражении или уведомление из вашего банка или сообщение от старого друга.

Конфиденциальные данные, такие как пароли, являются главной целью киберпреступников. Помимо использования вредоносных программ для перехвата паролей в момент их ввода, злоумышленники также могут собирать пароли с веб-сайтов и других компьютеров, которые они взломали. Вот почему так важно использовать уникальный и сложный пароль для каждой учетной записи. Он должен состоять из 15 и более символов, включающих буквы, цифры и специальные символы. Таким образом, если киберпреступникам удастся взломать один аккаунт, они не получат доступ ко всем вашим учетным записям. К сожалению, большинство пользователей имеют очень слабые пароли: вместо того, чтобы придумать труднодоступную комбинацию, они обращаются к standby-паролям типа «123456» или «Password123», которые преступники легко подбирают. Даже контрольные вопросы не всегда могут служить эффективной защитой, потому что многие люди дают один и тот же ответ на вопрос «Ваше любимая еда?», например,

если вы находитесь в Соединенных Штатах, то почти наверняка ответ будет - «Пицца».

Признаки заражения

Хотя большинство вредоносных программ не оставляет никаких явных следов, и ваш компьютер работает нормально, иногда все же можно заметить признаки возможного заражения. Самый первый из них - снижение производительности, т.е. процессы происходят медленные, загрузка окон занимает больше времени, в фоновом режиме работают какие-то случайные программы. Еще одним настораживающим признаком может считаться измененных домашних интернет-страниц в вашем браузере или более частое, чем обычно, появление всплывающих объявлений. В некоторых случаях вредоносное ПО даже может влиять на базовые функции компьютера: не открывается Windows, нет подключения к Интернету или доступа к более высокоуровневым функциям управления системой более высокого уровня. Если вы подозреваете, что ваш компьютер может быть заражен, немедленно произведите проверку системы. Если заражение не обнаружено, но вы все еще сомневаетесь, получите второе мнение - запустите альтернативный антивирусный сканер.

Антивирусное ПО

Антивирусное программное обеспечение создано для профилактики, выявления и уничтожения компьютерных вирусов. Способы обнаружения и лечения зараженных файлов могут быть разными. В любом случае при обнаружении заражения какого-либо файла антивирус пытается удалить из него вредоносный код и, если это сделать невозможно, удаляет файл полностью.

Типы антивирусного ПО

Сканеры. После запуска сканируют файловую систему и ОЗУ (оперативную память) ПК и нейтрализуют найденные вирусы.

Мониторы (сторожа). Отслеживают запускаемые на компьютере процессы в реальном времени.

Полифаги. Наиболее эффективные, универсальные решения. Сканируют запускаемые файлы и загрузочные секторы жестких дисков на предмет появления новых вирусов.

Блокировщики. Могут обнаружить компьютерный вирус на ранней стадии заражения ПК (при его записи в загрузочный сектор жесткого диска).

Ревизоры. Создают базу сведений о параметрах файлов и контролируют их изменение. Не могут находить вирусы в новых файлах, так как не имеют о них данных в своей базе.

Блокировщики часто входят в BIOS (Basic Input-Output system – базовая система ввода/вывода, которая хранится на микросхеме материнской платы). Полифаги наиболее «тяжеловесные», они занимают много места на диске и «съедают» большой объем оперативной памяти.

Разновидности защит

В зависимости от типа угрозы (известная или неизвестная конкретному ПО) антивирус может осуществить проактивную или реактивную защиту:

Проактивная защита (эвристика). Защита от неизвестных вирусов, основанная на изучении кода и поведения программ, характерных для вредоносного ПО. Такой тип защиты показывает лучший результат при борьбе с модифицированными вирусами. За основу принимаются данные об уже существующих угрозах. Эвристика в антивирусном контексте — набор правил, которые используются для обнаружения действий вредоносных программ без необходимости определения конкретной угрозы.

Реактивная защита (вирусная сигнатура). Защита от уже известных вирусов, основанная на информации о коде и остальных особенностях вредоносного ПО. Для максимально эффективной работы такие антивирусы должны постоянно обновлять свои базы данных вирусных сигнатур.Защита,

основанная на вирусных сигнатурах подразумевает обращение к словарю с уже известными вирусами, которые составили разработчики антивирусного ПО.

Главный недостаток проактивной защиты — так называемые «ложные срабатывания», частые блокировки незараженного программного обеспечения. Минус реактивной защиты — невозможность защититься от новых угроз. В современном антивирусном ПО используется и проактивная, и реактивная защита.

Как только антивирус обнаруживает вредоносный код, он может выполнить следующие действия (в зависимости от настроек пользователя):

Попытаться «вылечить» зараженный файл, убрав из него вредоносный код.

Отправить инфицированный файл в карантин. Актуально для ценных пользователю файлов. Находясь в карантине, зараженный файл не сможет навредить ПК; позже его можно вылечить самостоятельно или с помощью сторонних специалистов.

Удалить зараженный файл. Если исправить код не удалось, файл можно безвозвратно удалить с жесткого диска.

Не выполнять никаких действий. Если предполагается, что файл был помечен как «вредоносный» по ошибке, можно добавить этот файл в список исключений антивируса.

Полноценное антивирусное ПО защищает компьютер в режиме реального времени постоянно. То есть антивирус загружается вместе с ОС, всегда держит под контролем ОЗУ и файловую систему ПК, а также проводит мониторинг всех запускаемых и скачиваемых программ. Антивирусное программное обеспечение значительно снижает риск потери ценных данных, а также предотвращает попадание на ПК вредоносного ПО.

DNS

Что такое DNS?

Прежде чем начать говорить о DNS-серверах, расскажем о самой технологии DNS (Domain Name System). DNS — это технология, которая позволяет браузеру вроде Firefox, Chrome или Edge найти запрошенный пользователем сайт по его имени.

Как работает DNS?

Принцип работы DNS похож на поиск и вызов контактов из телефонной книги смартфона. Ищем имя, нажимаем «позвонить», и телефон соединяет нас с нужным абонентом. Понятно, что смартфон в ходе звонка не использует само имя человека, вызов возможен только по номеру телефона. Если вы внесете имя без номера телефона, позвонить человеку не сможете.

Так и с сайтом. Каждому имени сайта соответствует набор цифр формата 000.000.000.000. Этот набор называется IP-адресом, примером реального IP-адреса является 192.168.0.154 или 203.113.89.134. Когда пользователь вводит в адресной строке браузера имя сайта, например google.com, компьютер запрашивает IP-адрес этого сайта на специальном DNS-сервере и после получения корректного ответа открывает сам сайт.

Что такое DNS-сервер?

Это как раз и есть «книга контактов» интернета. DNS-сервер — это специализированный компьютер (или группа), который хранит IP-адреса сайтов. Последние, в свою очередь, привязаны к именам сайтов и обрабатывает запросы пользователя. В интернете много DNS-серверов, они есть у каждого провайдера и обслуживают их пользователей.

Зачем нужны DNS-серверы и какие они бывают?

Основное предназначение DNS-серверов — хранение информации о доменах и ее предоставление по запросу пользователей, а также кэширование

DNS-записей других серверов. Это как раз «книга контактов», о которой мы писали выше.

В случае кэширования все несколько сложнее. Дело в том, что отдельно взятый DNS-сервер не может хранить вообще всю информацию об адресах сайтов и связанных с ними IP-адресами. Есть исключения — корневые DNS-серверы, но о них позже. При обращении к сайту компьютера пользователя браузер первым делом проверяет локальный файл настроек DNS, файл hosts. Если там нет нужного адреса, запрос направляется дальше — на локальный DNS-сервер интернет-провайдера пользователя.

Локальный DNS-сервер в большинстве случаев взаимодействует с другими DNS-серверами из региона, в котором находится запрошенный сайт. После нескольких обращений к таким серверам локальный DNS-сервер получает искомое и отправляет эти данные в браузер — запрошенный сайт открывается. Полученные данные сохраняются на локальном сервере, что значительно ускоряет его работу. Поскольку, единожды «узнав» IP-адрес сайта, запрошенного пользователем, локальный DNS сохраняет эту информацию. Процесс сохранения полученных ранее данных и называется кэшированием.

Если пользователь обратится к ранее запрошенному сайту еще раз, то сайт откроется быстрее, поскольку используется сохраненная информация. Правда, хранится кэш не вечно, время хранения зависит от настроек самого сервера.

IP-адрес сайта может измениться — например, при переезде на другой хостинг или сервер в рамках прежнего хостинга. Что происходит в этом случае? В этом случае обращения пользователей к сайту, чей IP-адрес поменялся, некоторое время обрабатываются по-старому, то есть перенаправление идет на прежний «айпишник». И лишь через определенное время (например, сутки) кэш локальных серверов обновляется, после чего обращение к сайту идет уже по новому IP-адресу.

Где находятся главные DNS-серверы?

DNS-серверы верхнего уровня, которые содержат информацию о корневой DNS-зоне, называются корневыми. Этими серверами управляют разные операторы. Изначально корневые серверы находились в Северной Америке, но затем они появились и в других странах. Основных серверов — 13. Но, чтобы повысить устойчивость интернета в случае сбоев, были созданы запасные копии, реплики корневых серверов. Так, количество корневых серверов увеличилось с 13 до 123.

В Северной Америке находятся 40 серверов (32,5%), в Европе – 35 (28,5%), еще 6 серверов располагаются в Южной Америке (4,9%) и 3 – в Африке (2,4%). Если взглянуть на карту, то DNS-серверы расположены согласно интенсивности использования интернет-инфраструктуры. Есть сервера в Австралии, Китае, Бразилии, ОАЭ и других странах, включая Исландию.

В России тоже есть несколько реплик корневых серверов DNS, среди которых:

F.root (Москва);

I.root (Санкт-Петербург);

J.root (Москва, Санкт-Петербург);

K.root (Москва, Санкт-Петербург, Новосибирск);

L.root (Москва, Ростов-на-Дону, Екатеринбург).

Один из узлов корневого DNS-сервера K-root размещен в Selectel.

Что такое DNS-зоны?

В этой статье мы рассматриваем лишь вариант «один — один IP-адрес». На самом деле, ситуация может быть и сложнее. Так, с определенным доменным именем может быть связано несколько ресурсов — сайт и почтовый сервер. У этих ресурсов вполне могут быть разные IP-адреса, что дает возможность повысить надежность и эффективность работы сайта или

почтовой системы. Есть у сайтов и поддомены, IP-адреса которых тоже могут быть разными.

Вся эта информация о связи сайта, поддоменов, почтовой системы хранится в специальном файле на DNS-сервере. Его содержимое называется DNS-зона. Файл содержит следующие типы записей:

A — адрес веб-ресурса, который привязан к конкретному имени домена.

MX — адрес почтового сервера.

CNAME — чаще всего этот тип записи используется для подключения поддомена.

NS — адрес DNS-сервера, который отвечает за содержимое других ресурсных записей.

TXT — любая текстовая информация о доменном имени.

SPF — данные с указанием списка серверов, которые входят в список доверенных для отправки писем от имени указанного домена.

SOA — исходная запись зоны, в которой указаны сведения о сервере и которая содержит шаблонную информацию о доменном имени.

А что с новыми доменами?

После регистрации доменного имени нужно «рассказать» о нем DNS-серверам. Для этого нужно прописать ресурсные записи, что обычно делается в админке хостинг-провайдера или доменного провайдера. Примерно через сутки DNS-записи пропишутся в локальном сервере, также они попадут и в реестры всех прочих DNS-серверов. Как только это произойдет, новый домен станет нормально открываться браузером. «DNS сайта», как иногда ошибочно называют доменное имя, активируется.

Еще немного о DNS

Инфраструктура DNS-серверов, вернее, ее основа, была заложена в начале 1980-х годов. С тех пор менялась она лишь незначительно — например,

добавлялись новые доменные зоны. Так, в РФ в 2010 году появился кириллический домен .рф. До этого доменные имена могли быть лишь латинскими.

От DNS-инфраструктуры зависит нормальная работа всей глобальной сети, поэтому за работоспособностью серверов постоянно следят. В частности, предпринимаются меры по усилению безопасности системы. Кроме того, вводятся и меры на случай стихийных бедствий, проблем с электричеством и других экстренных ситуаций.

Ключевые характеристики DNS

DNS обладает следующими характеристиками:

Распределённость администрирования. Ответственность за разные части иерархической структуры несут разные люди или организации.

Распределённость хранения информации. Каждый узел сети в обязательном порядке должен хранить только те данные, которые входят в его зону ответственности, и (возможно) адреса корневых DNS-серверов.

Кэширование информации. Узел может хранить некоторое количество данных не из своей зоны ответственности для уменьшения нагрузки на сеть.

Иерархическая структура, в которой все узлы объединены в дерево, и каждый узел может или самостоятельно определять работу нижестоящих узлов, или делегировать (передавать) их другим узлам.

Резервирование. За хранение и обслуживание своих узлов (зон) отвечают (обычно) несколько серверов, разделённые как физически, так и логически, что обеспечивает сохранность данных и продолжение работы даже в случае сбоя одного из узлов.

DNS важна для работы Интернета, так как для соединения с узлом необходима информация о его IP-адресе, а для людей проще запоминать буквенные (обычно осмысленные) адреса, чем последовательность цифр IP-

адреса. В некоторых случаях это позволяет использовать виртуальные серверы, например, HTTP-серверы, различая их по имени запроса. Первоначально преобразование между доменными и IP-адресами производилось с использованием специального текстового файла hosts, который составлялся централизованно и автоматически рассылался на каждую из машин в своей локальной сети. С ростом Сети возникла необходимость в эффективном, автоматизированном механизме, которым и стала DNS.

DNS была разработана Полом Мокапетрисом в 1983 году; оригинальное описание механизмов работы содержится в RFC 882 и RFC 883. В 1987 публикация RFC 1034 и RFC 1035 изменила спецификацию DNS и отменила RFC 882, RFC 883 и RFC 973 как устаревшие.

Терминология и принципы работы

Ключевыми понятиями DNS являются:

Доме́н (англ. domain «область») — узел в дереве имён, вместе со всеми подчинёнными ему узлами (если таковые имеются), то есть именованная ветвь или поддерево в дереве имён. Структура доменного имени отражает порядок следования узлов в иерархии; доменное имя читается слева направо от младших доменов к доменам высшего уровня (в порядке повышения значимости): вверху находится корневой домен (имеющий идентификатор «.»(точка)), ниже идут домены первого уровня (доменные зоны), затем — домены второго уровня, третьего и т. д. (например, для адреса ru.wikipedia.org. домен первого уровня — org, второго — wikipedia, третьего — ru). DNS позволяет не указывать точку корневого домена.

Поддомен (англ. subdomain) — подчинённый домен (например, wikipedia.org — поддомен домена org, а ru.wikipedia.org — домена wikipedia.org). Теоретически такое деление может достигать глубины 127 уровней, а каждая метка может содержать до 63 символов, пока общая длина вместе с точками не достигнет 254 символов. Но на практике регистраторы

доменных имён используют более строгие ограничения. Например, если у вас есть домен вида mydomain.ru, вы можете создать для него различные поддомены вида mysite1.mydomain.ru, mysite2.mydomain.ru и т. д.

Ресурсная запись — единица хранения и передачи информации в DNS. Каждая ресурсная запись имеет имя (то есть привязана к определённому доменному имени, узлу в дереве имён), тип и поле данных, формат и содержание которого зависит от типа.

Зона — часть дерева доменных имён (включая ресурсные записи), размещаемая как единое целое на некотором сервере доменных имён (DNS-сервере, см. ниже), а чаще — одновременно на нескольких серверах (см. ниже). Целью выделения части дерева в отдельную зону является передача ответственности (см. ниже) за соответствующий домен другому лицу или организации. Это называется делегированием (см. ниже). Как связная часть дерева, зона внутри тоже представляет собой дерево. Если рассматривать пространство имён DNS как структуру из зон, а не отдельных узлов/имён, тоже получается дерево; оправданно говорить о родительских и дочерних зонах, о старших и подчинённых. На практике большинство зон 0-го и 1-го уровня ('.', ru, com, …) состоят из единственного узла, которому непосредственно подчиняются дочерние зоны. В больших корпоративных доменах (2-го и более уровней) иногда встречается образование дополнительных подчинённых уровней без выделения их в дочерние зоны.

Делегирование — операция передачи ответственности за часть дерева доменных имён другому лицу или организации. За счёт делегирования в DNS обеспечивается распределённость администрирования и хранения. Технически делегирование выражается в выделении этой части дерева в отдельную зону, и размещении этой зоны на DNS-сервере (см. ниже), управляемом этим лицом или организацией. При этом в родительскую зону включаются «склеивающие» ресурсные записи (NS и A), содержащие

указатели на DNS-сервера дочерней зоны, а вся остальная информация, относящаяся к дочерней зоне, хранится уже на DNS-серверах дочерней зоны.

DNS-сервер — специализированное ПО для обслуживания DNS, а также компьютер, на котором это ПО выполняется. DNS-сервер может быть ответственным за некоторые зоны и/или может перенаправлять запросы вышестоящим серверам.

DNS-клиент — специализированная библиотека (или программа) для работы с DNS. В ряде случаев DNS-сервер выступает в роли DNS-клиента.

Авторитетность (англ. authoritative) — признак размещения зоны на DNS-сервере. Ответы DNS-сервера могут быть двух типов: авторитетные (когда сервер заявляет, что сам отвечает за зону) и неавторитетные (англ. Non-authoritative), когда сервер обрабатывает запрос, и возвращает ответ других серверов. В некоторых случаях вместо передачи запроса дальше DNS-сервер может вернуть уже известное ему (по запросам ранее) значение (режим кеширования).

DNS-запрос (англ. DNS query) — запрос от клиента (или сервера) серверу. Запрос может быть рекурсивным или нерекурсивным (см. Рекурсия).

Система DNS содержит иерархию DNS-серверов, соответствующую иерархии зон. Каждая зона поддерживается как минимум одним авторитетным сервером DNS (от англ. authoritative — авторитетный), на котором расположена информация о домене.

Имя и IP-адрес не тождественны — один IP-адрес может иметь множество имён, что позволяет поддерживать на одном компьютере множество веб-сайтов (это называется виртуальный хостинг). Обратное тоже справедливо — одному имени может быть сопоставлено множество IP-адресов: это позволяет создавать балансировку нагрузки.

Для повышения устойчивости системы используется множество серверов, содержащих идентичную информацию, а в протоколе есть средства, позволяющие поддерживать синхронность информации, расположенной на

разных серверах. Существует 13 корневых серверов, их адреса практически не изменяются.

Протокол DNS использует для работы TCP- или UDP-порт 53 для ответов на запросы. Традиционно запросы и ответы отправляются в виде одной UDP-датаграммы. TCP используется, когда размер данных ответа превышает 512 байт, и для AXFR-запросов.

Рекурсия

Термином рекурсия в DNS обозначают алгоритм поведения DNS-сервера: «выполнить от имени клиента полный поиск нужной информации во всей системе DNS, при необходимости обращаясь к другим DNS-серверам».

DNS-запрос может быть рекурсивным — требующим полного поиска, и нерекурсивным (или итеративным) — не требующим полного поиска.

Аналогично — DNS-сервер может быть рекурсивным (умеющим выполнять полный поиск) и нерекурсивным (не умеющим выполнять полный поиск). Некоторые программы DNS-серверов, например, BIND, можно сконфигурировать так, чтобы запросы одних клиентов выполнялись рекурсивно, а запросы других — нерекурсивно.

При ответе на нерекурсивный запрос, а также при неумении или запрете выполнять рекурсивные запросы, DNS-сервер либо возвращает данные о зоне, за которую он ответственен, либо возвращает ошибку. Настройки нерекурсивного сервера, когда при ответе выдаются адреса серверов, которые обладают большим объёмом информации о запрошенной зоне, чем отвечающий сервер (чаще всего — адреса корневых серверов), являются некорректными, и такой сервер может быть использован для организации DoS-атак.

В случае рекурсивного запроса DNS-сервер опрашивает серверы (в порядке убывания уровня зон в имени), пока не найдёт ответ или не обнаружит, что домен не существует (на практике поиск начинается с

наиболее близких к искомому DNS-серверов, если информация о них есть в кэше и не устарела, сервер может не запрашивать другие DNS-серверы).

Рассмотрим на примере работу всей системы.

Предположим, мы набрали в браузере адрес ru.wikipedia.org. Браузер ищет соответствие этого адреса IP-адресу в файле hosts. Если файл не содержит соответствия, то далее браузер спрашивает у сервера DNS: «какой IP-адрес у ru.wikipedia.org»? Однако сервер DNS может ничего не знать не только о запрошенном имени, но и даже обо всём домене wikipedia.org. В этом случае сервер обращается к корневому серверу — например, 198.41.0.4. Этот сервер сообщает — «У меня нет информации о данном адресе, но я знаю, что 204.74.112.1 является ответственным за зону org.» Тогда сервер DNS направляет свой запрос к 204.74.112.1, но тот отвечает «У меня нет информации о данном сервере, но я знаю, что 207.142.131.234 является ответственным за зону wikipedia.org.» Наконец, тот же запрос отправляется к третьему DNS-серверу и получает ответ — IP-адрес, который и передаётся клиенту — браузеру.

В данном случае при разрешении имени, то есть в процессе поиска IP по имени:

браузер отправил известному ему DNS-серверу рекурсивный запрос — в ответ на такой тип запроса сервер обязан вернуть «готовый результат», то есть IP-адрес, либо пустой ответ и код ошибки NXDOMAIN;

DNS-сервер, получивший запрос от браузера, последовательно отправлял нерекурсивные запросы, на которые получал от других DNS-серверов ответы, пока не получил ответ от сервера, ответственного за запрошенную зону;

остальные упоминавшиеся DNS-серверы обрабатывали запросы нерекурсивно (и, скорее всего, не стали бы обрабатывать запросы рекурсивно, даже если бы такое требование стояло в запросе).

Иногда допускается, чтобы запрошенный сервер передавал рекурсивный запрос «вышестоящему» DNS-серверу и дожидался готового ответа.

При рекурсивной обработке запросов все ответы проходят через DNS-сервер, и он получает возможность кэшировать их. Повторный запрос на те же имена обычно не идёт дальше кэша сервера, обращения к другим серверам не происходит вообще. Допустимое время хранения ответов в кэше приходит вместе с ответами (поле TTL ресурсной записи).

Рекурсивные запросы требуют больше ресурсов от сервера (и создают больше трафика), так что обычно принимаются от «известных» владельцу сервера узлов (например, провайдер предоставляет возможность делать рекурсивные запросы только своим клиентам, в корпоративной сети рекурсивные запросы принимаются только из локального сегмента). Нерекурсивные запросы обычно принимаются ото всех узлов сети (и содержательный ответ даётся только на запросы о зоне, которая размещена на узле, на DNS-запрос о других зонах обычно возвращаются адреса других серверов).

Обратный DNS-запрос

Основная статья: Обратный просмотр DNS

DNS используется в первую очередь для преобразования символьных имён в IP-адреса, но он также может выполнять обратный процесс. Для этого используются уже имеющиеся средства DNS. Дело в том, что с записью DNS могут быть сопоставлены различные данные, в том числе и какое-либо символьное имя. Существует специальный домен in-addr.arpa, записи в котором используются для преобразования IP-адресов в символьные имена. Например, для получения DNS-имени для адреса 11.22.33.44 можно запросить у DNS-сервера запись 44.33.22.11.in-addr.arpa, и тот вернёт соответствующее символьное имя. Обратный порядок записи частей IP-адреса объясняется тем,

что в IP-адресах старшие биты расположены в начале, а в символьных DNS-именах старшие (находящиеся ближе к корню) части расположены в конце.

Записи DNS

Основная статья: Типы ресурсных записей DNS

Записи DNS, или ресурсные записи (англ. resource records, RR), — единицы хранения и передачи информации в DNS. Каждая ресурсная запись состоит из следующих полей:

имя (NAME) — доменное имя, к которому привязана или которому «принадлежит» данная ресурсная запись,

тип (TYPE) ресурсной записи — определяет формат и назначение данной ресурсной записи,

класс (CLASS) ресурсной записи; теоретически считается, что DNS может использоваться не только с TCP/IP, но и с другими типами сетей, код в поле класс определяет тип сети,

TTL (Time To Live) — допустимое время хранения данной ресурсной записи в кэше неответственного DNS-сервера,

длина поля данных (RDLEN),

поле данных (RDATA), формат и содержание которого зависит от типа записи.

Наиболее важные типы DNS-записей:

Запись A (address record) или запись адреса связывает имя хоста с адресом протокола IPv4. Например, запрос A-записи на имя referrals.icann.org вернёт его IPv4-адрес — 192.0.34.164.

Запись AAAA (IPv6 address record) связывает имя хоста с адресом протокола IPv6. Например, запрос AAAA-записи на имя K.ROOT-SERVERS.NET вернёт его IPv6-адрес — 2001:7fd::1.

Запись CNAME (canonical name record) или каноническая запись имени (псевдоним) используется для перенаправления на другое имя.

Запись MX (mail exchange) или почтовый обменник указывает сервер(ы) обмена почтой для данного домена.

Запись NS (name server) указывает на DNS-сервер для данного домена.

Запись PTR (pointer[4][5]) обратная DNS-запись или запись указателя связывает IP-адрес хоста с его каноническим именем. Запрос в домене in-addr.arpa на IP-адрес хоста в reverse-форме вернёт имя (FQDN) данного хоста (см. Обратный DNS-запрос). Например (на момент написания), для IP-адреса 192.0.34.164 запрос записи PTR 164.34.0.192.in-addr.arpa вернёт его каноническое имя referrals.icann.org. В целях уменьшения объёма нежелательной корреспонденции (спама) многие серверы-получатели электронной почты могут проверять наличие PTR-записи для хоста, с которого происходит отправка. В этом случае PTR-запись для IP-адреса должна соответствовать имени отправляющего почтового сервера, которым он представляется в процессе SMTP-сессии.

Запись SOA (Start of Authority) или начальная запись зоны указывает, на каком сервере хранится эталонная информация о данном домене, содержит контактную информацию лица, ответственного за данную зону, тайминги (параметры времени) кеширования зонной информации и взаимодействия DNS-серверов.

SRV-запись (server selection) указывает на серверы для сервисов, используется, в частности, для Jabber и Active Directory.

Зарезервированные доменные имена

Документ RFC 2606 (Reserved Top Level DNS Names — Зарезервированные имена доменов верхнего уровня) определяет названия доменов, которые следует использовать в качестве примеров (например, в документации), а также для тестирования. Кроме example.com, example.org и example.net, в эту группу также входят test, invalid и др.

Интернациональные доменные имена

Доменное имя может состоять только из ограниченного набора ASCII-символов, позволяя набрать адрес домена независимо от языка пользователя. ICANN утвердил основанную на Punycode систему IDNA, преобразующую любую строку в кодировке Unicode в допустимый DNS набор символов.

Настройка DNS в системе Debian

В debian следует установить пакет bind

 # apt-get install bind9

Отредактируйте файл /etc/bind/named.conf добавив в него описание вашей зоны (предположим, что домен называется example.com)

```
zone "example.com" {
        type master;
        file "/etc/bind/example.com";
};
```

Отредактируйте файл /etc/bind/example.com добавив в него строки

```
$TTL    3600
@ IN     SOA     ns1.example.com. root.example.com. (2012000001 10800 3600 604800 86400)
@  IN      NS      ns1
@  IN      NS      ns2
ns1     IN      A       10.10.10.10
ns2     IN      A       192.168.1.1
@       IN      A       10.10.10.10
www     IN      A       10.10.10.10
```

После чего можно перечитать конфигурацию DNS-сервера

 # rndc reload

После приведенной настройки можно указывать сервера имен ns1.example.com/10.10.10.10 и ns2.example.com/192.168.1.1 у регистратора. В биллинге BILLmanager следует указывать записи через символ /, у регистраторов может быть меню Child Nameservers где указывается данное соответствие.

Настройка DNS в системе Ubuntu

Пакет bind9 уже установлен в шаблоне ubuntu, его необходимо запустить и добавить в автозапуск

 # /etc/init.d/bind9 start

 # update-rc.d bind9 start 15 2 3 4 5 . stop 70 0 1 6 .

Дальнейшая настройка выполняется точно также как в случае настройки на Debian.

Настройка DNS в системе CentOS

DNS сервер уже предустановлен в шаблоне ПО centos. Его следует настроить и запустить.

Отредактируйте /etc/named.conf и удалите или закоментируйте символами // строки

listen-on port 53 { 127.0.0.1; };

allow-query { localhost; };

Следующий шаг (создание и настройка использование rndc.key) можно пропустить если не будете пользоваться командой rndc - в таком случае перезапускайте DNS-сервер командой /etc/init.d/named restart

Создайте ключ /etc/rndc.key командой

 # rndc-confgen -a

Добавьте в /etc/named.conf

```
key "rndc-key" {
    algorithm hmac-md5;
    secret "sUUony71pdPDHSZjNrkDFQ==";
};
controls {
     inet 127.0.0.1 port 953
     allow { 127.0.0.1; } keys { "rndc-key"; };
```

};

Строку secret

secret "sUUony71pdPDHSZjNrkDFQ==";

следует брать из файла /etc/rndc.key

Запустите named и добавьте его в автозапуск

/etc/init.d/named start

 # chkconfig named --level 2345 on

Настройка зоны производится аналогично настройке на Debian / Ubuntu со следующими отличиями:

Файл конфигурации DNS-сервера /etc/named.conf

Зоны расположены в каталоге /var/named/

Настройка DNS в Windows 2008

Зайдите в Server manager - Roles и добавьте там (Add roles) DNS-сервер.

Далее в Start - Administrative Tools - DNS Manager - Forward Lookup Zones добавьте зону и в мастере создания зоны выберите Primary zone - Do not allow dynamic updates

После чего в том же приложении настраивается зона - создаются записи NS (вначале добавляется в форме IP-адрес, затем запись) и создаются записи типа A ns1 и ns2 на соответствующие IP-адреса.

Протокол DHCP

DHCP (Dynamic Host Configuration Protocol – протокол динамической настройки узла) – сетевой протокол, позволяющий сетевым устройствам автоматически получать IP-адрес и другие параметры, необходимые для работы в сети TCP/IP. Данный протокол работает по модели «клиент-сервер». Для автоматической конфигурации компьютер-клиент, на этапе конфигурации сетевого устройства, обращается к серверу DHCP, и получает от него нужные параметры. Системный администратор может задать диапазон адресов, распределяемых сервером среди компьютеров. Это позволяет избежать ручной настройки компьютеров сети и уменьшить количество ошибок. Протокол DHCP используется в большинстве сетей TCP/IP.

DHCP является расширением протокола BOOTP, использовавшегося ранее для обеспечения бездисковых рабочих станций IP-адресами при их загрузке. DHCP сохраняет

обратную совместимость с BOOTP.

Протокол DHCP предоставляет три способа распределения IP-адресов:

Ручное распределение. При этом способе, сетевой администратор сопоставляет аппаратному адресу (для Ethernet сетей это MAC-адрес), каждого клиентского компьютера, определённый IP-адрес. Фактически, данный способ распределения адресов, отличается от ручной настройки каждого компьютера лишь тем, что сведения об адресах хранятся централизованно на сервере DHCP, и потому их проще изменять при необходимости.

Автоматическое распределение. При данном способе, каждому компьютеру на постоянное использование выделяется произвольный свободный IP-адрес из определённого администратором диапазона.

Динамическое распределение. Этот способ аналогичен автоматическому распределению за исключением того, что адрес выдаётся компьютеру не на постоянное пользование, а на определённый срок. Это называется арендой адреса. По истечении срока аренды IP-адрес вновь считается свободным, и клиент обязан запросить новый. Кроме того, клиент сам может отказаться от полученного адреса.

Протокол DHCP является клиент-серверным, то есть в его работе участвуют клиент DHCP и сервер DHCP. Передача данных производится при помощи протокола UDP. По умолчанию запросы от клиента делаются на 67 порт к серверу, сервер в свою очередь отвечает на порт 68 к клиенту, выдавая адрес IP и другую необходимую информацию, такую, как сетевую маску, маршрутизатор и серверы DNS.

Все сообщения протокола DHCP разбиваются на поля, каждое из которых содержит определённую информацию. Все поля, кроме последнего имеют фиксированную длину. Описание полей указано в таблице 1.1.

Таблица 1.1 – поля протокола DHCP

Поле	Описание	Длина (в байтах)
1	2	3
op	Тип сообщения. Например может принимать значения: BOOTREQUEST (0x01, запрос от клиента к серверу) и BOOTREPLY (0x02, ответ от сервера к клиенту).	1
htype	Тип аппаратного адреса. Допустимые значения этого поля определены в RFC 1700 «Assigned Numbers». Например, для MAC-адреса Ethernet это поле принимает значение 0x01.	1
hlen	Длина аппаратного адреса в байтах. Для MAC-адреса Ethernet – 0x06.	1
hops	Количество промежуточных маршрутизаторов (так называемых агентов ретрансляции DHCP), через которые прошло сообщение. Клиент устанавливает это поле в 0x00.	1
xid	Уникальный идентификатор транзакции в 4 байта, генерируемый клиентом в начале процесса получения адреса.	4
secs	Время в секундах с момента начала процесса получения адреса. Может не использоваться (в этом случае оно устанавливается в 0x0000).	2

flags	Поле для флагов – специальных параметров протокола DHCP.	2
ciaddr	IP-адрес клиента. Заполняется только в том случае, если клиент уже имеет собственный IP-адрес и способен отвечать на запросы ARP (это возможно, если клиент выполняет процедуру обновления адреса по истечении срока аренды).	4
yiaddr	Новый IP-адрес клиента, предложенный сервером.	4
siaddr	IP-адрес сервера. Возвращается в предложении DHCP (см. ниже).	4
giaddr	IP-адрес агента ретрансляции, если таковой участвовал в процессе доставки сообщения DHCP до сервера.	4
chaddr	Аппаратный адрес (обычно MAC-адрес) клиента.	16
sname	Необязательное имя сервера в виде нуль-терминированной строки.	64
file	Необязательное имя файла на сервере, используемое бездисковыми рабочими станциями при удалённой загрузке. Как и sname, представлено в виде нуль-терминированной строки.	128
options	Поле опций DHCP. Здесь указываются различные дополнительные параметры конфигурации. В начале этого поля указываются четыре особых байта со значениями 99, 130, 83, 99 («волшебные числа»), позволяющие серверу определить наличие этого поля. Поле имеет переменную длину, однако DHCP-клиент должен быть готов принять DHCP-сообщение длиной в 576 байт (в этом сообщении поле options имеет длину 340 байт).	переменная

Рассмотрим пример процесса получения IP-адреса клиентом от сервера DHCP. Предположим, клиент ещё не имеет собственного IP-адреса, но ему известен его предыдущий адрес – 192.168.1.100. Процесс состоит из четырёх этапов. Эти этапы часто сокращаются как DORA (Discovery, Offer, Request, и Acknowledgement).

Обнаружение DHCP. В начале клиент выполняет широковещательный запрос по всей физической сети с целью обнаружить доступные DHCP-серверы. Он отправляет сообщение типа DHCPDISCOVER, при этом в качестве IP-адреса источника указывается 0.0.0.0 (если компьютер ещё не имеет собственного IP-адреса), а в качестве адреса назначения – широковещательный адрес 255.255.255.255.

Клиент заполняет несколько полей сообщения начальными значениями:

– В поле xid помещается уникальный идентификатор транзакции, который позволяет отличать данный процесс получения IP-адреса от других, протекающих в то же время;

– В поле chaddr помещается аппаратный адрес (MAC-адрес) клиента;

– В поле опций указывается последний известный клиенту IP-адрес. В данном примере это 192.168.1.100. Это необязательно и может быть проигнорировано сервером.

Сообщение DHCPDISCOVER может быть распространено за пределы локальной

физической сети при помощи специально настроенных агентов ретрансляции DHCP, перенаправляющих поступающие от клиентов сообщения DHCP серверам в других подсетях.

Не всегда процесс получения IP адреса начинается с DHCPDISCOVER. В случае, если клиент ранее уже получал IP адрес и срок его аренды ещё не прошёл – клиент может пропустить стадию DHCPDISCOVER начав с запроса DHCPREQUEST отправляемого с идентификатором сервера, который выдал адрес в прошлый раз. В случае же отсутствия ответа от DHCP сервера, выдавшего настройки в прошлый раз, клиент отправляет DHCPDISCOVER. Тем самым, клиент начинает процесс получения с начала, обращаясь уже ко всем DHCP серверам в сегменте сети.

Предложение DHCP. Получив сообщение от клиента, сервер определяет требуемую конфигурацию клиента в соответствии с указанными сетевым администратором настройками. В данном случае DHCP-сервер согласен с запрошенным клиентом адресом 192.168.1.100. Сервер отправляет ему ответ (DHCPOFFER), в котором предлагает конфигурацию. Предлагаемый клиенту IP-адрес указывается в поле yiaddr. Прочие параметры (такие, как адреса маршрутизаторов и DNS-серверов) указываются в виде опций в соответствующем поле.

Это сообщение DHCP-сервер отправляет хосту, пославшему DHCPDISCOVER, на его MAC, при определенных обстоятельствах сообщение может распространяться как широковещательная рассылка. Клиент может получить несколько различных предложений DHCP от разных серверов, из них он должен выбрать то, которое его «устраивает».

Запрос DHCP. Выбрав одну из конфигураций, предложенных DHCP-серверами, клиент отправляет запрос DHCP (DHCPREQUEST). Он рассылается широковещательно, при этом к опциям, указанным клиентом в сообщении DHCPDISCOVER, добавляется специальная опция – идентификатор сервера, указывающая адрес DHCP-сервера, выбранного клиентом.

Подтверждение DHCP. Наконец, сервер подтверждает запрос и направляет это подтверждение (DHCPACK) клиенту. После этого клиент должен настроить свой сетевой интерфейс, используя предоставленные опции.

После того как микропрограмма сетевой карты запустилась, она посылает широковещательный запрос, с целью найти DHCP сервер, который предоставит все вышеописанные настройки. Это означает, что в сети должен присутствовать DHCP сервер. При наличии DHCP сервера, и получении от него ответа, микропрограмма сетевой карты примет переданные ей настройки. А именно установит сетевые параметры (IP адрес, маску подсети, шлюз), после чего попытается подключится к серверу загрузки по протоколу TFTP и загрузить с него указанный исполняемый файл. То есть, кроме DHCP сервера в сети еще должен присутствовать TFTP сервер.

При наличии TFTP сервера, и присутствии на нем указанного файла, микропрограмма сетевой карты загрузит и попытается его выполнить. Если данный файл действительно исполняемый, то дальнейшая загрузка будет продолжаться согласно инструкциям, прописанным в данном файле. Обычно данным файлом является загрузчик (bootmgr, grub4dos, syslinux, pxelinux, efigrub) которому будет передано дальнейшее управление.

Настройка DHCP на Linux

Пакет DHCP-сервера доступен в официальных репозиториях основных дистрибутивов Linux, его установка довольно проста, просто выполните следующую команду:

 # yum install dhcp ***#CentOS***

$ sudo apt install isc-dhcp-server ***#Ubuntu***

После завершения установки настройте интерфейс, на котором вы хотите, чтобы демон DHCP обслуживал запросы, в файле конфигурации /etc/default/isc-dhcp-server или /etc/sysconfig/dhcpd.

 # vim /etc/sysconfig/dhcpd ***#CentOS***

$ sudo vim /etc/default/isc-dhcp-server ***#Ubuntu***

Например, если вы хотите, чтобы демон DHCPD прослушивал eth0, установите его с помощью следующей настройки.

DHCPDARGS=”eth0”

Сохраните файл и выйдите.

Основной файл конфигурации DHCP находится по адресу /etc/dhcp/dhcpd.conf, который должен содержать настройки того, что делать, где делать и все сетевые параметры, предоставляемые клиентам.

Этот файл в основном состоит из списка операторов, сгруппированных в две широкие категории:

Глобальные параметры: укажите, выполнять ли задачу, как выполнять задачу или какие параметры конфигурации сети предоставить DHCP-клиенту.

Объявления: определить топологию сети, указать состояние клиентов, предложить адреса для клиентов или применить группу параметров к группе объявлений.

Теперь откройте и отредактируйте файл конфигурации для настройки вашего DHCP-сервера.

---------- CentOS ----------

 # cp /usr/share/doc/dhcp-4.2.5/dhcpd.conf.example /etc/dhcp/dhcpd.conf

 # vi /etc/dhcp/dhcpd.conf

---------- Ubuntu ----------

$ sudo vim /etc/dhcp/dhcpd.conf

Начните с определения глобальных параметров, которые являются общими для всех поддерживаемых сетей, в верхней части файла. Они будут применяться ко всем объявлениям:

```
option domain-name "merionet.ru";
option domain-name-servers ns1.merionet.ru, ns2.merionet.ru;
default-lease-time 3600;
max-lease-time 7200;
authoritative;
```

Затем вам необходимо определить диапазон для внутренней подсети и дополнительные настройки:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    option routers              192.168.1.1;
    option subnet-mask          255.255.255.0;
    option domain-search         " merionet.ru ";
    option domain-name-servers     192.168.1.1;
    range   192.168.10.10   192.168.10.100;
    range   192.168.10.110   192.168.10.200;
```

}

Тут:

subnet – сеть, в которой будут работать настройки;

option routers – шлюз по-умолчанию;

option subnet-mask – маска подсети;

range – диапазон IP-адресов;

option domain-name-servers – DNS-сервера;

option domain-name – суффикс доменного имени;

option broadcast-address — адрес сети для широковещательных запросов;

default-lease-time, max-lease-time — время и максимальное время в секундах, на которое DHCP-клиент получит адрес;

Обратите внимание, что хосты, которым требуются специальные параметры конфигурации, могут быть перечислены в инструкциях хоста в справке.

man dhcp-options

Теперь, когда вы настроили демон DHCP-сервера, вам нужно запустить службу на некоторое время и включить ее автоматический запуск при следующей загрузке системы, а также проверить, работает ли она, используя следующие команды.

---------- CentOS ----------

 # systemctl start dhcpd

 # systemctl enable dhcpd

 # systemctl enable dhcpd

---------- Ubuntu ----------

$ sudo systemctl start isc-dhcp-server

$ sudo systemctl enable isc-dhcp-server

$ sudo systemctl enable isc-dhcp-server

Затем разрешите выполнение запросов к демону DHCP в брандмауэре, который прослушивает порт 67/UDP, запустив его.

---------- CentOS ----------

 # firewall-cmd --zone=public --permanent --add-service=dhcp

 # firewall-cmd --reload

#---------- Ubuntu ----------

$ sudo ufw allow 67/udp

$ sudo ufw reload

Настройка DHCP на Windows Server

1. Откройте окно диспетчера сервера, выберите пункт Добавить роли и компоненты.

2. В появившемся окне нажмите Далее.

3. Выберите пункт Установка ролей и компонентов, затем нажмите Далее.

4. Выберите сервер на который будет производиться установка роли, затем нажмите Далее.

5. Поставьте галочку напротив DHCP-сервер.

6. Мастер установки ролей предупредит, что для установки роли DHCP-сервера нужно установить несколько компонентов. Нажмите Добавить компоненты.

7. Убедитесь, что после установки необходимых компонентов напротив DHCP-сервер стоит галочка, затем нажмите Далее.

8. На этапе добавления компонентов оставьте все значения по умолчанию и нажмите

Далее.

9. Ознакомьтесь с дополнительной информацией касательно DHCP-сервера, затем нажмите Далее.

10. Для начала установки роли нажмите Установить.

11. После окончания установки нажмите Завершение настройки DHCP.

12. Мастер настройки уведомит о том, что далее будут созданы две локальные группы безопасности для управления доступом к DHCP-серверу, а затем будет произведена авторизация DHCP-сервера в Active Directory. Нажмите Далее.

13. В разделе Использовать учётные данные следующего пользователя укажите учетную запись с правами администратора домена, затем нажмите Фиксировать.

14. Мастер настройки DHCP проинформирует об успешном создании групп безопасности и авторизации DHCP-сервера. Нажмите Закрыть.

15. В Мастере добавления ролей и компонентов нажмите Закрыть.

16. Откройте Диспетчер серверов, затем Средства и выберите пункт DHCP.

17. Нажмите правой кнопкой мыши на IPv4, затем в открывшемся меню выберите Создать область….

18. В открывшемся окне Мастера создания области нажмите Далее.

19. В поле Имя укажите имя для нового диапазона адресов (прим. в данном руководстве это Local-net), затем нажмите Далее.

20. Укажите необходимый диапазон IP-адресов и введите в поля Начальный IP-адрес и Конечный IP-адрес (прим. в данном руководстве это 192.168.0.150 — 192.168.0.200, т.е. 50 IP-адресов). В параметрах конфигурации, распространяемых DHCP-клиенту в соответствующем поле введите маску подсети (прим. в данном руководстве она выбрана по умолчанию 255.255.255.0), затем нажмите Далее.

21. При необходимости можно указать диапазон, для которого DHCP-сервер не будет раздавать настройки. Это может пригодиться, если Вы знаете, что в определенном диапазоне адресов находятся серверы, принтеры или другие устройства, которым уже присвоен статический IP-адрес. В таком случае нужно исключить эту часть диапазона, так как IP-адреса из него уже используются. Также нужно исключить IP-адрес, который присвоен шлюзу (прим. в данном руководстве, в качестве примера исключены адреса с 192.168.0.190 по 192.168.0.200, т.е. 10 адресов). После того как Вы выбрали необходимый диапазон нажмите Добавить, затем нажмите Далее.

22. Далее можно выбрать на какое время IP-адреса будут выдаваться устройствам в аренду. Оставьте настройки без изменений и нажмите Далее.

23. Теперь необходимо указать сетевые настройки (шлюз, DNS), которые DHCP-сервер будет раздавать для устройств в локальной сети. Выберите Да, настроить эти параметры сейчас, затем нажмите Далее.

24. В поле IP-адрес укажите IP-адрес вашего маршрутизатора и нажмите Добавить, затем нажмите Далее.

25. В поле Родительский домен укажите доменное имя (прим. в данном руководстве это example.local), в поле IP-адрес введите адрес DNS-сервера (прим. это 192.168.0.104) и нажмите Добавить, затем нажмите Далее.

26. Т.к. в данном руководстве WINS-серверы не рассматриваются — нажмите Далее.

27. Для активации выбранного диапазона IP-адресов выберите Да, я хочу активировать эту область сейчас, затем нажмите Далее.

28. Настройка DHCP-сервера завершена. Теперь все устройства, подключаемые к локальной сети, будут получать сетевые настройки (IP-адрес, маска подсети, шлюз, DNS) и смогут взаимодействовать друг с другом. Нажмите Готово.

Интернет-шлюз

Сетевой шлюз конвертирует протоколы одного типа физической среды в протоколы другой физической среды (сети). Например, при соединении локального компьютера с сетью Интернет обычно используется сетевой шлюз.

Маршрутизатор (он же — роутер) является одним из примеров аппаратных сетевых шлюзов.

Сетевые шлюзы работают на всех известных операционных системах. Основная задача сетевого шлюза — конвертировать протокол между сетями. Роутер сам по себе принимает, проводит и отправляет пакеты только среди сетей, использующих одинаковые протоколы. Сетевой шлюз может с одной стороны принять пакет, сформатированный под один протокол (например Apple Talk) и конвертировать в пакет другого протокола (например TCP/IP) перед отправкой в другой сегмент сети. Сетевые шлюзы могут быть аппаратным решением, программным обеспечением или тем и другим вместе, но обычно это программное обеспечение, установленное на роутер или компьютер. Сетевой шлюз должен понимать все протоколы, используемые роутером. Обычно сетевые шлюзы работают медленнее, чем сетевые мосты, коммутаторы и обычные маршрутизаторы. Сетевой шлюз — это точка сети, которая служит выходом в другую сеть. В сети Интернет узлом или конечной точкой может быть или сетевой шлюз, или хост. Интернет-пользователи и компьютеры, которые доставляют веб-страницы пользователям — это хосты, а узлы между различными сетями — это сетевые шлюзы. Например, сервер, контролирующий трафик между локальной сетью компании и сетью Интернет — это сетевой шлюз.

В крупных сетях сервер, работающий как сетевой шлюз, обычно интегрирован с прокси-сервером и межсетевым экраном. Сетевой шлюз часто

объединен с роутером, который управляет распределением и конвертацией пакетов в сети.

Сетевой шлюз может быть специальным аппаратным роутером или программным обеспечением, установленным на обычный сервер или персональный компьютер. Большинство компьютерных операционных систем использует термины, описанные выше. Компьютеры под Windows обычно используют встроенный мастер подключения к сети, который по указанным параметрам сам устанавливает соединение с локальной или глобальной сетью. Такие системы могут также использовать DHCP-протокол. Dynamic Host Configuration Protocol (DHCP) — это протокол, который обычно используется сетевым оборудованием, чтобы получить различные данные, необходимые клиенту для работы с протоколом IP. С использованием этого протокола добавление новых устройств и сетей становится простым и практически автоматическим.

Шлюз рассматривается как сетевое устройство, которое действует как точка входа из одной сети в другую. Это соединение, которое соединяет два компьютера с Интернетом. Шлюз действует как портал между двумя программами и как средство связи между протоколами, которое позволяет им обмениваться данными на одних и тех же компьютерных устройствах или между различными компьютерными системами. С помощью шлюза пользователи смогут общаться и отправлять данные туда-сюда. Интернет не будет иметь никакого смысла без шлюзов. Шлюз также относится к узлу в компьютерной сети. Узел - это просто физическое место, куда данные останавливаются для транспортировки, чтения или использования. Узлы могут быть:

Компьютер, управляющий трафиком, который получает интернет-провайдер или провайдер услуг Интернета.

В интернете узел, являющийся остановкой, может быть шлюзом или узлом хоста.

Если вся семья имеет доступ к Интернету и в доме есть беспроводная сеть, то шлюз - это модем, который ISP предоставляет для подключения к сети.

Шлюз создается на границе сети таким образом, что он будет управлять всей передачей данных, которая маршрутизируется из сети внутри или снаружи. Шлюз также обладает информацией о внутренних путях хост-сети и изученных путях различных удаленных сетей. Кроме того, он работает в качестве маршрута доступа к другой сети. Шлюз является обязательной характеристикой всех маршрутов, даже если другие устройства могут также работать в качестве шлюза или узла.

Как правило, шлюз действует как защита для всех локальных сетей и соединяет локальные сети с сетями общего пользования. Он обеспечивает безопасность, функционируя подобно брандмауэру, с помощью NAT (Network Address Translation - трансляция сетевых адресов). Шлюз получает пакеты из локальной сети и заменяет внешний IP-адрес и новый адрес порта в ресурсные поля заголовков IP (Internet Protocol) и UDP (User Data Protocol).

Ниже перечислены различные функции шлюза:

Это лучший вариант для достижения мультимедийного взаимодействия между различными сетями, потому что каждая сеть имеет свои протоколы и характеристики.

Он работает в качестве интерфейса между локальными и глобальными протоколами, такими как TCP/IP в Интернете.

Это ключевой механизм любой телефонной связи, являющийся мостом между телефонной сетью и Интернетом.

Типы

Шлюзы могут принимать различные формы и выполнять несколько задач, например, следующие:

Облачный шлюз хранения данных

Данное сетевое устройство имеет форму программного или аппаратного обеспечения, обеспечивающего связь и трансляцию протоколов

между провайдером услуг облачного хранения данных и локальным клиентским приложением. Этот шлюз также известен как контроллер облачного хранения данных или облачное устройство хранения данных.

Шлюз безопасности электронной почты

Этот шлюз помогает защитить и предотвратить передачу сообщений электронной почты, которые нарушают политику компании, рассылают вредоносное ПО и передают данные с намерениями злоумышленников. Это предотвращает потерю данных, выполняет шифрование электронной почты и защищает устройство от известных и неизвестных вредоносных программ.

Медиа-шлюз

Это переводческое устройство, которое используется для преобразования различных типов цифровых медиапротоколов с целью обеспечения эффективной мультимедийной связи. Его основная функция заключается в преобразовании нескольких методов кодирования и передачи, которые позволят осуществлять связь между сетями.

Брандмауэр веб-приложений

Этот шлюз помогает защитить все веб-приложения посредством фильтрации и мониторинга HTTP-трафика между веб-приложением и Интернетом. Обычно это защищает веб-приложения от различных атак. Фильтрует и проверяет входящий трафик, который будет рассматриваться как потенциальная угроза и содержит вредоносные действия.

Amazon API шлюз

Этот вид шлюза публикует, создает, контролирует, защищает и обслуживает REST и API веб-сокетов в любом масштабе. Он выполняет все задачи, связанные с приемом и обработкой сотен тысяч одновременных вызовов API, управлением трафиком, авторизацией и контролем доступа.

API, SOA или XML шлюз

Он управляет потоком трафика, поступающего и уходящего из сервиса, микро-услугами ориентированной архитектуры или веб-службой на базе XML.

Шлюз Интернета вещей

IoT, или также известный как Интернет вещей, является шлюзом, который служит в качестве точки соединения между облаком и контроллерами, датчиками и интеллектуальными устройствами.

Магистральный шлюз VoIP

Это интерфейс, который облегчает использование старой телефонной связи. Этот шлюз соединяет абонентов с сетью VoIP без участия операторов и без взимания платы с телефонной компании.

История

В 1970 году исследователи из Соединенных Штатов и Франции попытались изучить возможности объединения различных типов пакетных сетей. Разработчики сети хотели создать межсетевые устройства, которые могли бы поддерживать несколько протоколов, обеспечивать динамическую маршрутизацию пакетов вокруг изменяющихся условий линии и гарантировать, что пакеты могут быть направлены по каналам с ограниченной скоростью. В 1987 и 1988 годах межсетевое взаимодействие расширилось, а названия "Шлюз" и "Маршрутизатор" стали более понятными. Наблюдатели отрасли были взволнованы быстрыми темпами развития межсетевого взаимодействия и перспективными рынками мостов, маршрутизаторов и шлюзов.

Общие вопросы

Сравнение маршрутизатора и шлюза поможет прояснить распространенное заблуждение между этими двумя терминами. И шлюз, и маршрутизатор используются для регулирования сетевого трафика между двумя или более отдельными сетями. Однако шлюзы регулируют трафик между двумя разными сетями, в отличие от маршрутизаторов, которые

регулируют трафик между похожими сетями. Шлюзы могут использоваться для снижения сетевого трафика. Он также чаще всего используется для обеспечения связи в различных средах, в то время как маршрутизаторы могут быть использованы для сегментирования трафика в корпоративных сетях.

Настройка интернет шлюза на Linux Debian

Подготовка шлюза

Для начала нужно обновить программное обеспечение на будущем сервере:

 # apt-get update

 # apt-get upgrade

Настройка маршрутизации, firewall и nat

Первым делом включим маршрутизацию пакетов между сетевыми интерфейсами. Для этого редактируем конфиг /etc/sysctl.conf:

 # mcedit /etc/sysctl.conf

net.ipv4.ip_forward=1

Либо раскомментируйте эту строку, либо добавьте, если ее нет. Но она по-умолчанию быть должна, закомментированная со значением 1. Применяем эту настройку:

 # sysctl -p

На выходе работы команды в консоли будет выведен измененный параметр со значением 1.

Теперь приступаем к самому главному - настройке фаервола iptables и nat в нем для обеспечения выхода в интернет из локальной сети. Я очень подробно рассмотрел эту тему в отдельной статье. Хотя там речь идет о другом дистрибутиве, сами правила iptables абсолютно одинаковые с точностью до строчки, за исключением маленького нюанса, связанного с тем, что правила нужно сохранять в другой файл для применения их после перезагрузки.

Я приведу здесь сразу готовый вариант файла с правилами iptables, необходимых для работы интернет шлюза в debian. В файле даны подробные комментарии ко всем значениям, так что вы без проблем разберетесь и закомментируете или наоборот раскомментируете необходимые вам значения. Качаем скрипт правил iptables - iptables-debian.sh

Копируем содержимое файла и создаем скрипт с правилами на сервере:

 # mcedit /etc/iptables.sh

Вставляем в редактор правила. Редактируем их под свои нужды, обязательно заменяя переменные WAN и LAN на свои. Сохраняем файл.

Прежде чем двигаться дальше предупреждаю, что все работы по настройке фаервола должны производиться только если у вас есть доступ к консоли сервера, чтобы в случае ошибки и потери удаленного доступа вы смогли откатить изменения. Даже если вы абсолютно уверены в своих знаниях, вас может подвести банальная ошибка или опечатка. Я сам, к сожалению, сталкивался с такими ситуациями, поэтому считаю необходимым предупредить об этом вас.

Делаем файл с правилами, исполняемым:

 # chmod 0740 /etc/iptables.sh

Прежде чем применить новые правила, посмотрим на текущие:

 # iptables -L -v -n

Теперь применим новые правила и посмотрим на результат:

 # /etc/iptables.sh

Теперь сделаем так, чтобы новые правила применялись после перезагрузки. В последней строчке скрипта есть команда:

/sbin/iptables-save > /etc/iptables.rules

С ее помощью готовый набор правил iptables выгружаются в файл. Нам нужно сделать так, чтобы эти правила применялись при включении сетевого

интерфейса во время загрузки сервера. Для этого открываем файл interfaces на редактирование и добавляем в самый конец строчку:

 # mcedit /etc/network/interfaces

post-up iptables-restore < /etc/iptables.rules

Для проверки перезагружаем шлюз и проверяем, все ли в порядке. По сути основная настройка программного роутера на debian завершена. Осталось сделать небольшое дополнение и настроить dhcp и dns сервер в локальной сети. Я для этих целей использую простой и легкий в настройке dnsmasq.

Установка и настройка dnsmasq в Debian

Выполним установку dnsmasq на дебиан:

 # apt-get install -y dnsmasq

Сделаем минимальную настройку программы. Нам нужно просто выдавать сетевые настройки пользователям. Для этого приводим конфигурационный файл dnsmasq к следующему виду:

 # mcedit /etc/dnsmasq.conf

domain-needed

bogus-priv

interface=eth1

dhcp-range=eth1,10.0.15.50,10.0.15.150,24h

В данном случае мы будем выдавать пользователям ip адреса в диапазоне от 10.0.15.50 до 150. Сохраняем конфиг, добавляем программу в автозагрузку и запускаем.

 # insserv dnsmasq

 # /etc/init.d/dnsmasq start

Теперь можно запускать компьютер пользователя локальной сети, получать сетевые настройки по dhcp и проверять работу интернет шлюза.

Посмотреть выданные leases можно в файле /var/lib/misc/dnsmasq.leases. На этом настройка интернет шлюза на debian

закончена. Все что нужно для обеспечения доступа в интернет из локальной сети сделано. Получился программный роутер с широкими возможностями по наращиванию функционала.

Просмотр загрузки сети с помощью iftop

Теперь представим ситуацию, что кто-то забил весь интернет канал и вам надо быстро выяснить, кто это сделал. По-умолчанию, никаких подручных и удобных средств на шлюзе для этого нету. Установим одно из таких средств - программу iftop. Это простая консольная утилита, которая дает возможность оперативно посмотреть статистику загруженности сетевого интерфейса в реальном времени.

Устанавливаем iftop на debian:

 # apt-get install -y iftop

Для просмотра активности сетевого интерфейса, запускаем утилиту, указывая необходимый ключ:

 # iftop -i eth1

Чтобы увидеть порты, по которым идет трафик, добавляем ключ -P:

 # iftop -i eth1 –P

Заключение

Вот так легко и быстро можно настроить роутер, маршрутизатор или шлюз в интернет. Названия разные, а суть одна. В данном случае использовали операционную систему Debian, но схожий функционал легко организовать на Freebsd или CentOS. Для решения текущей задачи разница в работе не будет заметна. Каждый выбирает то, что больше нравится и к чему привык.

Было проделано следующее:

- Подготовили сервер Debian к настройке шлюза.
- Настроили маршрутизацию, iptables, нат. Проверили, что весь функционал восстанавливается после перезагрузки.

- Установили и настроили простой dhcp сервер и кэширующий dns сервер - dnsmasq. С его помощью автоматизировали поучение сетевых настроек пользователями.
- Установили простое средство мониторинга сетевой активности в консоли в режиме реального времени с помощью утилиты iftop.

Настройка интернет шлюза на Windows Server

В первую очередь нужно установить роль Remote Access. Для этого откроем консоль Server Manager, выбираем Manage -> Add Roles and Features, находим и отмечаем роль Remote Access, в ее составе выбираем службу Routing, и, соглашаясь со всеми предложенными по умолчанию компонентами, запускаем ее установку (Install). Установка службы маршрутизации на Windows Server 2012 R2

После окончания установки открываем консоль Routing and Remote Access (rrasmgmt.msc), щелкаем по имени сервера (с красной стрелкой) и выбираем Configure and Enable Routing and Remote Access.Настройка службы RRAS в Windows Server 2012 r2

В открывшемся окне выбираем пункт Network Address Translation (NAT).

RRAS включаем Network Address Translation (NAT)

На следующей шаге (NAT Internet Connection) нужно выбрать сетевой интерфейс, подключённый ко внешней сети / Интернету (в нашем примере это интерфейс Internet с ip 192.168.1.20). Этот интерфейс будет «публичным интерфейсом» нашего NAT роутера. Выбор внешнего NAT интерфейса

Далее будет предложено указать должен ли NAT роутер обеспечить клиентов внутренней сети сервисами DHCP и DNS. Как правило, этот функционал во внутренней сети уже имеется, поэтому в нем мы не нуждаемся. Настройка DHCP и DNS

На этом, базовая настройка маршрутизации на Windows Server 2012 R2, завершена. Сервер уже должен выполнять маршрутизацию пакетов между двумя подключенными сетями и выполнять трансляцию сетевых адресов (NAT).

Чтобы в этом убедиться, в консоли RRAS откройте свойства сервера. На вкладке General показано, что IPv4 маршрутизация включена (т.е. пакеты IPv4 будут пересылаться с одной сетевой карты на другую).

Проверить работу маршрутизации можно, указав на клиентском компьютере во внутренней сети (к которой подключен интерфейс сервера LAN) в качестве шлюза IP-адрес сервера (10.0.1.1), и выполнить ping или трассировку маршрута к ресурсу, расположенному во внешней сети или интернете. Эти попытки должны быть успешными.Простейший роутер на базе Windows Server 2012 R2

Примечание. Windows Server 2012 R2 поддерживает статическую маршрутизацию, протокол динамической маршрутизации RIPv2 и BGPv4. Поддержка OSPF была прекращена еще в Windows Server 2008.

В нашем случае на сервере осуществялется статическая маршрутизация. Если нужно добавить новый маршрут, щелкните ПКМ по Static Routes, выберите пункт меню New static route и создайте новое статическое правило маршрутизации.

Биллинговые системы: основные понятия

Системы, вычисляющие стоимость услуг связи для каждого клиента и хранящие информацию обо всех тарифах и прочих стоимостных

характеристиках, которые используются телекоммуникационными операторами для выставления счетов абонентам и взаиморасчетов с другими поставщиками услуг, носят название биллинговых; цикл выполняемых ими операций именуется биллингом. Биллинговая система (БС) — это бухгалтерская система, программное обеспечение, иными словами — «софт», разработанный специально для операторов. Каких операторов? Телекоммуникационных. Т. е. речь не идет лишь об операторах сотовой связи. БС используются также операторами обычной (стационарной, проводной) связи. В малых офисах, например, можно вести биллинг телефонии (анализировать: кто звонил, когда, сколько длился разговор). IP-телефония — другая область применения БС. А интернет-провайдеры? Они тоже используют БС, например, для формирования счетов, учета трафика. Любая БС создается на основе определенной системы управления базами данных (СУБД). Большинство БС в мире создавалось на основе СУБД Oracle. Среди других СУБД можно выделить Sybase и Informix как рассчитанные на большие объемы информации. А вот названия некоторых биллинговых систем: BIS, Flagship, CBOSS, Arbor, Bill-2000-prepaid. Стоит упомянуть, что под БС может подразумеваться и аппаратное обеспечение, участвующие в организации биллинга.

Терминология

Существуют несколько названий биллинговой системы: АСР — автоматизированная система расчетов; ИБС — информационная биллинговая система.

Одним из важных качеств БС является ее гибкость, то есть способность приспосабливаться к изменившимся обстоятельствам. Гибкая система адаптирована не только к сиюминутным потребностям оператора; за счет таких качеств, как настраиваемость, модульность и открытость она позволяет решать перспективные задачи. Чем больше у системы возможностей для

настроек, тем лучше. А что такое модульность? Модульный принцип построения системы — это такой принцип, при котором вся система собирается из отдельных частей (модулей), как дом собирается по кирпичикам. БС тоже состоит из таких модулей — подсистем. БС включает в себя, например, подсистему предварительной обработки данных, подсистему оперативного управления биллингом, подсистему оповещения клиентов (читайте ниже о структуре и функциях БС). Под открытостью системы подразумевается открытость исходного кода программного продукта, что позволяет оператору не зависеть от разработчика в будущем и самостоятельно обслуживать, и модернизировать систему. Тесно связано с гибкостью БС и следующее качество автоматизированных систем расчета — масштабируемость.

Масштабируемость по нагрузке. При росте абонентской базы, появлении дополнительных услуг не должна появляться необходимость изменять или дорабатывать программную часть БС. Увеличение возможностей БС должно достигаться за счет модернизации аппаратной части системы. Что важно учитывать при проектировании масштабируемых систем? Необходимо использовать СУБД, рассчитанные на большие объемы данных. СУБД должна быть совместима с различными компьютерными платформами, чтобы обеспечивать поддержку многопроцессорного режима работы.

Надежность — одно из основных требований, предъявляемым к любой системе. Надежность БС определяется надежностью СУБД и технологий, используемых при разработке системы. Далеко не последнее место занимает надежность поставщика (разработчика) прикладного программного обеспечения: время его работы на рынке и, как косвенный показатель, процент присутствия разработанных им систем на телекоммуникационном рынке.

Мультиязычность — возможность устанавливать различные языки для представления информации.

Мультивалютность — возможность работать с любыми валютами

Отложенный биллинг — биллинг, при котором расчеты производятся после состоявшихся звонков.

Горячий биллинг — изменение баланса счета происходит в процессе разговора, и информацию об остатке на Вашем счету можно получить сразу после звонка.

Оптимизация биллинга — улучшение, совершенствование оператором своей БС.

Большие БС — системы, применяемые крупными операторами.

Постинг биллинга — фиксация результатов расчета биллинга; после расчетов результаты становятся доступными пользователям (рассылаются, печатаются).

Так как БС предназначена для автоматизации расчетов с клиентом, то она и должна обеспечивать эту автоматизацию начиная с заключения договора до выписки счетов за услуги сотовой связи, причем корректно. При помощи подсистем автоматических услуг и автоматического сбора данных АСР должна предоставлять абонентам возможность самообслуживания. Некоторые БС позволяют абонентам оформлять заказы на подключение и производить оплату услуг через Интернет.

Структура и функции БС

Схема организации биллинга не сложна: информация о соединениях и их продолжительности записывается коммутатором и после предварительной обработки передается в расчетную систему. Расчетной системе «известны» тарифы. Она идентифицирует вызов и выполняет необходимые расчеты, формируя тем самым счет абонента. Очевидно, что в памяти системы должны храниться не только нормативы, тарифы и информация об услугах, но и данные о клиентах, заключенных контрактах с абонентами и сторонними поставщиками услуг связи (если таковые имеются), а также о стоимости передачи информации по разным каналам и направлениям (системой должно

быть также предусмотрено наличие дилеров: у них могут быть другие расценки, например, на подключение). Кроме этого, любая БС должна иметь базу, хранящую историю платежей: только эти сведения позволяют контролировать процесс оплаты и автоматизировать так называемую активацию/деактивацию абонентов. Эту функцию БС можно еще назвать защитной, так как она не позволяет пользоваться услугами сотовой связи тем, кто за них не платит.

По функциональным возможностям БС можно разделить на три класса: предназначенные для транснациональных операторов связи, заказные национального масштаба и системы среднего класса для региональных сетей.

БС, относящиеся к первому классу, должны обеспечивать взаимодействие сетей на межнациональном уровне, в различных временных зонах, т. е. они должны быть мультивалютными и мультиязычными.

Заказные системы национального масштаба создаются под определенного оператора. Оператору может понадобиться новая БС, совместимая с уже существующей расчетной системой. Разумеется, стоимость таких единичных систем значительно выше.

В масштабе региона можно вполне обойтись стандартными БС. Однако и такие системы должны обладать качествами, перечисленными выше: гибкостью, масштабируемостью, надежностью.

Любая БС создается и настраивается на бизнес-процесс определенного оператора связи, имеет собственный набор функций, соответствующий технологическому циклу предоставления услуг, и может работать с конкретным сетевым оборудованием, поставляющим ей информацию о вызовах и соединениях, — то есть БС не является «коробочным» продуктом. Но существует и стандартный набор функций, поддерживаемых практически всеми БС. В него входят:

- операции, выполняемые на этапе предварительной обработки и анализа исходной информации, например, функция получения данных о соединениях и услугах (запросы к коммутатору);
- операции управления сетевым оборудованием: функции активации/деактивации (блокировки/разблокировки) абонентов и команды изменения условий подписки абонентов, передаваемые непосредственно в коммутатор;
- основные функции приложения СУБД, включающие в себя: тарификацию записей коммутатора о вызовах и услугах; формирование и редактирование таблиц базы данных расчетной системы; выставление счетов и их печать; кредитный контроль счетов; составление отчетов; архивацию.

Как уже было сказано, БС должна обладать гибкостью или модульностью. Каждый элемент АСР обеспечивает реализацию конкретного участка технологической цепочки обслуживания клиента. Основные подсистемы, характерные для биллинга, это: подсистема предварительной обработки данных о соединениях, оперативное управление биллингом и подсистема оповещения клиентов.

Подсистема предварительной обработки данных.

Это приложение анализирует исходную информацию о соединении, определяет класс предоставляемой услуги и параметры трафика (направление вызова, источник, зоны взаиморасчетов, условия роуминга). В состав данной подсистемы входит декодер исходной информации о соединениях. Одна из сложнейших процедур этой подсистемы — поддержка роуминга. Дело в том, что требуется конвертировать роуминговые записи всевозможных форматов от разных коммутаторов (с учетом различных стандартов передачи информации в канале связи) и разных биллинговых систем в тот формат записи, которым пользуется данная БС. Программное обеспечение (ПО) тарифицирует все записи о соединениях между операторами (согласно

проходящему трафику) и создает служебные таблицы, которые используются остальными подсистемами для выполнения расчетов с абонентами, взаиморасчетов операторов связи и формирования отчетов. Современные БС позволяют обрабатывать различные телекоммуникационные услуги, обеспечивая удобное выставление счетов (один клиент — один баланс — один счет). Это достигается за счет применения «интеллектуальных систем» предварительной обработки исходной информации о соединениях, трафике и услугах, выполняющих тарификацию независимо от вида связи.

Подсистема оперативного управления биллингом.

Данная подсистема дает возможность автоматически или через оператора биллинговой системы изменять условия подписки абонентов на коммутаторе, т. е. блокировать связь конкретного абонента или снимать эту блокировку, включать или отменять услугу. Вы звоните оператору и говорите: «Включите мне, пожалуйста, голосовой ящик». Вам отвечают: «Пожалуйста, назовите свой номер». После еще нескольких «обменов любезностями» Ваш голосовой ящик оказывается включенным.

Подсистема оповещения клиентов.

Неотъемлемая часть современного биллинга — подсистема оповещения клиентов с помощью голосовых или электронных сообщений. Информацию для рассылки уведомлений и объявлений данная подсистема берет из таблиц базы.

Перечисленное деление на функциональные подсистемы не является «строгим» для всех БС. Это лишь пример «классической» АСР.

Стандарты биллинга

Чтобы обеспечить взаимопонимание между различными БС разных операторов (это, например, требуется при роуминге, были разработаны группы стандартов биллинга. Основных международных групп стандартов три.

В 1998 г. американский институт стандартов ANSI утвердил стандарт ANSI 124. Дальнейшим усовершенствованием и поддержкой ANSI 124 занимается ассоциация TIA. После этого компания CIBERNET создала рабочую группу для определения спецификаций бизнес-процессов при передаче сообщений в стандарте ANSI 124, которые получили название NSDP-B&S. Данные спецификации устанавливают однозначное соответствие между бизнес-процессами телекоммуникационных операторов и информацией, передаваемой при обмене данными между коммутаторами по стандарту ANSI 124.

В 1998 г. было опубликовано описание первого североамериканского биллингового стандарта CIBER, который в настоящее время поддерживается фирмой CIBERNET и ее комитетом CAC-IS. Этот комитет объединяет разработчиков биллинговых систем и телекоммуникационных операторов. Главная область применения CIBER — сотовые сети стандарта AMPS.

Европейский (по происхождению) стандарт TAP появился в 1992 г. Он поддерживается рабочей группой TADIG. Большинство операторов Европы используют TAP2, хотя существует и третья версия. С 1995 г. модификация TAP2, известная как спецификация TD.27, или NAGTAP2, начала применяться и в США.

WEB-сервер

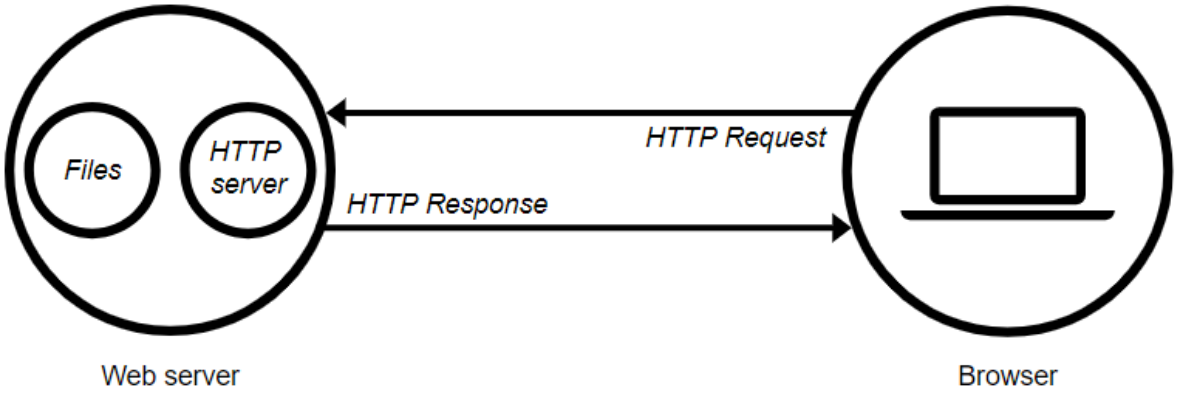
Описание

Понятие Веб-сервер может относиться как к железу, так и к программному обеспечению (ПО).

С точки зрения железа Веб-сервер — это компьютер, который хранит ресурсы сайта (HTML документы, CSS стили, JavaScript файлы и другое) и доставляет их на устройство конечного пользователя (веб-браузер и т.д.). Обычно он подключен к сети Интернет и может быть доступен через доменное имя, например, mozilla.org.

С точки зрения ПО, Веб-сервер включает в себя некоторые вещи, которые контролируют доступ Веб-пользователей к размещенным на сервере файлам, это минимум HTTP сервера. HTTP сервер — это часть ПО, которая понимает URL'ы (веб-адреса) и HTTP (протокол который использует ваш браузер для просмотра веб-станиц).

Простыми словами, когда браузеру нужен файл, размещенный на веб-сервере, браузер запрашивает его через HTTP. Когда запрос достигает нужного веб-сервера (железо), сервер HTTP (ПО) передает запрашиваемый документ обратно, также через HTTP.



Чтобы опубликовать веб-сайт, нужен либо статический, либо динамический веб-сервер.

Статический веб-сервер или стек состоит из компьютера (железо) с сервером HTTP (ПО). Мы называем это «статикой», потому что сервер посылает размещенные на нем файлы в браузер не изменяя их.

Динамических веб-сервер состоит из статического веб-сервера плюс дополнительного программного обеспечения, наиболее часто сервером

приложений и базы данных. Мы называем его «динамический», потому что сервер приложений изменяет исходные файлы перед отправкой в ваш

Например, для получения итоговой страницы, которую вы видите в браузере, сервер приложений может заполнить HTML шаблон данными из базы данных. Такие сайты, как MDN (Mozilla Developer Network) или Википедия состоят из тысяч веб-страниц, но они не являются реальными HTML документами, лишь несколько HTML шаблонов и гигантские базы данных. Эта структура упрощает и ускоряет сопровождение веб-приложений и доставку контента.

Более детально

Чтобы загрузить веб-страницу, как мы уже говорили, браузер отправляет запрос к веб-серверу, который приступает к поиску запрашиваемого файла в своем собственном пространстве памяти. Найдя файл, сервер считывает его, обрабатывает так, как ему это необходимо, и направляет его в браузер. Давайте рассмотрим эти шаги более подробно.

Хостинг файлов

Во-первых, веб-сервер хранит файлы веб-сайта, а именно все HTML документы и связанные с ними ресурсы, включая изображения, CSS стили, JavaScript файлы, шрифты и видео.

Технически, вы можете разместить все эти файлы на своем компьютере, но гораздо удобнее хранить их на выделенном веб-сервере, который:

- всегда запущен и работает
- постоянно в сети Интернет
- имеет один и тот же IP адрес все время (не все провайдеры предоставляют статический IP адрес для домашнего подключения)
- обслуживается на стороне

Таким образом, выбор хорошего хостинг-провайдера является важной частью создания сайта. Рассмотрите различные предложения компаний и выберите то, что соответствует вашим потребностям и бюджету (предложения варьируются от бесплатных до тысяч долларов в месяц).

Связь по HTTP

Во-вторых, веб-сервер обеспечивает поддержку HTTP (hypertext transfer protocol). Как следует из названия, HTTP указывает, как передавать гипертекст (т.е. связанные веб-документы) между двумя компьютерами.

Протокол представляет собой набор правил для связи между двумя компьютерами. HTTP является текстовым протоколом без сохранения состояния.

Ни клиент, ни сервер, не помнят о предыдущих соединениях. Например, опираясь только на HTTP, сервер не сможет вспомнить введенный вами пароль, или на каком шаге транзакции вы находитесь. Для таких задач вам потребуется сервер приложений.

HTTP задает строгие правила, как клиент и сервер должны общаться. Более подробно смотри http-protocol. Вот некоторые из них:

Только клиенты могут отправлять HTTP запросы, и только на сервера. Сервера отвечают только на HTTP запросы клиента.

Когда запрашивается физический файл, клиент должен сформировать file URL (file:///var/log/syslog)

Веб-сервер должен ответить на каждый HTTP запрос, по крайней мере с сообщением об ошибке.

На веб-сервере, HTTP сервер отвечает за обработку входящих запросов и ответ на них.

При получении запроса, HTTP сервер сначала проверяет существует ли ресурс по данному URL.

Если это так, веб-сервер отправляет содержимое файла обратно в браузер. Если нет, сервер приложений создает необходимый ресурс.

Если это невозможно, веб-сервер возвращает сообщение об ошибке в браузер, чаще всего «404 Not Found». (Эта ошибка настолько распространена, что многие веб-дизайнеры тратят большое количество времени на разработку 404 страниц об ошибках.)

Статика и Динамика

Грубо говоря, сервер может отдавать статическое или динамическое содержимое.

«Статическое» означает «отдается как есть». Статические веб-сайты проще всего установить, поэтому мы предлагаем вам сделать свой первый сайт статическим.

«Динамическое» означает, что сервер обрабатывает данные или даже генерирует их на лету из базы данных. Это обеспечивает больше гибкости, но технически сложнее в обслуживании, что делает его более сложным для создания веб-сайта.

Существует много серверов приложений для разных запросов, поэтому довольно трудно выбрать какой-то один универсальный. Некоторые серверы приложений удовлетворяют определенной категории веб-сайтов, такие как блоги, вики или интернет-магазины; другие, называемые CMS (системы управления контентом), являются более общими. Если вы создаете динамический сайт, потратьте немного времени на выбор инструмента, который соответствует вашим потребностям. Если вы не хотите изучать веб-программирование, то вам не нужно создавать свой собственный сервер приложений. Это будет очередной велосипед.

Передача информации веб-сервера выполняется по протоколу HTTP (HyperText Transfer Protocol), изначально созданного для работы с HTML-страницами. Уже позже стало возможным отправлять через HTTP файлы любых типов. В последнее время преобладают сайты, работающие через HTTPS. Это улучшенная версия HTTP, которая отличается от

предшественника тем, что поддерживает шифрование трафика TLS/SSL между пользователем и сервером.

WEB-серверы

Звание самого популярного веб-сервера в мире уже более 25 лет удерживает за собой Apache HTTP Server, который принято называть сокращенно Apache или «Апач». Сегодня программа обслуживает более 40% всех существующих серверов, включая проекты IBM, eBay, PayPal и Facebook.

Рассмотрим причины популярности Apache подробнее. Это не только пополнит копилку знаний об интернет-технологиях, но и поможет сделать правильный выбор веб-сервера для размещения сайта в будущем.

Что это такое

Apache – это свободное программное обеспечение для размещения веб-сервера. Он хорошо показывает себя в работе с масштабными проектами, поэтому заслуженно считается одним из самых популярных веб-серверов. Кроме того, Apache очень гибок в плане настройки, что даёт возможность реализовать все особенности размещаемого веб-ресурса.

История создания

Apache HTTP Server был выпущен в 1995 году разработчиком Робертом Маккулом из Университета штата Иллинойс (UIUC). Продукт возник как доработанная версия другого HTTP-клиента – NCSA HTTPd 1.3, созданного Робертом ранее.

Основой для модификации стали многочисленные «патчи» или программные «заплатки» для NCSA. Именно отсюда (а не от индейского племени апачей) изначально и происходит название Apache. Оно расшифровывается как «a patchy server» или «сервер с патчами».

Что такое Apache - история

Разработкой и поддержкой продукта с 1999 года занимается организация Apache Software Foundation (ASF) – сообщество экспертов-энтузиастов со всего мира. Этим же некоммерческим фондом была создана официальная лицензия ПО – Apache License.

В 2000 году ASF представило новую версию Apache 2.0 с полностью переработанной архитектурой, свободной от кода NCSA. С этого момента веб-сервер развивается по двум основным веткам – 1.x и 2.x.

Как устроен Apache

Что такое Apache - устройство

Архитектура

Apache состоит из ядра и динамической модульной системы. Параметры системы изменяются с помощью конфигурационных файлов.

Ядро

Ядро Apache разработано Apache Software Foundation на языке C. Основные функции — обработка конфигурационных файлов, протокол HTTP/HTTPS и загрузка модулей. Ядро может работать без модулей, но будет иметь ограниченный функционал.

Модульная система

Модуль – отдельный файл, подключение которого расширяет изначальный функционал ядра. Они могут включаться в состав ПО при первоначальной установке или подгружаться позже через изменение конфигурационного файла.

Большинство из них отвечает за определенный аспект обработки клиентского запроса – поддержку различных языков программирования, безопасность, кэширование, аутентификацию и т.д. Таким образом, большая задача разбивается на несколько фаз, каждую из которых решает отдельный, узкоспециализированный модуль.

Для Apache существует больше 500 модулей. Многие популярные веб-приложения сразу выпускаются в виде модуля к Apache. Например, ISPmanager и VDSmanager.

Конфигурация

Система конфигурации Apache работает на текстовых файлах с прописанными настройками. Она подразделяется на три условных уровня, для каждого из которых имеется свой конфигурационный файл:

Уровень конфигурации сервера (файл httpd.conf) – основной конфигурационный файл. Действие распространяется на весь механизм веб-сервера.

Уровень каталога (файл .htaccess) – дополнительный конфигурационный файл. Его директивы охватывают только каталог, где расположен файл, а также вложенные подкаталоги.

Уровень виртуального хоста (файл httpd.conf> или extra/httpd-vhosts.conf).

Обычно конфигурационные файлы Apache находятся в папке «conf», а дополнительные конфигурационные файлы во вложенной в нее папке «extra». Внести изменения можно как через редактирование самого файла, так и через командную строку.

Виртуальные хосты

Веб-хост – это компонент сервера, отвечающий за обслуживание одного размещенного на нем объекта (сайта, виртуального сервера). Система виртуальных хостов Apache позволяет одновременно запускать несколько проектов с одного IP-адреса.

В Apache можно установить настройки модуля и ядра, а также вводить лимиты на потребление серверных ресурсов (трафик, RAM, CPU) для каждого виртуального хоста в отдельности. Это технологическая основа всего механизма веб-хостинга.

Достоинства и недостатки Apache

Плюсы

Доступность. Это программное обеспечение с открытым исходным кодом. Значит, его может бесплатно использовать или модифицировать любой желающий. Разработчики по всему миру создают конфигурации и модули веб-сервера для своих специфических нужд. По этой же причине Apache регулярно получает полезные дополнения, расширяющие его базовый функционал.

Гибкость настройки. Apache использует несколько конфигурационных файлов для управления веб-сервером. Это позволяет настроить ПО под узконаправленные задачи.

Функциональность. У Apache динамическая модульная структура. Можно быстро подключать дополнительный функционал в виде скачиваемых модулей, даже без обращения к внешним источникам. Это позволяет решать целый комплекс важнейших задач в области безопасности, кэширования, редактирования URL, распределения нагрузки. Благодаря гибридным модулям MPM, Apache может одинаково успешно обслуживать статический и динамический контент. Есть возможность оперативно отключать ненужные модули и ускорять работу веб-сервера

Кроссплатформенность. Apache работает как на Windows, так и на всех Unix-подобных системах. Администрирование веб-сервером не имеет серьёзных отличий на разных ОС. Индивидуален только процесс установки и расположение директорий с файлами программы.

Совместимость. Apache работает на базе скриптовых или веб-ориентированных языков (PHP, Python, Tcl, Ruby, Perl, ASP), что делает его совместимым с самым широким спектром баз данных и серверного ПО. Многие веб-приложения и инструменты сразу выходят со средствами запуска из-под Apache в виде PHP-модуля. Веб-сервер, поддерживает технологии

FastCGI и CGI, позволяющие пользоваться программными продуктами на объектно-ориентированных языках Java, sh, C, C++.

Масштабируемость. Подходит для веб-ресурсов любого масштаба. Apache хорошо работает как на одностраничном сайте (лендинге), так и на многостраничном сайте с ежедневной аудиторией в десятки тысяч посетителей.

Поддержка пользователей. Apache удерживает первенство популярности среди веб-серверов с 1996 года. За прошедшее время для него создана обширнейшая база документации – как официальной, так и созданной сторонними разработчиками. Готовые, подробно описанные руководства можно найти практически на любой сценарий.

Что такое Apache - плюсы и минусы

Минусы

Производительность. Скорость обработки запросов Apache несколько ниже, по сравнению со своими конкурентами. Гибкость веб-сервера в некоторых случаях вредит производительности. Например, Apache приходится каждый раз считывать несколько конфигурационных файлов на сервере, затрачивая системные ресурсы и время. Но этот и многие другие факторы можно исправить, отключив ненужные опции. Правда в таком случае функциональность Apache не будет сильно отличаться от других веб-серверов.

Сложная конфигурация повышает уязвимость. Возможность подключать модули в Apache это не всегда преимущество. Чем больше модулей, тем сложнее становятся настройки. Соответственно, больше шансов допустить критические пробелы в контуре безопасности.

Синтаксис конфигов.. В файлах с параметрами программы используются разнообразные переменные, поэтому настройка и управление веб-сервером может показаться сложной новичкам. Упростить администрирование Apache можно с помощью бесплатного инструмента Apache GUI.

Излишний функционал. Даже без дополнительных модулей Apache предоставляет пользователям массу возможностей. Правда, большинство использует лишь небольшую часть базового функционала приложения. Поэтому часто после установки приходится тратить время на отключение «лишних» модулей.

Альтернативы Apache

NGINX

Nginx (Engine-X, «энжинкс») — второе по популярности веб-серверное приложение и главный конкурент Apache. Было выпущено в 2004 году под открытой лицензией BSD. Изначально приложение создавалось для решения проблемы масштабирования, известной как «10 тысяч соединений» (C10к). Это значит, что до Nginx веб-сервер не был способен одновременно обрабатывать пользовательские запросы более чем с 10 000 подключений.

У этого веб-сервера асинхронная событийно-ориентированная архитектура (event-driven), которая позволяет добиваться быстрого масштабирования даже при минимальных ресурсах. Вместо того, чтобы создавать новый процесс для каждого пользовательского запроса, Nginx обрабатывает множество соединений в едином потоке.

Nginx отлично подходит для веб-проектов с высокой посещаемостью. Однако веб-сервер не может самостоятельно работать с динамическим контентом. Поэтому его чаще используют для статических веб-сайтов или например, в связке с PHP-FPM или Apache HTTP Server как прокси-сервер.

Lighttpd

Веб-сервер Lighttpd (произносится «лайти») — кроссплатформенное программное обеспечение на языке C. Выпущено в 2003 году под лицензией

BSD. «Лайти» работает на операционных системах Windows и семейства Unix/Linux. Приложение поддерживает технологии FastCGI, SCGI, HTTP proxy, Auth, перезаписи URL и AJP (с версии 1.5).

Как и Nginx, изначально «Лайти» создавалось для решения проблемы «C10к». Неудивительно, что его специализация — веб-проекты с большой посещаемостью. В числе компаний, использующих Lighttpd, такие гиганты, как Google, Википедия, Яндекс и Ubuntu.

Microsoft IIS

Internet Information Services (IIS) — набор сервисов для создания веб-сервера от компании Microsoft. Распространяется в комплекте с операционными системами Windows NT как дополнительно устанавливаемый компонент. Веб-сервер поддерживает технологии CGI, FastCGI, ISAPI и SSI.

Главная сила IIS – в глубокой интеграции и поддержке продуктов Microsoft. Его часто выбирают те, чьи ресурсы работают на движке ASP.NET и используют скриптовый язык ASPX. Главный недостаток – жесткая привязка к операционной системе Windows и отсутствие версий для Unix/Linux.

УСТАНОВКА APACHE

Для установки сначала обновим систему до самой новой версии:

sudo apt update

sudo apt upgrade

Затем установка apache2:

sudo apt install apache2

В других дистрибутивах пакет программы называется либо так, либо httpd и его установка у вас не вызовет трудностей.

После завершения установки нужно добавить веб-сервер в автозагрузку, чтобы не запускать его вручную после включения компьютера:

sudo systemctl enable apache2

НАСТРОЙКА APACHE

Уже прошло то время, когда конфигурация Apache хранилась в одном файле. Но оно и правильно, когда все распределено по своим директориям, в конфигурационных файлах легче ориентироваться.

Все настройки содержатся в папке /etc/apache/:

Файл /etc/apache2/apache2.conf отвечает за основные настройки

/etc/apache2/conf-available/* - дополнительные настройки веб-сервера

/etc/apache2/mods-available/* - настройки модулей

/etc/apache2/sites-available/* - настойки виртуальных хостов

/etc/apache2/ports.conf - порты, на которых работает apache

/etc/apache2/envvars

Как вы заметили есть две папки для conf, mods и site. Это available и enabled. При включении модуля или хоста создается символическая ссылка из папки available (доступно) в папку enable (включено). Поэтому настройки лучше выполнять именно в папках available. Вообще говоря, можно было бы обойтись без этих папок, взять все и по старинке свалить в один файл, и все бы работало, но сейчас так никто не делает.

Сначала давайте рассмотрим главный файл конфигурации:

vi /etc/apache2/apache2.conf

Timeout - указывает как долго сервер будет пытаться продолжить прерванную передачу или прием данных. 160 секунд будет вполне достаточно.

KeepAlive On - очень полезный параметр, позволяет передавать несколько файлов, за одно соединение, например, не только саму html страницу, но и картинки и css файлы.

MaxKeepAliveRequests 100 - максимальное количество запросов за одно соединение, чем больше, тем лучше.

KeepAliveTimeout 5 - таймаут соединения, обычно для загрузки страницы достаточно 5-10 секунд, так что больше ставить не нужно, но и рвать соединение раньше чем загрузились все данные тоже не нужно.

User, Group - пользователь и группа, от имени которых будет работать программа.

HostnameLookups - записывать в логи вместо ip адресов доменные имена, лучше отключить, чтобы ускорить работу.

LogLevel - уровень логирования ошибок. По умолчанию используется warn, но чтобы логи заполнялись медленнее достаточно включить error

Include - все директивы include отвечают за подключение рассмотренных выше конфигурационных файлов.

Директивы Directory отвечают за настройку прав доступа к той или иной директории в файловой системе. Синтаксис здесь такой:

<Directory /адрес/в/файловой/системе/>

Параметр значение

</Directory>

Здесь доступны такие основные опции:

AllowOverride - указывает нужно ли читать .htaccess файлы из этой директории, это такие же файлы настроек и таким же синтаксисом. All - разрешать все, None - не читать эти файлы.

DocumentRoot - устанавливает из какой папки нужно брать документы для отображенияа пользователю

Options - указывает какие особенности веб-сервера нужно разрешить в этой папке. Например, All - разрешить все, FollowSymLinks - переходить по символическим ссылкам, Indexes - отображать содержимое каталога если нет файла индекса.

Require - устанавливает, какие пользователи имеют доступ к этому каталогу. Require all denied - всем запретить, Require all granted - всем разрешить. можно использовать вместо all директиву user или group чтобы явно указать пользователя.

Order - позволяет управлять доступом к директории. Принимает два значения Allow,Deny - разрешить для всех, кроме указанных или Deny,Allow - запретить для всех, кроме указанных. Теперь мы можем запретить доступ к директории для всех: Deny from all, а затем разрешить только для приложения от losst.ru: Allow from losst.ru.

Здесь все эти директивы не используются, поскольку нас устраивают значения по умолчанию, но вот в файлах .htaccess они могут быть очень полезны.

У нас остался файл /etc/apache2/ports.conf:

В нем только одна директива, Listen, которая указывает программе на каком порту нужно работать.

Последний файл /etc/apache2/envvars, его вы вряд ли будете использовать, в нем указанны переменные, которые можно использовать в других конфигурационных файлах.

Дальше поговорим немного о htacess. Совсем немного.

НАСТРОЙКА СЕРВЕРА APACHE ЧЕРЕЗ HTACCESS

Файлы .htaccess позволяют настраивать веб-сервер на Ubuntu для поведения в определенной директории. Все инструкции, указанные в этом файле выполняются как бы они были обвернуты в тег <directory адрес_папки> если бы находились в основном файле.

Важно заметить, что для того, чтобы сервер читал инструкции из .htaccess настройки для этой папки в основном файле или файле виртуального хоста не должны содержать AllowOverride None, чтобы могли работать все настройки нужно AllowOverride All.

А в остальном, здесь может выполняться любая настройка сервера apache, от включения модулей, до обычного изменения доступа к папке. Поскольку все параметры мы уже рассмотрели просто приведем пару примеров:

Order Deny,Allow

Deny from all

Запрещает всем доступ к этой папке, важно применить, для папок с конфигурацией. Чаще всего .htaccess используется для работы с модулем mod_rewrite, который позволяет изменять запросы на лету:

RewriteEngine on

RewriteRule ^product/([^/\.]+)/?$ product.php?id=$1 [L]

Но это очень обширная тема и выходит за рамки этой статьи.

НАСТРОЙКА МОДУЛЕЙ APACHE

Как я уже говорил, Apache - модульная программа, ее функциональность можно расширять с помощью модулей. Все доступные модули загрузчики и конфигурационные файлы модулей находятся в папке /etc/apache/mods-available. А активированные в /etc/apache/mods-enable.

Но вам необязательно анализировать содержимое этих папок. Настройка Apache 2.4 с помощью добавления модулей выполняется с помощью специальных команд. Посмотреть все запущенные модули можно командой:

apache2ctl -M

Включить модуль можно командой:

sudo a2enmod имя_модуля

А отключить:

sudo a2dismod имя_модуля

После включения или отключения модулей нужно перезагрузить apache:

sudo systemctl restart apache2

Во время выполнения одной из этих команд создается или удаляется символическая ссылка на файл модуля с расширением load в директории mods-available. Можете посмотреть содержимое этого файла, там только одна строка. Например:

vi /etc/apache2/mods-available/deflate.load

Это к тому, что активировать модуль можно было просто добавив эту строчку в файл apache2.conf. Но принято делать именно так, чтобы избежать путаницы.

Настройки модулей находятся в той же папке, только в файле с расширением .conf вместо load. Например, посмотрим настройки того же модуля для сжатия deflate:

vi /etc/apache2/mods-available/deflate.conf

Файлы в папке conf-available, это такие же модули, только они установлены отдельно от apache, это может быть конфигурационные файлы для включения модуля php или любого другого языка программирования. Здесь работает все точно так же, только команды для включения и отключения этих модулей немного другие:

a2enconf имя_модуля

a2disconf имя модуля

Как вы убедились, включать модули очень просто. Давайте включим несколько необходимых, но не включенных по умолчанию модулей:

sudo a2enmod expires

sudo a2enmod headers

sudo a2enmod rewrite

sudo a2enmod ssl

Модули expires и headers уменьшают нагрузку на сервер. Они возвращают заголовок Not Modified, если документ не изменился с последнего запроса. Модуль expiries позволяет устанавливать время, на которое браузер должен кэшировать полученный документ. Rewrite позволяет изменять запрашиваемые адреса на лету, очень полезно при создании ЧПУ ссылок и т д. А последний для включения поддержки шифрования по SSL. Не забудьте перезагрузить apache2 после завершения настроек.

НАСТРОЙКА ВИРТУАЛЬНЫХ ХОСТОВ APACHE

Было бы не совсем удобно, если на одной физической машине можно было размещать только один сайт. Apache может поддерживать сотни сайтов на одном компьютере и выдавать для каждого из них правильное содержимое. Для этого используются виртуальные хосты. Сервер определяет к какому домену приходит запрос и отдает нужное содержимое из папки этого домена.

Настройки хостов Apache расположены в папке /etc/apache2/sites-available/. Для создания нового хоста достаточно создать файл с любым именем (лучше кончено с именем хоста) и заполнить его нужными данными. Обернуть все эти параметры нужно в директиву VirtualHost. Кроме рассмотренных параметров здесь будут использоваться такие:

ServerName - основное имя домена

ServerAlias - дополнительное имя, по которому будет доступен сайт

ServerAdmin - электронная почта администратора

DocumentRoot - папка с документами для этого домена

Например:

```
vi /etc/apache2/sites-available/test.site.conf
<VirtualHost *:80>
ServerName test.site
ServerAlias www.test.site
ServerAdmin webmaster@localhost
DocumentRoot /var/www/test.site/public_html
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Виртуальные хосты, как и модули нужно активировать. Для этого есть специальные утилиты. Чтобы активировать наберите:

```
sudo a2ensite test.site
```

Здесь test.site - имя файла виртуального хоста. Для отключения тоже есть команда:

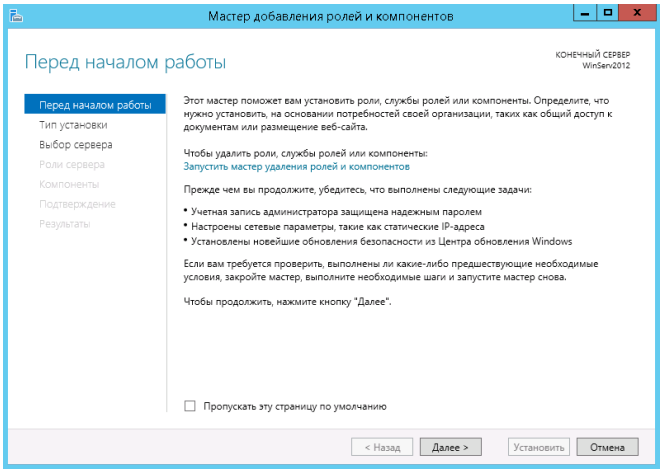
```
sudo a2dissite test.site
```

Настройка виртуальных хостов Apache завершена и на публичном сервере это все бы уже работало, но если вам нужна настройка Apache на домашней машине, то вы ваш новый сайт не откроется в браузере. Браузер не знает такого сайта. И откуда ему знать? DNS службы не могут ничего сообщить об этом доменном имени. Но в системе Linux мы можем сами указать ip адреса для доменных имен в файле /etc/hosts. Поэтому добавляем в конец файла такие строки:

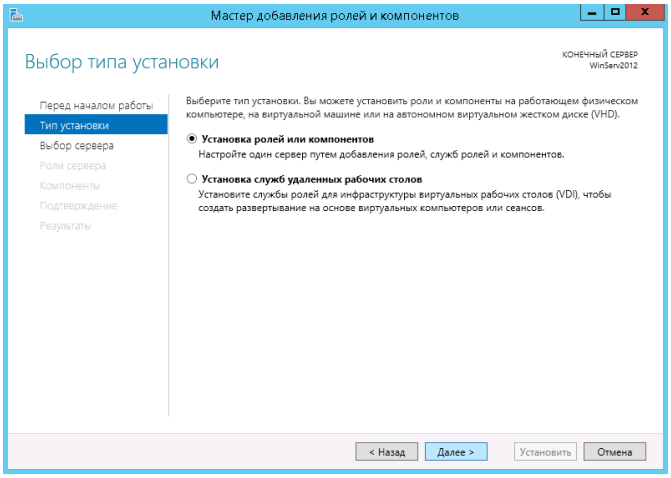
```
vi /etc/hosts
127.0.0.1 test.site
127.0.0.1 www.test.site
```

Установка IIS

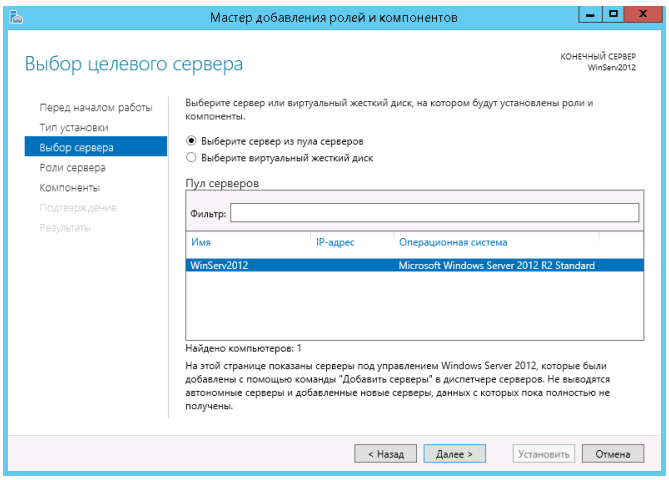
Зайдите в **Диспетчер серверов**, в правом верхнем углу выберете Управление -> Добавить роли и компоненты.



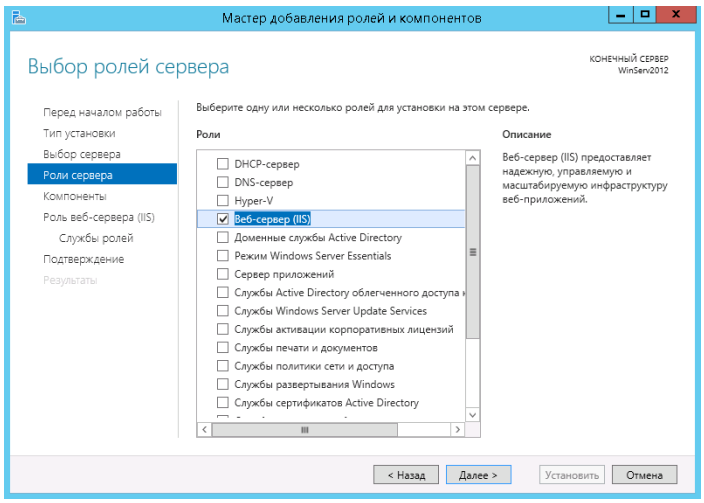
Выберете тип: установка ролей и компонентов.

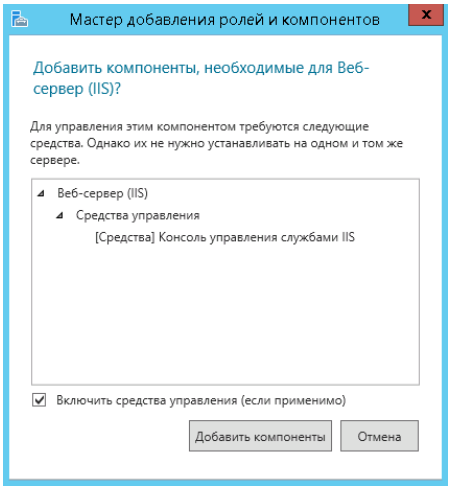


Выберете сервер из пула и нажмите **Далее**.

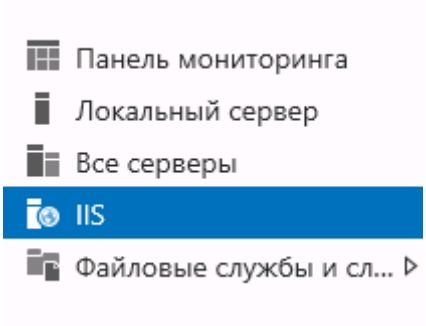


На следующем шаге отметьте галочкой нужную нам роль - Веб-сервер IIS.



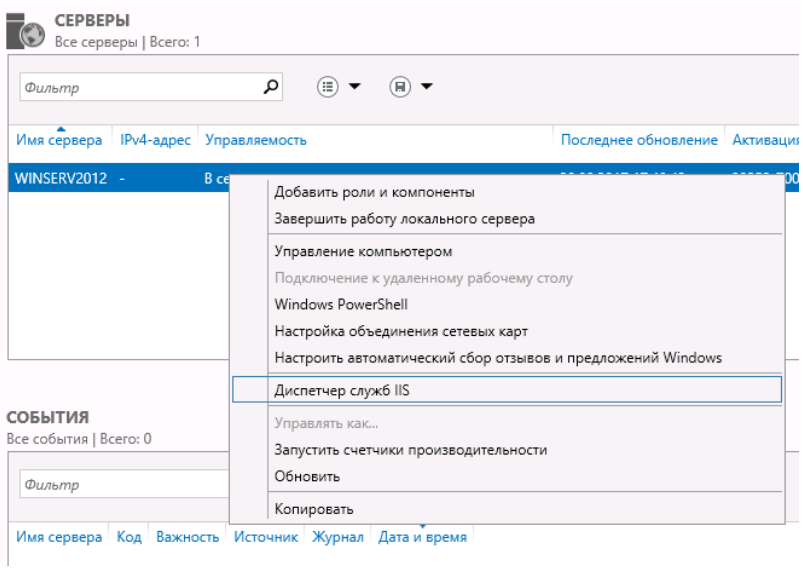


На следующем этапе **важно** отметить дополнительный компонент “**Функции .NET Framework 3.5**”, по желанию можно отметить дополнительные составляющие, но для базовой работы IIS они не являются необходимыми. В результате выполните установку веб-сервера, он отобразится в диспетчере.

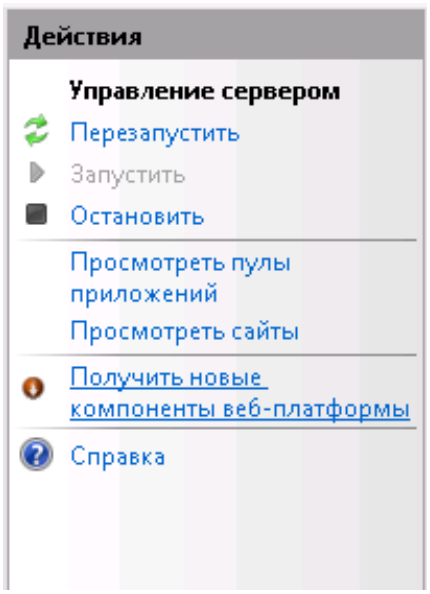


Установщик веб-платформы

Необходимо установить сервис под названием Установщик веб-платформы, с помощью которого будет происходить установка PHP и MySQL. Откройте диспетчер служб IIS как показано на изображении.



В вертикальном меню справа выберете “Получить новые компоненты веб-платформы”.



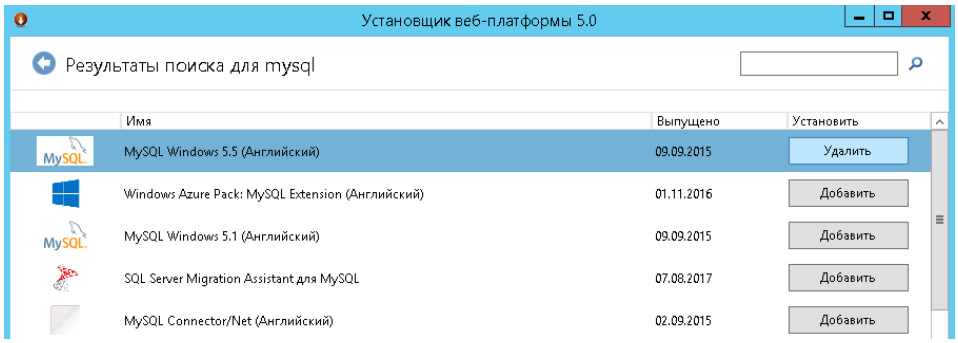
Вы будете перенаправлены на веб-сайт в браузере по умолчанию. Скачайте данный сервис и установите его.

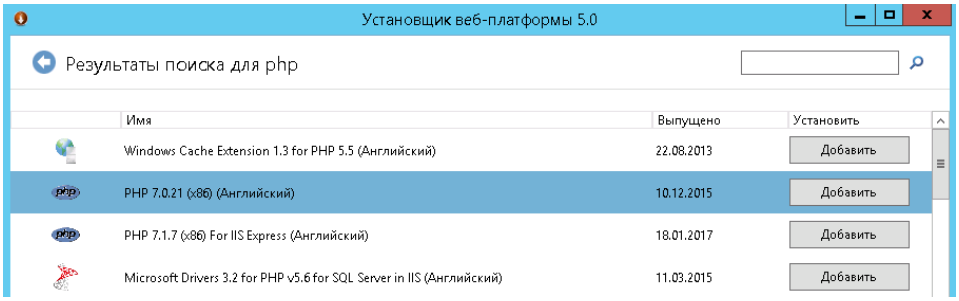
Примечание: если у вас не получается скачать файл в IE из-за настроек безопасности, необходимо их отключить.

Примечание: чтобы открыть приложение, выберете тот же пункт меню "Получить новые компоненты веб-платформы".

Настройка PHP и MySQL на IIS

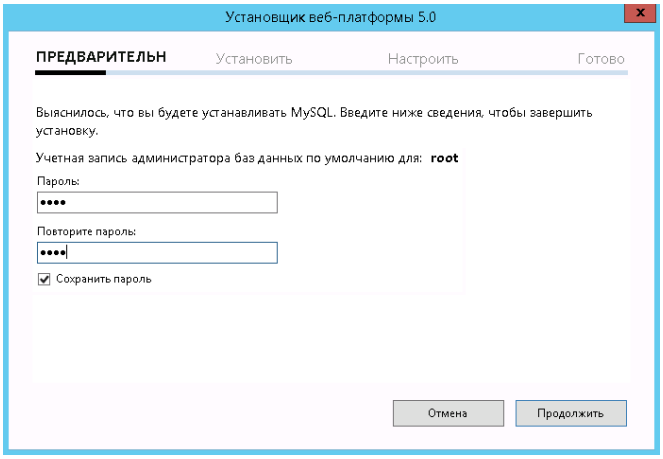
Зайдите в Установщик веб-платформы, с помощью поиска найдите последнюю доступную версию MySQL и PHP и нажмите **Добавить**.





Затем установите выбранные приложения.

Перед вами появится окно для ввода пароля для суперпользователя СУБД.

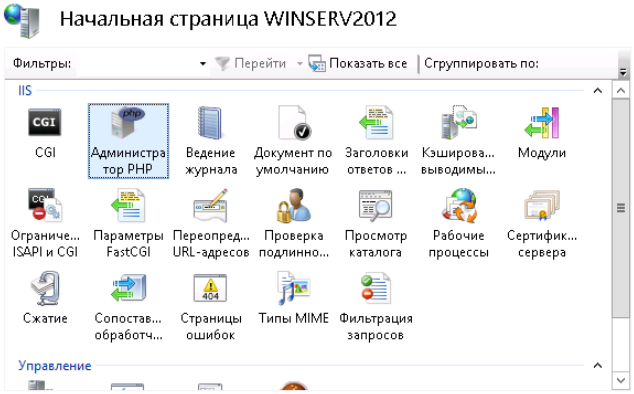


Выполните установку.

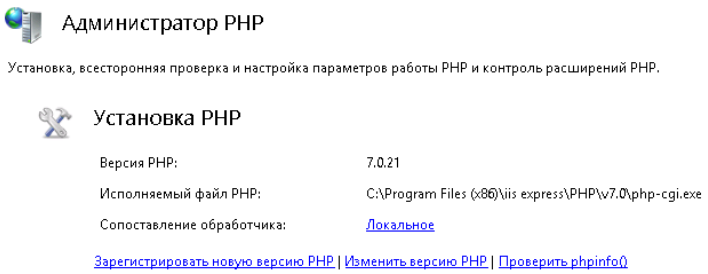
Примечание: при возникновении ошибок проверьте присутствие .NET Framework 3.5. Если он установлен, возможно, вы столкнулись с проблемой, известной в поздних версиях IIS: ошибкой проверки сигнатур при загрузке пакетов установки PHP Manager. В этом случае установите PHP Manager вручную. После установки таким способом при проверке компонентов PHP может появиться информация о несоответствии версий этих компонентов. Сообщение можно проигнорировать.

Проверка

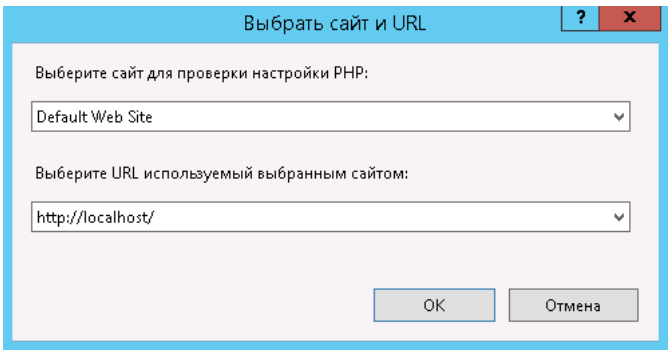
Проверить корректность установки на IIS можно следующем образом. В диспетчере служб появится иконка Администратор PHP.



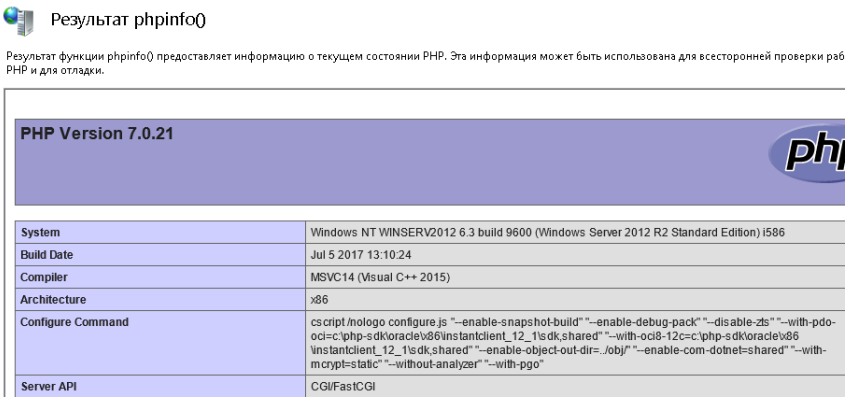
Откройте утилиту и перейдите по ссылке “Проверить phpinfo()”.



В появившемся окне выберете сайт для проверки.



В результате должна отобразиться похожая страница.



Для проверки установки СУБД откройте PowerShell и перейдите в директорию с помощью команды:

```
cd “C:\Program Files\MySQL\MySQL Server 5.5\bin”
```

Запустите СУБД сервер и введите пароль:

```
./mysql -u root -p
```

Анализ сетевого трафика в режиме реального времени: обзор прикладных задач, подходов и решений

История развития средств анализа сетевого трафика

Зарождение технологий анализа сетевого трафика можно отнести к началу 90х годов прошлого века. Потребности в их возникновении появились примерно в одно время в нескольких областях.

Усложнение схем сетей и многообразие сетевых устройств привели к усложнению их настройки и поддержки сети в работоспособном

состоянии — необходим был инструмент позволяющий, с одной стороны локализовать проблему, а с другой предоставить как можно более исчерпывающую информацию о природе проблемы. Собственно, объектом, который содержит в себе всю необходимую информацию и является сетевой трафик. Одним из инструментов, изначально предназначенным для решения именно этой проблемы стал сетевой сниффер/анализатор Wireshark (ранее Ethereal), созданный инженером Джеральдом Комбом (Gerald Comb) в 1997 году. Wireshark продолжает активно развиваться и является стандартом в определённой области сетевого анализа.

В это же время начинает применяться технология трансляции адресов NAT, предназначенной как для того, чтобы сэкономить IP адреса, так и для того, чтобы скрыть от внешнего наблюдателя устройство и ресурсы внутренней локальной сети. Для реализации этой технологии требовался инструмент — аппаратный или программный транслятор адресов. Данный функционал в результате был внедрён в качестве составной части в большинство маршрутизаторов. Существуют и программные реализации, как в составе серверных операционных систем, так и в виде отдельных приложений.

К этому же времени относятся первые упоминания о вирусах и DoS/DDoS атаках, в основном типа Syn flood — первое упоминание о DDoS относится к 1996 году. Для защиты от этих угроз требовался инструмент, анализирующий и фильтрующий пакеты до их попадания на основной сервер. Одним из видов таких защит стали межсетевые экраны (firewall). Первое поколение данных решений относилось к типу пакетных фильтров (packet filters), которые обрабатывали пакеты по одному (не учитывая предысторию) и анализировали только уровни L1-L3 модели OSI и (для

протоколов TCP/UDP) номера портов из транспортного уровня L4 (см. рис. Для определения типа трафика (web, email и т.д.) использовался список фиксированных номера портов из каталога IANA. Процесс анализа заключался в сравнении данных, извлечённых из пакета, с набором заданных правил и, в зависимости от результата — блокировка или пропуск пакета в сеть с занесением события в журнал и опциональным уведомлением источника пакета о ситуации. Например, правило

«Блокировка Telnet трафика» выглядело, как правило, описывающее пакеты, транспортный протокол которых — TCP, номер целевого порта — 23, а действие при выявлении такого пакета — блокировка. Одним из первых подобных решений был продукт DEC SEAL.

Ближе к концу 90х — началу 2000х годов, в связи с ростом сетевых потоков данных, актуальными стали ещё две задачи, требовавшие сетевого анализа: балансировка нагрузки между серверами и ускорение работы отдельных видов сетевых приложений. К сетевым приложениям, требовавшим ускорения, относились, прежде всего, приложения, использующие протоколы HTTP, DNS, SSL. Для решения второй проблемы использовались, т.н. прокси-сервера, осуществляющие кэширование поступающих данных, минимизируя, так образом, обмены по сети.

Устройства, разработанные для решения обеих этих задач (инкапсулирующие,

в частности, функционал прокси-серверов) носили название контроллеры

доставки приложений (Application delivery controllers, ADC). Такие решения в частности были разработаны компаниями Alteon, Radware, F5,

Brocade, Cisco. В первой половине 2000х годов сетевые технологии получили бурное развитие

— появились средства голосового обмена по сети (VoIP) и обмена данными в

одноранговых сетях P2P (Napster, KaZaA), что, в частности, привело к очередному резкому скачку объёмов передаваемых по сети данных. Для развивающихся сетей крупных корпораций потребовалось объединять в единую локальную сеть территориально разнесённые площадки. Более частыми и сложными стали сетевые атаки, что требовало более развитых средств защиты.

Для реализации передачи управляющих сигналов и данных VoIP с использованием таких протоколов как SIP и RTP между различными провайдерами, как телефонной связи, так и интернета требовались специальные устройства – пограничные контроллеры сессий (session border controllers, SBC), которым требовалось выделять соответствующий трафик из общего потока. Данные устройства производились в таких компаниях как Acme Packet, Audiocodes, Cisco, Genband.

Для решения проблемы эффективного обмена данными между разными сегментами распределённой сети, соединёнными каналом ограниченной пропускной способности (данная проблема имеет название Channel optimization) был разработан целый спектр техник под общим названием Wan Optimizations. Среди этих техник можно указать:

- Дедупликация (Deduplication)– уменьшение повторной передачи данных за счёт сохранения на обоих концах обмена повторяющихся элементов данных и последующей передачи ссылок на эти данные вместо самих данных. Может осуществляться на разных

уровнях сетевого стека (в частности, TCP и IP)

- Сжатие (Compression) – передача данных по каналу в сжатом виде с последующим разжатием на другой стороне.
- Оптимизация латентности - упреждающая отправка сетевых пакетов-подтверждений TCP.
- Кэширование получаемого содержимого. Реализовывалось с помощью прокси-серверов, наиболее распространёнными из которых были Web-прокси, кэшировавшие содержимое сайтов. Примерами такого ПО являются Squid и NetCache.
- Объединение нескольких пакетов интенсивных сетевых протоколов, таких как CIFS, в один (protocol spoofing).

Данные техники впоследствии реализовывались как в виде отдельных сетевых устройств (Middleboxes), так и программно, на мощных серверах (Network appliances). Одним из первых производителей стала компания Riverbed, впоследствии купившая анализатор Wireshark и интегрировавшая его в свои продукты.

В сфере сетевой безопасности в этот период также произошли значительные изменения. Усложнение сетевых атак привело к тому, что их стало затруднительно с достаточной точностью определять по отдельным пакетам, а скорость появления новых атак — к необходимости реагирования на ещё неизвестные их виды. В совокупности это привело к появлению методов защиты на основе анализа поведения сетевых потоков (tcp session behaviour analysis). В то же время стали появляться вредоносные сайты, заражающие их посетителей, а также методы внедрения вредоносного функционала в не заражённые сайты. Для защиты от таких атак потребовалось внедрение обновляемых чёрных списков сайтов и необходимость фильтрации и блокировки по URL. Среди производителей средств защиты можно указать Arbor, BlueCoat, SonicWall.

Наиболее полное развитие технология анализа сетевого трафика получила, начиная со второй половины 2000х годов, в связи с несколькими факторами:

- Непрекращающийся рост объёмов передаваемых данных.
- Рост ширины каналов, обеспечивающих возможности для передачи этих объёмов.
- Увеличение количества разнообразия передаваемых данных, в частности тех, которые могут использоваться для составления различных профилей, как отдельных пользователей, так и различных групп.
- Рост как разнообразия сетевых угроз и атак,

так и их количественные характеристик.

Эти факторы привели к росту потребностей со стороны провайдеров интернета (internet service providers, ISP) и различных компаний. Интересы этих групп различны, но, в тоже время, имеют значительные пересечения.

Так, например, общей областью интересов является защита сетевых ресурсов, которая, в свою очередь, делится на ряд направлений:

- Антивирусные решения (AV).
- Развитые межсетевые экраны Next Generation Firewalls (NGFW).
- Системы обнаружения и предотвращения сетевых атак Intrusion detection/prevention systems IDS/IPS.
- Системы защиты от DDoS-атак.

В то же время, специфичной областью интересов провайдеров интернета является:

- Обеспечение качество связи в часы наибольшей нагрузки (ЧНН) с учётом экономии на расширении арендуемых каналов связи.
- Получение конкурентного преимущества за счёт возможности предлагать более выгодные индивидуальные тарифы с учётом индивидуального профиля пользования сетевым каналом.
- Регулирование полосы пропускания для некоторых видов трафика. Одной из основных проблем является P2P трафик, который, может

  занимать значимую часть арендуемого провайдером канала (до 60- 80%), приводя к тому, что чтобы обеспечить необходимое качество сервиса (quality of service, QoS) провайдеру приходится ускоренными (по сравнению с прогнозами роста абонентской базы и пользовательских потребностей) темпами расширять данный канал.

Основной областью интересов компаний, предлагающих свои товары и услуги с использованием Интернета, являются «профили» пользователей с точки зрения их интересов и предпочтений. Подобные профили можно опосредованно выявить, в частности, с помощью списка сайтов, которые пользователь посещает, набора его поисковых запросов, сетевых приложений, которые он использует.

К другой группе относятся компании, предоставляющие различные интернет сервисы, например, с помощью технологии виртуализации сетевых функций (Network Function Virtualization, NFV). К таким сервисам можно отнести:

- облачные сервисы,

- сервисы защиты,
- хранения и др.

Для этих компаний, специфичным является вопрос управления большими объёмами входящего трафика — требуется балансировка и интеллектуальное управление.

В соответствии с приведённым выше историческим развитием потребностей в области сетевых сервисов происходило развитие технологий анализа сетевого трафика, ложащихся в основу аппаратных, программных и гибридных решений.

Анализ сетевого трафика в режиме реального времени: обзор прикладных задач, подходов и решений

Направления развития технологий анализа сетевоготрафика

Можно выделить два основных направлений развития.

- Рост «глубины» анализа для отдельного сетевого пакета, то есть увеличение уровня модели OSI, данные которого подвергаютсяанализу.
- Полнота учёта состояния потока, к которому относится пакет, а также других потоков, связанных с данным.

В следующих разделах будут рассмотрены оба этих направления развития.

Глубина анализа сетевых пакетов

По этой «оси» технологии анализа трафика развивались последовательно, каждая последующая наследовала часть предыдущих механизмов и добавляла свои. Можно выделить три уровня развития

технологии, которые приведены на рис. 1.

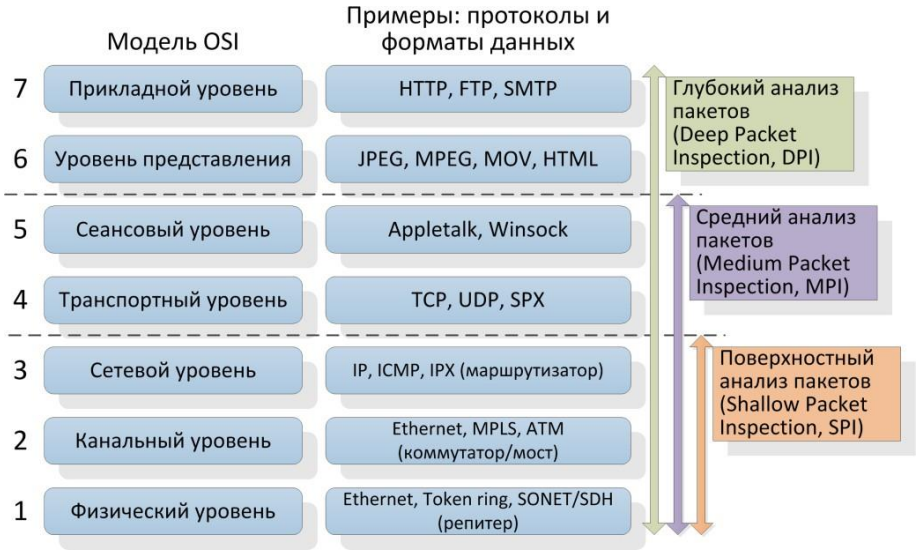


Рис. 1 – Уровни развития технологии анализа сетевого трафика по «глубине».

Рассмотрим эти уровни более детально.

Поверхностный анализ пакетов (SPI)

Технология анализа трафика, основывающаяся исключительно на заголовках пакета уровней L1-L3 по модели OSI. Предъявляет низкие требования к вычислительным ресурсам, что позволяет анализировать большие объёмы трафика. Технология широко распространена, на её основе работает большинство межсетевых экранов операционных систем, маршрутизаторов и других сетевых устройств. На её основе реализованы сетевые списки контроля доступа на уровне IP адресов и портов (Access Control List, ACL). Таким образом, данная технология хорошо подходит для разграничения доступа извне к отдельным компьютерам (IP) и сервисам

(порты) внутренней сети.

Средний анализ пакетов (MPI)

Технология анализа трафика, основывающаяся на инспектировании сессий и сеансов связи, инициированных приложением, но устанавливаемых шлюзом- посредником (см. рис. 2). Также применяется термин «прокси приложений» (application proxy). В рамках данной технологии содержимое пакетов анализируется частично и по предопределенным правилам. Не используются сложные методы анализа типа сигнатурного. Устройства, реализующие данный функционал размещаются между провайдером интернета и конечным

пользователем. Данные устройства разбирают заголовки вплоть до транспортного уровня и небольшую часть данных пакета для сопоставления разобранной части с некоторым списком разбора (parse list), с последующей реакцией в случае их обнаружения. Данные списки обычно короче списков ACL и предоставляют более широкий диапазон действий в отличие от

«разрешить/запретить» в случае ACL. Эти списки также более выразительны, так как позволяют привязываться не к IP-адресам, а к формату данных пакетов и данным некоторых протоколов уровня приложения, например, URL-адресам в случае протокола HTTP. С помощью MPI можно, например, заблокировать возможность получения flash-файлов или картинок с определённых интернет сервисов (на уровне представления OSI) или заблокировать часть команд (на уровне приложения OSI) в отдельных протоколах. Набор протоколов, как правило, очень ограничен. Например, в первых версиях CheckPoint FireWall-1 (CheckPoint FW-1) поддерживались протоколы Telnet, FTP, HTTP, а в Cisco Private Internet Exchange (Cisco PIX) - FTP, HTTP, H.323, RSH, SMTP и SQLNET. Впоследствии данные наборы незначительно расширялись. Межсетевые экраны, использующие данную технологию, относятся ко второму поколению.

Данная технология более гибкая в сравнении с SPI и, помимо разграничения доступа, подходит для большего числа задач — кэширование содержимого, анализ сжатого/шифрованного трафика, ограничение функционала отдельных протоколов путём запрета отдельных команд. Благодаря подключению в режиме прокси, может служить в качестве Wan Optimizer'a (см. выше).

Основной недостаток MPI — плохая масштабируемость: каждая команда и протокол требуют отдельного «шлюза» (входной-выходной порты). Кроме того, работа в режиме прокси сильно снижает скорость обработки. Для снижения нагрузки на прокси-сервер был разработан протокол ICAP, позволяющий прокси-серверам отправлять проходящие через них данные для проведения анализа сторонним серверам на предмет безопасности или анализа содержимого. Эта схема реализована в антивирусном продукте ClamAV, который может подключаться к прокси-серверам Squid и NetCache, упомянутым выше.

Эти факторы сильно ограничивают применение данной технологии на уровне провайдеров интернета вследствие необходимости анализа большого числа протоколов и команд на широких каналах связи.

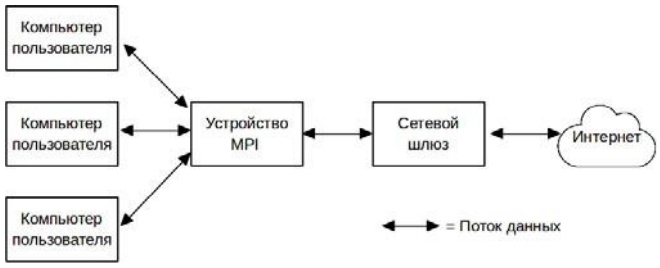


Рис. 2 - Схема применения устройств анализа на основе технологии MPI.

Глубокий анализ пакетов (DPI)

Иногда употребляют более узкий термин — DPP (Deep Packet Processing), который подразумевает такие действия над пакетами, как

модификация, фильтрация или перенаправление. Сегодня оба термина часто используются как взаимозаменяемые. Данная технология является логичным развитием MPI. В рамках данного подхода анализатор просматривает содержимое каждого пакета полностью. Одним из важных отличий от предыдущих технологий является то, что системы на базе DPI могут принимать решение не только по содержимому пакетов, но и по косвенным признакам, присущим каким-то определённым сетевым программам и протоколам. Для этого может использоваться статистический анализ. Например, анализ частоты встречи определённых символов, длин пакетов, расстояние между метками времени последовательных пакетов и т.д. Также, по сравнению с предыдущими подходами, значительно расширен список применений технологии: классификация, ограничение полосы, приоритезация, маркировка, кэширование и т. д. Технология DPI получила развитие, прежде всего, из-за стремительного роста вычислительных способностей процессоров, их быстродействия и, соответственно, возможностей для более полного и точного анализа сетевых данных.

В отличие от MPI, данная технология изначально разрабатывалась для высокоскоростной обработки и идентификации большого числа приложений в реальном времени. Таким образом, решения на основе DPI хорошо масштабируются как по ширине сетевого канала (известны решения, работающие на каналах порядка 100 Гбит/сек), так и по числу идентифицируемых приложений (в существующих решениях — порядка нескольких тысяч). С точки зрения реализации, основной компонент любого решения DPI - модуль классификации, отвечающий за классификацию сетевых потоков. При этом в зависимости от целей применения DPI, классификация может выполняться с различной точностью:

- тип протокола или приложения (например, Web, P2P, VoIP)
- конкретный протокол уровня приложения (HTTP, BitTorrent, SIP)

- приложение, использующее протокол (Google Chrome, µTorrent,Skype)

Важно отметить, что соответствие между классами различных уровней точности не однозначно, что показано на рис. 3.

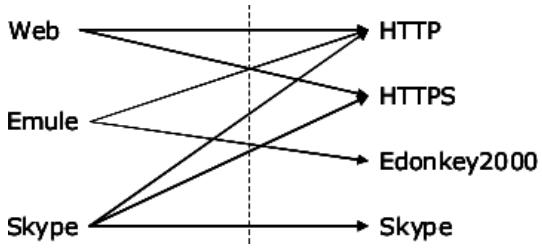


Рис. 3 - Различие между идентификацией приложений (слева) и протоколов (справа).

Технология DPI на данный момент является текущим стандартом де-факто для средств анализа сетевого трафика и относится к области критически важных технологий необходимых для обеспечения, как сетевой безопасности, так и требований законодательства. Вследствие этого в последнее время на международном уровне был принят ряд стандартов, требований и рекомендаций по особенностям реализации, внутреннему устройству и набору функций соответствующих средств. Эта технология редко применяется в межсетевых экранах — это скорее область IDS/IPS систем, в качестве исключений можно указать экраны Hogwash и Shield. Однако межсетевые экраны, относящиеся к четвёртому поколению, могут учитывать данные IDS/IPS систем в процессе анализа.

Учёт состояния потока при анализе сетевого трафика

Вторым направлением развития технологии анализа можно назвать учёт состояния протокола (потока) в процессе анализа — т.н. stateless/statefull виды анализа. Данное направление актуально только для протоколов, использующих транспортный протокол с установлением соединения (connection-oriented). Это означает, что перед любым обменом

командами и данными происходит процесс

«установления соединения», в ходе которого стороны обмениваются фиксированной последовательностью пакетов, которая часто называется «рукопожатием» (handshake), а после завершения обмена происходит аналогичный процесс «закрытия соединения». К connection-oriented протоколам, в частности, относится протокол TCP, но не UDP. Однако следует учесть, что поверх UDP может быть реализован другой транспортный протокол, с установлением соединения. В качестве примера можно привести протокол Quick UDP Internet Connections (QUIC) — протокол транспортного уровня с установлением соединения, использующий UDP. Из этого следует, что, в общем случае, нельзя полностью исключить statefull анализ для UDP пакетов.

Для описания различий описанных подходов требуется дать определение понятию «поток пакетов». Известны различные определения данного понятия. Часть из наиболее широко используемых приведена на сайте Center for Applied Internet Data Analysis (CAIDA).

«односторонний поток транспортного уровня» — последовательность пакетов передающихся с заданного IP-адреса и TCP/UDP порта на данный IP-адрес и TCP/UDP порт, с указанием протокола транспортного уровня (TCP/UDP). Таким образом, поток задаётся пятёркой <srcIP, srcPort, dstIP, dstPort, protocol>. С учётом данного определения, можно сформулировать отличие statefull от stateless подхода. Оно состоит в том, что в случае statefull подхода учитывается тот факт, к какому именно потоку относится анализируемый пакет, и результат (состояние) анализа предыдущих пакетов этого же потока, если данный пакет не первый. В случае если пакет первый — проверяется, что он является корректным пакетом установления соединения. Следует также отметить, что понятие «statefull» не вполне чёткое и может иметь разные градации с различным «состоянием», что приводит к различному балансу точность

анализа/ресурсоёмкость/скорость работы [24, 25]. Один из вариантов градации можно видеть на рис. 4. Список уровней учёта состояния потока, который там отражён – следующий:

- Анализ отдельных пакетов без учёта потоков и состояний (Packet Based No State, PBNS).
- Анализ пакетов в рамках потоков (Packet Based Per Flow State, PBFS).
- Анализ сообщений в рамках потока (Message Based Per Flow State, MBFS), т.е. произведена сборка IP-фрагментов в IP-пакеты (IP-нормализация) и сборка TCP-сегментов в TCP-сеансы (TCP-нормализация).
- Анализ сообщений в рамках протокола (Message Based Per Protocol State, MBPS), т.е. учитывается состояние автомата протокола (возможность принимать тот или иной тип сообщений). Пример автомата состояний протокола HTTP приведён на рис. 5. Вершины соответствуют состояниям, рёбра — условиям перехода, к которым могут относиться приём/отправка сообщения, результаты обработки сообщений, истечение таймаута.

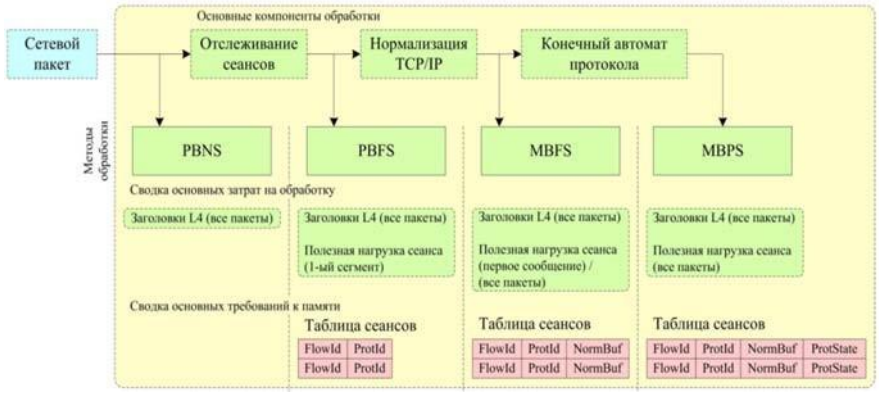


Рис. 4 - Градации полноты учёта состояния потока.

Базовые реализации технологии DPI часто относятся к stateless-

анализу, то есть анализ выполняется на уровне отдельных пакетов, состояние между анализом нескольких пакетов одного сетевого потока не сохраняется. Этого уровня точности хватает для многих практических приложений и позволяет значительно экономить ресурсы (см. рис. 4). В то же время, существуют задачи, для которых такого уровня точности недостаточно. В качестве примеров можно привести две технологии, использующие statefull подход — инспекция пакетов с хранением состояния (statefull packet inspection, SPI) и глубокий анализ содержимого (deep content inspection, DCI).

Анализ сетевых пакетов с учётом состояния потоков

В рамках SPI подхода, программа или устройство, которое его реализует, в момент открытия нового соединения проверяет его на соответствие заданной политике безопасности и до закрытия хранит параметры этого соединения в памяти. С помощью таких решений, в частности, осуществляется проверка корректности соединения, например отсутствие пакетов на открытом сетевом порте после завершения соединения. Реализации SPI содержатся в большинстве современных маршрутизаторов в виде SPI-брандмауэров. Также эта технология используется в программных межсетевых экранах, учитывающих состояние (stateful firewalls), компании CheckPoint и ряде IDS/IPS систем. Межсетевые экраны, использующие эту технологию, относят к третьему поколению. При данном подходе отслеживаются не только входящие и исходящие пакеты, но и состояние отдельных соединений, которое хранится в динамических таблицах. Благодаря этому при анализе очередного пакета могут учитываться не только заданные правила и политики по отношению к адресам и содержимому пакетов, но и состояние соединения, к которому относится пакет и предыдущих пакетов, которые к нему относятся, а также и других, связанных с данным, соединений. Классический пример преимущества межсетевого экрана поддерживающего состояние потока по

сравнению с межсетевыми

экранами без такой поддержки — обработка FTP протокола. Данный протокол открывает новый поток передачи данных на каждую соответствующую команду, причём поток открывается на случайном порте, большем 1024. Так как межсетевой экран не имеет возможности узнать, что новый поток относится к допустимому FTP протоколу — этот поток будет заблокирован. В случае наличия поддержки состояний потоков — адресная

информация нового потока будет добавлена в таблицу легитимных потоков и сессия будет пропущена в сеть.

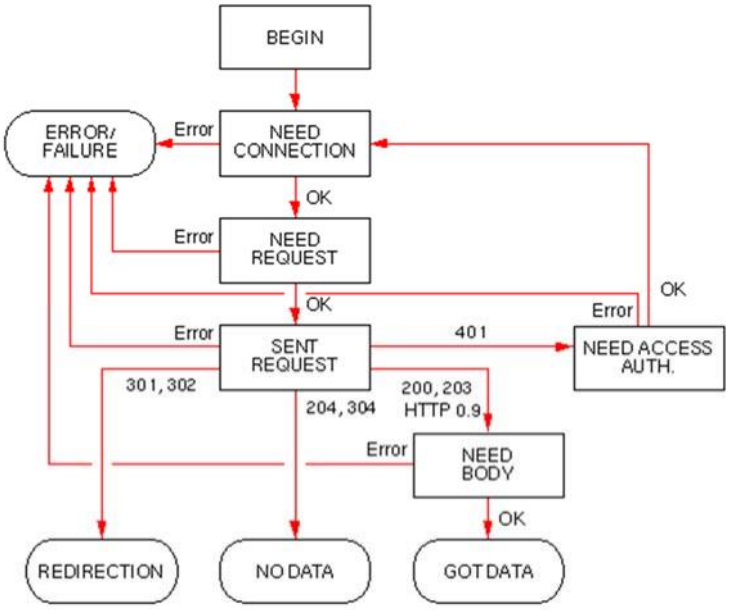


Рис. 5 - Пример автомата состояний протокола HTTP.

Анализ содержимого сетевых протоколов прикладного уровня

В рамках технологии DCI выполняется не только идентификация протокола конкретного сетевого потока, но и группировка потоков в группы, отвечающие за предоставление некоторого сервиса, например, сигнального протокола, например, SIP и протокола передачи данных, например, RTP, в случае VoIP. Также в процессе применения DCI, анализ не останавливается

на идентификации протокола, например, HTTP, но также делается попытка определить приложение, которое его использует (например, Gmail) и собрать контент этого приложения в том виде, в каком оно было передано приложением для отправки по сети (электронное письмо). Примером использования данной технологии может служить функция прослушивания VoIP звонков по перехваченному трафику в анализаторе WireShark.

С точки зрения функционала, основной вклад DCI в дополнение к модулю классификации (основной функционал DPI) — набор модулей разбора для различных протоколов прикладного уровня и различных видов данных в различных кодировках (например, MIME), которые они содержат. Функции модулей разбора, сводятся к двум основным:

1. Разбор буфера данных (сетевого пакета или собранной сессии), в соответствии с форматом сообщений протокола, описанным, как правило, на одном из специальных языков типа ASN.1 и P4.
2. Сборка сессий для протоколов с установлением соединения и их последующий разбор (пункт 1).

Одной из тенденций последнего времени в развитии средств DPI/DCI является универсализация и централизация анализа. Данная концепция может быть обозначена как «DPI как сервис» - под этим названием она была приведена в работе. Суть концепции заключается в том, что если в сети используется большое число различных средств, реализующих тот или иной анализ трафика(межсетевые экраны, системы IDS, оптимизаторы трафика и др.), то имеет смысл вынести весь анализ в отдельное устройство. Это устройство будет выполнять полный разбор сетевых данных и рассылать результаты анализа всем устройствам в зависимости от их потребностей, а те, в свою очередь, реализовывать только реакцию на поступающие данные. Переход к этой концепции в чём-то аналогичен переходу к программно-

конфигурируемым сетям (Software Defined Networks, SDN) в вопросах управления трафиком, при котором все решения по используемым алгоритмам маршрутизации и уровне её выполнения переходят от конкретных маршрутизаторов к выделенным устройствам - SDN-контроллерам. Такие подходы упрощают масштабирование систем и позволяют эффективно расширять функционал без дополнительных работ по интеграции и перенастройке оборудования.

Концепция «DPI как сервис» может быть эффективно реализована в рамках систем унифицированного управления угрозами (Unified threat management, UTM) и унифицированного управления безопасностью (Unified security management, USM). Эти системы также являются отражением тенденции централизации в виде объединения функционала межсетевых экранов, сетевых систем IDS/IPS, антивирусов, VPN-серверов, фильтров содержимого, балансировки нагрузки и предотвращения утечек данных в рамках единой системы.

Демонстрацией этих тенденций является выделение функционала распознавания протоколов и извлечения метаданных в виде отдельных модулей. Причём эти модули могут быть, как чисто программными, так и привязываться к некоторой аппаратуре. Примерами программных реализаций являются Qosmos Intelligence Engine, ipoque PACE, Windriver Content Inspection Engine, Procera PacketLogic Content Intelligence. Среди привязанных к аппаратуре модулей можно указать Cisco Network Based Application Recognition (NBAR) и Junos OS Next-Generation Application

Identification. Использование этих модулей в виде составной части систем контроля и управления трафиком позволяет формулировать политики безопасности и другие виды политик в гораздо более высокоуровневых терминах, например в терминах URL, имён приложений, отдельных функциональности в рамках этих приложений (например, блокирование передачи голоса в рамках Skype, при сохранении возможности

обмена текстовыми сообщениями). По сути, набор функций данных модулей аналогичен расширению функционала технологии MPI на произвольное множество протоколов, их команд и данных, которое поддерживаются конкретным модулем распознавания протоколов. Типичная схема использование такого решения [14] приведена на рис. 6, где «Внешний интерфейс» — решение типа «DPI как сервис», PCRF - Policy and Charging Rules Function — устройство, хранящее политики и правила, применяемые к трафику,

«Внутренний интерфейс» — устройство хранящее статистику, журналы, результаты применения правил к трафику, и т.д.

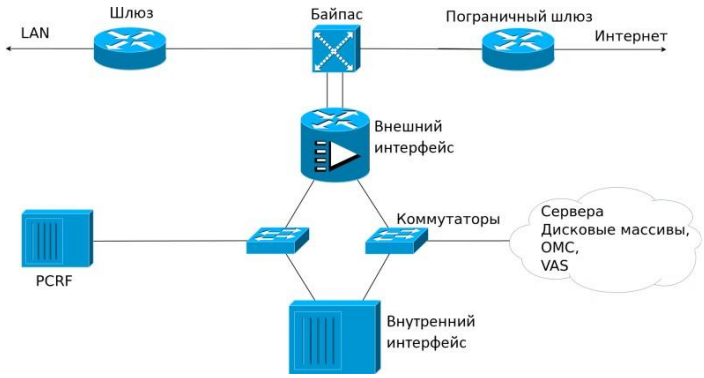


Рис. 6 - Схема использования системы DPI для применения политик к сетевому трафику.

Концепция «DPI как сервис» может также рассматриваться как отделение инфраструктурной части анализа сетевого трафика от бизнес-логики в рамках отдельных прикладных задач (сбор статистики, межсетевой экран, IDS/IPS системы и др.). В следующем разделе будет рассмотрена схема работы именно инфраструктурной части анализа, так как она является, с небольшими вариациями, идентичной в различных решениях для анализа сетевого трафика. В частности, будут выделены отдельные этапы анализа с кратким описанием их особенностей, а в последующих разделах каждый этап будет рассмотрен более подробно.

Анализ сетевого трафика в режиме реального времени: обзор прикладных задач, подходов и решений

Общая схема инфраструктурных алгоритмов анализасетевого трафика

Общая схема анализа сетевого трафика состоит из следующей последовательности шагов, каждый из которых приводит к повышению уровня представления объекта анализа.

1. Захват пакетов, проходящих через контролируемое сетевое соединение. Результатом данного шага является получение объекта анализа в виде сетевых пакетов. В зависимости от необходимой точности и скорости последующего анализа, а также доступных вычислительных мощностей могут использоваться различные подходы.

- Слайсинг (slicing), при котором анализу подвергаются не всё содержимое пакетов, а только некоторый префикс (n первых байт). В ряде исследований показано, что этот подход хорошо работает для последующей классификации трафика по протоколам. В частном случае, если перехватываемый размер равен суммарному размеру сетевых заголовков (L1-L3) является реализацией технологии SPI.
- Самплинг (sampling), при котором перехватываются не все пакеты, а только их часть, которая может выбираться по различным условиям, в зависимости от потребностей. В процессе развития технологии было предложено большое число стратегий отбора. Например, для задач мониторинга типов трафика подходит вариант

с выбором каждого n-го пакета (uniform sampling), где n может выбираться в зависимости от соотношения ширины канала и пропускной способности системы анализа. Задача получения информации о полном состоянии сети по результатам самплинга известна как inversion problem, в частности, при применении uniform sampling происходит недооценка среднего размера пакетов,так как чаще будут отбираться пакеты меньшего размера. Для передачи перехваченных данных используется протокол PSAMP.

- Наконец, для задач, в которых требуется максимально точный анализ трафика, например для систем обеспечения сетевой безопасности, требуется перехватывать все данные всего поступающего трафика без потерь — для обозначения этого подхода используется термин lossless capture или deep packet capture(DPC).

2. Агрегирование пакетов в потоки по некоторым адресным признакам (flow generaion), получение нового объекта для анализа — сетевого потока. Если при этом данные пакетов в дальнейшем анализе не учитываются, то такой вид анализа называется «анализ потоков» - flow

based analysis (в отличие от packet-based анализа, при котором анализируются данные пакетов). На рис. 7 показаны различия типичных схем packet и flow-based анализа. Flow-based анализ широко используется в силу значительно меньших требований к мощности вычислителя и пропускной способности, за счёт значительного снижения объёма данных для обработки. Такой вид анализа может выполняться как локально, так и удалённо от точки сбора данных. Для передачи собранных данных от точки сбора до точки анализа

используется большое число протоколов, часть из которых стандартизирована в виде IPFIX, а часть разработана отдельными производителями — Cisco NetFlow, Juniper Jflow. В рамках подхода записи, описывающие поток могут содержать разный набор данных. Наиболее общим набором таких данных является следующий:

- IP адреса источника и адресата,
- протокол транспортного уровня,
- в случае протоколов TCP/UDP — номера портов источника/адресата,
- набор счётчиков: количество переданных пакетов и байт, время создания и завершения потока.

Следует отметить, что хотя данный метод действительно значительно снижает требования к анализатору, тем не менее, он не является достаточно гибким, так как в отличие от слайсинга и самплинга не позволяет варьировать количество поступающих данных (оно зависит от входных данных). Более того в большинстве реальных задач количество потоков незначительно меньше количества пакетов (примерно на порядок) из-за большого числа очень коротких потоков, состоящих из нескольких пакетов — flash flows. Для решения этой проблемы было предложено использовать самплинг для потоков. Другой особенностью данного метода является то, что, вследствие ограниченности памяти, устройство, осуществляющее агрегацию пакетов, не может отслеживать один поток на протяжении произвольного промежутка времени. Для решения этой проблемы в конкретном решении обычно присутствует настройка, ограничивающая максимальную продолжительность потока (5 минут, в случае Cisco NetFlow). По истечении этого времени считается, что поток завершился, и информация о последующих пакетах агрегируется в рамках «нового» потока. Исследование точности flow-

based подхода и влияния этого эффекта на точность анализа содержится в работе. Также в этой публикации описан инструмент FLOW-REDUCE, осуществляющий «сборку» полной информации о потоке из фрагментов, на которые она была разбита из-за ограничений по времени.

3. Выполнение классификации по протоколу прикладного уровня или конкретному сетевому приложению. Результатом данной операции является получение нового объекта для анализа — сетевого потока конкретного протокола или приложения (в этом случае связанных потоков может быть несколько, например, в случае VoIP приложения это потоки SIP и RTP). После выполнения данной операции возможна следующая дополнительная обработка полученного объекта, конкретный вид которой зависит от решаемой прикладной задачи:

- разбор полей протокола (protocol parsing),
- сборка сессии протокола для протоколов с установлением соединения,
- извлечение данных приложения (content extraction) — страниц сайтов (HTML), файлов различных типов (исполняемые, изображения, текстовые документы, и т.д.), электронных писем, аудио-видео потоков и т. д.,
- разбор данных приложения (application content parsing).

Рис. 7 - Различия типичных схем packet (слева) и flow-based (справа) анализа.

Для полноты картины, следует сказать, что помимо указанных выше packet- based и flow-based подходов существует ещё один источник данных о сетевом трафике — т.н. база управляющей информации (Manage Information Base, MIB)

Модули для накопления, хранения и обмена данными в формате MIB реализованы в большинстве устройств. Передача данных осуществляется по протоколу SNMP. Данные получаемые таким путём имеют низкий объём и неспецифичны для протоколов. Например, в рамках данного подхода, можно получить сведения об общем количестве пакетов и байт прошедших через конкретный сетевой интерфейс конкретного сетевого устройства.

Следует сказать, что одной из причин развития MIB и flow-based подходов, несмотря на их сравнительно низкую точность, послужила до сих пор идущая глобальная дискуссия о законности и допустимости глубокого анализа

трафика с точки зрения нарушения безопасности, прав на частную жизнь и т. д. На данный момент одним из следствий данной дискуссии является, в частности, то, что в научных работах, трафик, который подвергается глубокому анализу предварительно проходит процедуру «анонимизации» с помощью специальных средств.

Далее будут более подробно рассмотрены отдельные шаги из приведённой общей схемы анализа сетевого трафика, методы, алгоритмы и подходы, а также их особенности и ограничения применимости.

Захват сетевых пакетов

Программные и аппаратные средства, осуществляющие захват трафика, относятся к классу сниферов (sniffers). Для решения задачи захвата трафика могут использоваться как стандартные серверные сетевые карты, так и специализированные сетевые карты, предназначенные для перехвата трафика на предельных скоростях без потерь. Специализированные карты, как правило, реализованы на базе FPGA или ASIC и имеют встроенные средства для проставления временных меток, аппаратной фильтрации, снятия некоторых заголовков низкоуровневых протоколов, балансировки нагрузки между процессорами на многопроцессорных компьютерах с учётом IP-потоков, выявления ошибочных и дублирующихся пакетов. При этом вся обработка (в том числе и копирование данных в память компьютера из памяти сетевой карты) осуществляется без привлечения ресурсов ЦПУ. По мере развития технологий многие из описанных свойств реализуются и на базе стандартных сетевых карт. Технология реализации таких дополнительных функций носит название TCP Offload Engine (TOE). Она включает в себя следующие различные технологии, базовыми из которых являются следующие:

- Large Segment Offload (LSO) или Giant send offload (GSO)— сегментация больших TCP-пакетов при отправке
- Large Receive Offload (LRO) — сборка приходящих отдельных сетевых пакетов в большие сегменты
- Checksum Offload — проверка контрольных сумм в заголовках

IPv4, IPv6, TCP и UDP

- IP Security (IPSec) Offload — шифрование/дешифрование трафика протокола IPSec

Основной проблемой для стандартных сетевых адаптеров является не скорость передачи данных, как таковая, а количество пакетов в единицу времени. Это обусловлено особенностями внутренней реализации обработчиков пакетов на сетевых картах, драйверов сетевых карт и программных сетевых стеков ОС. Вследствие этого, стандартные сетевые карты без специализированных драйверов и сетевых стеков не обеспечивают перехват трафика без существенных потерь на скоростях более 3 Mpps (миллионов пакетов в секунду). Причины такого ограничения будут рассмотрены ниже. Ещё одной проблемой является точное проставление временных меток.

Проблемы, возникающие при переходе к сетевым соединениям, поддерживающим более высокие скорости передачи данных, связаны в основном с несколькими факторами:

- Ограниченной пропускной способностью аппаратуры.
- Архитектурными ограничениями при взаимодействии аппаратуры с ОС и ОС с пользовательскими приложениями.
- Объёмом памяти, необходимым для хранения получаемых данных.

Большинство распространённых систем анализа трафика работают, используя библиотеки Libpcap (ОС Linux) и WinPcap (ОС Windows). Данные библиотеки работают в пользовательском режиме. Для обеспечения своей работы со стороны ОС они используют драйверы уровня ядра Berkeley Packet Filter (BPF)

и Netgroup Packet Filter (NPF) соответственно. Основная разница между этими драйверами заключается в схеме их работы с буферами памяти, использующимися для временного хранения пакетов, получаемых от сетевой карты. Драйвер BPF использует схему с двойной буферизацией, в то время как драйвер NPF использует кольцевой буфер.

Среди проблем этих решений, приводящих к снижению

производительности можно выделить.

- Двойное копирование данных пакета (из карты в память ядра, из памяти ядра в память пользовательского процесса).
- Большое число прерываний от сетевой карты (на каждый пакет, чтобы он был скопирован в буфер ядра).
- Большое число переключений между режимами ядра и пользователя (на каждый пакет при его копировании в память пользовательского процесса).
- Недостаточное использование параллелизма на уровне отдельных ядер и процессоров (по умолчанию все прерывания обрабатываются одним ядром).
- Проблемы с синхронизацией при доступе к данным из нескольких потоков выполнения. В случае, если полученные данные должны обрабатываться в несколько потоков между этими потоками возникает ситуация соревнования за ресурсы.

В зависимости от количества копирований данных пакетов, которые выполняются в процессе перехвата, решения разделяются следующим образом.

- **0-copy (zero-copy).** Для реализации подхода с нулевым копированием требуется аппаратная поддержка со стороны сетевой карты – она должна содержать собственный DMA контроллер, копирующий данные с карты в память программы пользователя, без дополнительного копирования через память ядра. Примером может служить библиотека PF_RING ZC в связке с сетевыми картами Intel или Napatech
- **1-copy.** Для реализации этого подхода возможны несколько

вариантов — разработка анализатора на уровне ядра, что является весьма сложной задачей или прямое отображение памяти ядра в память пользовательского процесса.

- **2-copy.** Стандартное решение на базе LibPcap или WinPcap.

Для решения перечисленных проблем было реализовано некоторое количество специализированных драйверов и сетевых стеков, к которым относятся, например, коммерческое решение Sniffer10G от Emulex и Myricom, а также открытая разработка PF_RING компании Ntop. Эти решения используют схему с кольцевым буфером, как более эффективную, а также оптимизированы для многопроцессорных и многоядерных компьютеров. В частности они реализуют следующий функционал:

- Обработка перехвата пакетов с использованием большого числа нитей исполнения (одна нить на входную очередь).
- Балансировка нагрузки между ядрами (одно ядро – одна входная очередь).
- Пакетная фильтрация внутри сетевой карты.

Для реализации этих функций используется как аппаратная поддержка со стороны архитектуры, так и поддержка со стороны ОС (специализированное API). Среди используемых технологий можно выделить следующие.

- Набор близких технологий Interrupt Moderation, Adaptive Interrupt Moderation, Interrupt Coalescing, Interrupt Blanking, Interrupt Throttling, позволяющих управлять задержкой доставки прерываний за счёт настраиваемого таймера и обрабатывать получение/отправку множества пакетов за одно прерывание.
- MSI-X — распределение I/O прерываний по нескольким процессорам и ядрам.

- New API (NAPI) — интерфейс уровня ядра ОС Linux, позволяющий применять технику уменьшения количества прерываний (interrupt mitigation) со стороны сетевых устройств.
- Receive-side Scaling (RSS) — технология, предоставляющая возможность динамической балансировки нагрузки входящих сетевых пакетов по нескольким ядрам и процессорам (прерывания поступают на разные процессоры). Существуют реализации для масштабирования на случаи более 64 процессоров. Данная технология поддерживается в семействе ОС Windows с появлением Scalable Networking Pack. В ОС Linux аналог этой технологии называется Linux Scalable I/O.

Также существует ряд аппаратных технологий от различных производителей процессоров, предназначенных для ускорения ввода/вывода.

- Intel Integrated I/O - технология прямого подключения шины PCI Express 3.0 к процессору (без отдельного PCI-контроллера), реализованная в семействе Intel Xeon E5.
- Direct Cache Access (DCA) – предоставление устройствам ввода/вывода, таким как сетевые адаптеры, возможности помещения данных напрямую в кеш процессора Intel.

Группировка сетевых пакетов в потоки

Группировка пакетов в потоки — достаточно стандартная и простая операция. Основное отличие разных реализаций данного функционала связано с тем, какие именно поля адресной информации и как использовать для идентификации потока. Наиболее употребляемое определение потока было дано ранее. Так как оно использует 5-ку полей как ключевую

информацию для определения принадлежности конкретного пакета к конкретному потоку, то для его обозначения и обычно используют термин 5-tuple. Также иногда используются двусторонние потоки, симметричные к перестановке пар <srcIP, srcPort> и <dstIP, dstPort>. Модуль, отвечающий за группировку пакета обычно называют генератором потоков (flow generator). В процессе работы данный модуль хранит в памяти отображение соответствующей ключевой информации на данные конкретных потоков. При появлении нового пакета, с ним производятся следующие операции.

1. Из пакета извлекается ключевая информация, позволяющая идентифицировать, к какому потоку он принадлежит.
2. Производится поиск по текущему множеству потоков.
3. Если поток найден – в данных потока увеличиваются соответствующие счётчики – как правило, к ним относятся время жизни потока, количество пакетов и байт в потоке. Если поток не найден - создаётся новая запись потока и в неё добавляется информация о текущем пакете.

В работе проведена оценка вычислительных ресурсов, необходимых для выполнения первых двух операций, а также для операции классификации (в случае использования детерминированных конечных автоматов). Результаты оценки приведены в Табл. 1. Абсолютные цифры, приведённые на рисунке, на данный момент могут быть не вполне актуальны, но их ценность, прежде всего, в относительной стоимости операций.

Таблица 1 - Оценка скорости выполнения основных операций при анализе трафика

Операция	Стоимость

	(тактыпроцессора)
Извлечение идентификатора потока	78
Поиск/добавление идентификатора потока	49
Поиск сигнатуры с помощью детерминированного конечного автомата (мин., ср., макс.)	13-4331-8900

Описанная выше базовая схема, хоть и является корректной, но неполной. Она содержит существенный недостаток — предполагается, что модуль располагает бесконечной памятью, так как отсутствует определение условий завершения потока и поэтому непонятно, когда следует удалять запись о потоке из отображения. В случае транспортного протокола с установкой соединения (например, TCP) в этом протоколе предусмотрена явная процедура завершения соединения (обмен FIN-ACK пакетами или посылка RST пакета). В случае протоколов без установления соединения (например, UDP) такой подход не работает, поэтому, как правило, используется один из вариантов, основанных на использовании таймера — например, обрыв соединения через 5 мин (такой вариант используется в коммутаторах Cisco NetFlow). Этот же подход используется для слишком долгих TCP-потоков.

Классификация сетевого трафика

Тема классификации сетевого трафика сама по себе является очень обширной. Прежде чем переходить к методам, которыми она осуществляется, перечислим варианты классификации по её результатам, то есть объектам, которые получаются на выходе данного алгоритма, их свойствам и возможностям их дальнейшей обработки. По этому критерию, можно выделить три основных варианта классификации. Далее они

перечислены в порядке увеличения «точности» классификации:

- **Тип трафика** не является достаточно содержательным способом классификации и, как правило, или не подвергается дальнейшему анализу, или подвергается достаточно простой дополнительной уточняющей классификации. В зависимости от сферы применения,типы могут быть различными. Среди примеров, можно указать:
  - P2P, видео-стриминг, веб-трафик — в случае систем сборастатистики и мониторинга,
  - трафик сетевой атаки/нормальный трафик — в случае системзащиты от сетевых атак,
  - трафик, содержащий/не содержащий объекты копирайта, вслучае систем контроля копирайта.
- **Использующийся протокол прикладного уровня (protocol identification)** является достаточно содержательным и может, как использоваться непосредственно — например, в системах сбора статистики и мониторинга для повышения уровня точности. Основным способом дальнейшей обработки является разбор протокола, включающий два основных функции — сборка сессии прикладного уровня, в случае необходимости извлечение данных протокола из отдельных его полей (метаинформация уровня протокола).
- **Приложение, передающее данные (application identification),** дает максимально детализированный уровень классификации. На этом уровне могут осуществляться те же виды обработки, что и на уровне протокола прикладного уровня, а также

извлекаться и интерпретироваться данные (метаинформация) конкретного приложения, что соответствует более высокому уровню их представления. Например, поле типа «строка», определённое на уровне протокола, может соответствовать «имени пользователя» на уровне приложения.

В различных прикладных задачах результаты идентификации протоколов и приложений могут интерпретироваться и, соответственно подвергаться различной последующей обработке (как и в случае идентификации типа трафика).

Например, в случае системы защиты от вредоносного кода, под протоколом может пониматься командный (command-and-control, C&C) протокол ботнета, а под приложением — конкретный вирус. Соответственно, извлекаемая метаинформация — команды ботнета, передаваемые им данные, а цель анализа

— выяснение его функционала, оценка распространённости и исследование возможностей его деактивации.

В случае системы составления профиля пользователя для последующей демонстрации таргетированной рекламы (например, iMarker) в роли протокола может выступать HTTP, в роли приложения — браузер, а объектом анализа является запрос пользователя к поисковой системе, который подвергается дальнейшему текстовому анализу для извлечения ключевых слов.

Выбор конкретной прикладной задачи может значительно влиять как на выбор алгоритма классификации, так и на его параметры и производительность. В качестве примера можно рассмотреть следующее сравнение. В случае системы статистики, алгоритм классификации обычно работает последовательно на пакетах каждого потока «до первого срабатывания». Схема такой классификации приведена на рис. 8.

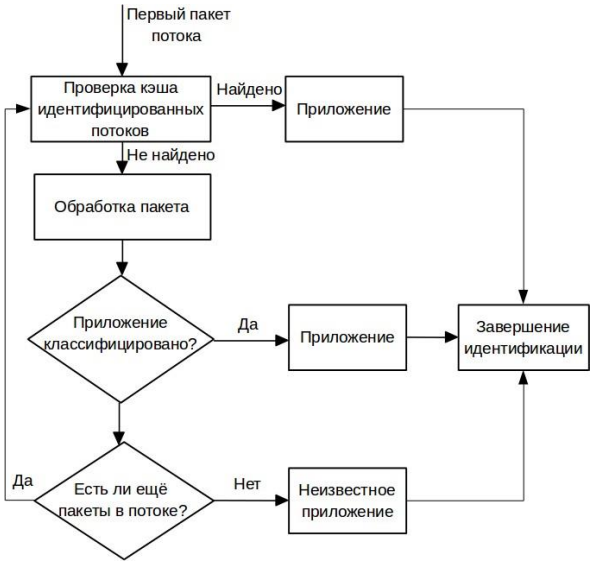


Рис. 8 - Схема классификации «до первого срабатывания».

В случае систем фильтрации, по ключевым словам, такой метод не подходит, так как в одном и том же сетевом потоке, в различных пакетах могут встретиться различные слова и, с точки зрения системы классификации, в этом случае данный поток попадёт сразу в несколько классов.

В общем случае, очевидно, что первый подход гораздо производительнее, так как приходится анализировать значительно меньшие объёмы данных. Кроме того, в ряде подходов, для дополнительного ускорения, анализируют не всё содержимое пакета, а только некоторый его префикс (по аналогии со слайсингом). Например, в работе, для идентификации потоков, содержащих шифрованные и сжатые данные, используются только первые 16 байт пакетов.

В работе проведена оценка влияния размера анализируемого префикса пакета на точность классификации по протоколам и скорость работы классификатора на трёх снятых сетевых трассах Unibs-GT, Polito, Polito-GT. Результаты приведены на рис. 9, где на левом графике ошибки

классификации

обозначены как misclassified, а трафик, который не удалось классифицировать, как unknown.

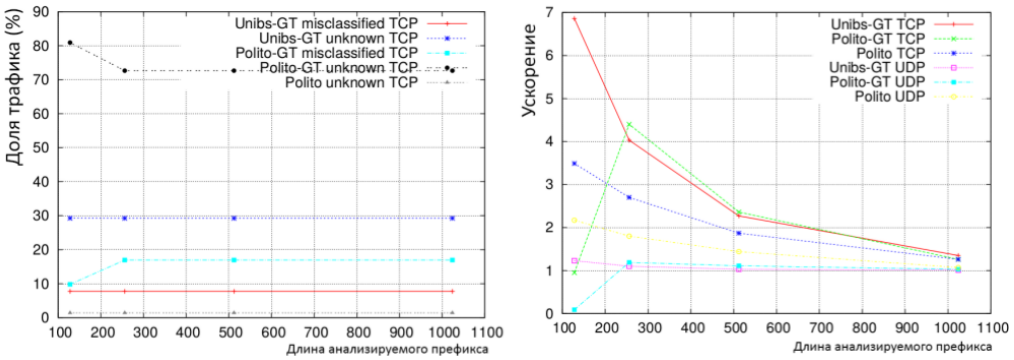


Рис. 9 - Оценка влияния длины префикса на точность классификации (слева) и скорость (справа).

На основе этих исследований, в частности делается вывод об избыточности проведения IP-дефрагментации и TCP-нормализации при решении данной задачи, так как данные алгоритмы (особенно второй) достаточно ресурсоёмки и практически не влияют на точность. Это происходит из-за того, что для классификации, как правило, используется не более 256 байт пакетов, а минимальный размер фрагмента обычно не меньше 576 байт. То есть, для данной задачи PBFS подход более предпочтителен, чем подход MBFS (см. рис. 4).

Рассмотрев виды классификации по получаемым результатам и подходы в разных прикладных задачах, перейдём к рассмотрению конкретных алгоритмов классификации.

Классическим подходом к классификации является анализ содержимого пакетов (payload-based). При этом, как правило, выполняется поиск т.н.

«сигнатур» (signature-based подходы) - характерных признаков, которые заранее создаются для каждого приложения или их групп. Классификация может выполняться как на уровне отдельных пакетов

(stateless анализ), или может учитываться состояние потока (statefull анализ). Для повышения точности распознавания часть подходов использует уточнённые «сигнатуры» на основе автоматов состояний протоколов (см. рис. 5). При таком подходе, получаемые сообщения, после их классификации, сопоставляются с переходами в различных автоматах протоколов, и оценивается корректность последовательностей таких переходов. Эта группа подходов называется Stateful Protocol Analysis Detection.

Как было показано на рис. 9, классификация является наиболее нагруженным алгоритмом анализа сетевых пакетов. Исторически, из-за нехватки вычислительных мощностей, предпринимались попытки достижения увеличения производительности алгоритма за счёт выбора источника данных, используемых алгоритмом в процессе классификации, таким образом, чтобы обрабатываемые данные, будучи не менее информативными, чем содержимое пакетов, были бы более компактны. Эта группа подходов (в отличие от

«сигнатурного») относится к классу «основанных на выводе» (inference-based). Одним из важных преимуществ inference-based подходов является то, что качество анализа не зависит от представления данных в сетевых пакетах, в частности, отсутствуют ограничения при анализе сжатого/шифрованного трафика. Далее будут рассмотрены основные подходы к решению задачи классификации, их особенности и ограничения применимости.

Подходы на основе вывода

Все подходы на основе вывода можно разделить на группы по двум основным параметрам:

- используемые для вывода данные,
- используемый для их анализа алгоритм. Все виды

данных, в свою очередь, можно разделить на:

- характеристики отдельных пакетов в рамках отдельного потока

(packet based),

- характеристики потоков в целом (flow based).

К первой группе относятся подходы, использующие такие характеристики как: временные промежутки между пакетами, последовательности размеров пакетов, и др.

Ко второй группе относятся два основных подхода.

- Подход на основе анализа портов (port-based) при котором идентификация происходит по одному из номеров портов потока, на основе базы данных о характерных статичных портах, которые используют зарегистрированные в IANA протоколы (регистрировать можно любой номер порта, а не только первые 1024). Этот метод считается малоэффективным, так как на данный момент существует большое число протоколов с динамическими номерами портов. В частности, к таким протоколам относятся практически все реализации P2P. Кроме того, часто используются схемы, при которых трафик некоторого протокола (например, HTTP) передаётся по нехарактерному для него номеру порта (не 80 в случае HTTP).

- Подходы на основе статистической информация об активности отдельных хостов в сети: в скольких и каких именно обменах данными (потоках) участвовал данный хост, сколько данных, и в какую сторону передавалось и т.д. Эти данные сопоставлялись с набором заранее созданных шаблонов различных видов серверов. Один из таких подходов описан в работе [57].

Алгоритмы анализа данных делятся на два основных направления:

- сравнение с тем или иным видом заранее созданного шаблона,
- подход на основе машинного обучения и последующего распознавания.

Методы на основе машинного обучения в последнее время получили бурное развития. Одной из причин этого развития является доступность большого числа разнообразных данных для обучения (социальные сети, крупные БД, результаты поисковиков и т.д.). Эта группа методов на данный момент представлена большим числом алгоритмов: байесовские сети, деревья принятия решений, методы опорных векторов, методы k-средних и др. Данные методы, в свою очередь делятся на группы по методу обучения [58], который применяется для их конфигурирования:

- классификация (обучение с учителем),
- кластеризация (обучение без учителя),
- ассоциирование (association),
- численное предсказание (numeric prediction).

Методы на основе сигнатур

Недостатком этих методов является их высокая ресурсоёмкость, связанная с необходимостью просмотра больших объёмов данных. Однако в настоящее время вычислительные мощности позволяют использовать более точные, чем основанные на выводе, сигнатурные методы, которые, в свою очередь, делятся на две большие группы:

- поиск строк (string matching)
- поиск регулярных выражений (regexp matching).

Сигнатуры на основе строк.

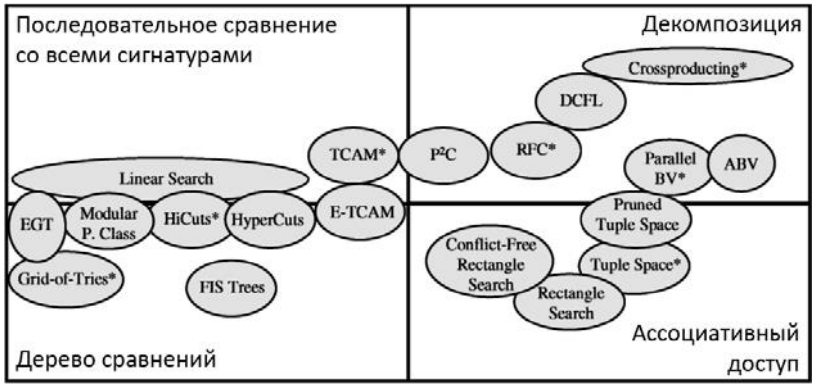
В процессе развития, для поиска строк применялось большое число

различных алгоритмов поиска строк, обладающих различными преимуществами и недостатками, что определяло область их применения [59,60]. Наиболее известными алгоритмами являются: прямой перебор (brute force, BF), Кнут- Морис-Пратт (KMP), Бойер-Мур (BM), Ахо-Корасик (AC), AC-BM (использующийся в Snort), Wu-Manber, Commentz Walter (CW), фильтры Блума(вероятностная структура на основе хеша).

В работе проводится обзор и сравнение большого числа методов поиска строк по тому как реализован алгоритм сравнения с имеющимися сигнатурами. Выделено 4 группы методов.

- Последовательное сравнение со всеми сигнатурами (Exhaustive Search).
- Дерево сравнений (Decision Tree).
- Декомпозиция (Decomposition), при которой отдельные части сигнатура обрабатываются независимо, с последующим объединением результатов.
- Ассоциативный доступ (Tuple Space), при котором сигнатуры разбиваются на группы бит, с которыми проводятся операции сравнения.

На рис. 10 приведено распределение большого числа алгоритмов по данным группам. Алгоритмы, лежащие на границах, используют гибридные



подходы.

Рис. 10 - Распределение алгоритмов поиска строковых сигнатур по данным группам.

Сигнатуры на основе регулярных выражений.

С ростом числа протоколов и их сложности строковое представление было признано недостаточно выразительным, в связи с чем, для описания сигнатур стали использовать регулярные языки в виде грамматик и регулярных выражений. Для эффективного поиска сигнатур регулярный язык, описывающий сигнатуру, представляются в форме конечного автомата. Выделяют два основных вида автоматов - детерминированные или недетерминированные. Оба эти представления имеют свои достоинства и недостатки.

Одна из открытых баз сигнатур такого вида используется в открытом приложении для классификации l7-filter. Кроме того, такие подходы могут не срабатывать в случае, если сигнатура была разделена на несколько пакетов на уровне IP или TCP. Для решения этой проблемы, перед поиском сигнатуры необходимо выполнить IP-дефрагментацию и TCP-нормализацию соответственно.

Основным достоинством недетерминированных конечных автоматов (НКА, NFA) является их компактность: объем занимаемой памяти пропорционален числу символов, входящих в регулярные выражения. Однако для обработки каждого символа входных данных недетерминированным конечным автоматам может потребоваться до O(N) обращений к памяти, где N – число состояний автомата. По этой причине возможности применения НКА в высоконагруженных системах ограничены.

В свою очередь, детерминированные конечные автоматы (ДКА, DFA) требуют для каждого входного символа совершить единственное обращение

к памяти. Их использование может представлять трудности в связи с их большим

размером: число состояний ДКА может экспоненциально расти («экспоненциальный взрыв»), и ограничено $O(2^l)$, где l – суммарная длина регулярных выражений в каноническом представлении. В работе было проведено исследование влияния разных типов регулярных выражений на рост размеров автомата. Результаты показаны на рис. 11. Было выделено 3 типа регулярных выражений, с точки зрения их влияния на размер автомата:

- выражения, привязанные к началу пакета (поиск осуществляется только в начале пакета);
- выражения, привязанные к началу пакета и содержащие звёздочку Клини (*);
- выражения, не привязанные к началу и содержащие звёздочку Клини (*).

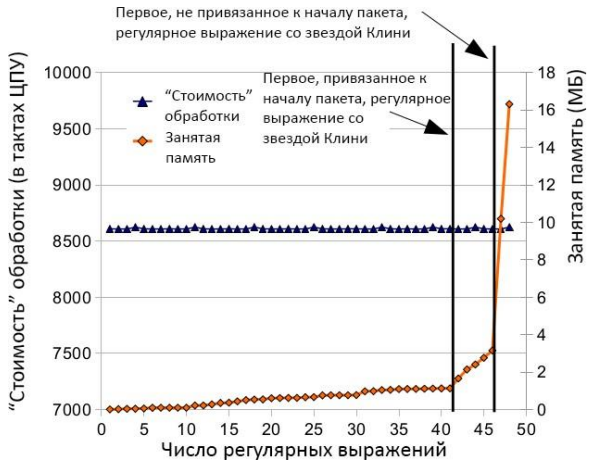


Рис. 11 - Экспоненциальный взрыв размера DFA при добавлении регулярных выражений со звёздочкой Клини.

Для снижения размеров автоматов часто применяют различные виды сжатия. Такие автоматы носят название сжатые ДКА (Compressed DFA, cDFA). В табл.

2 приведено сравнение трёх основных видов автоматов по размеру и

производительности поиска, взятое из работы [38]. Данные автоматы были построены по регулярным выражениям классификатора L7. Для предотвращения экспоненциального роста размера, детерминированные автоматы были разделены на 4 части.

Таблица 2 - Сравнение размеров и скорости работы основных видов конечныхавтоматов.

Алгоритм	Стоимость в тактах ЦПУ(мин, ср., макс.)	Количество автоматов	Размер автомата
НКА	$2.2*10^4$, $4.1*10^7$, $8.9*10^7$	1	509 Кб
ДКА	52, $2.5*10^4$, $3.6*10^4$	4	230 Мб
Сжатый ДКА	268, $1.2*10^5$, $1.7*10^5$	4	53 Мб

Несмотря на проблемы с требованиями к памяти, детерминированные конечные автоматы (и их модификации) получили намного большее распространение в высокоскоростных системах анализа.

Современные системы анализа трафика предъявляют высокие требования, как к скорости обработки данных, так и к количеству регулярных выражений, задействованных в обработке и, соответственно, размеру итогового автомата. Так как ни ДКА, ни НКА не могут удовлетворить одновременно требования и по скорости, и по размеру памяти, в настоящее время ведется большое количество исследований по разработке гибридных представлений. С точки зрения реализации, автоматы представляют собой таблицы из состояний, в каждой ячейке которых находится список возможных переходов из этого состояния в другое. Поэтому два основных направления работ сосредоточены на уменьшении числа состояний и переходов соответственно. В качестве примеров, можно

привести представления D^2FA и δFA, реализующие сжатие переходов и группу представлений MDFA, H-FA, XFA и Dual FA, реализующих различные подходы к сокращению числа состояний.

Анализ данных в разных представлениях

Одну из важных проблем для классификаторов на основе содержимого представляет тот факт, что одни и те же данные (например, строка) могут быть при передаче по сети быть закодированы по-разному, в зависимости, например, от используемого протокола. В частности, под «различными представлениями» в данном разделе имеются следующие аспекты.

- Различные методы кодировки, в частности для текстовых данных — ASCII и Unicode кодировки, а для бинарных данных — различные транспортные кодировки, например представление в виде текста (binary-to-text), примером которых является Base64.
- Сжатия данных для уменьшения загруженности каналов передачи данных, например использования gzip и deflate алгоритмов для сжатия содержимого HTTP-сообщении.
- Шифрование данных для обеспечения безопасности, например использование криптографических алгоритмов RC4 и AES в протоколах SSL/TLS.

По данным различных исследований, сжатый и шифрованный трафик (иногда используется общий термин «непрозрачный», opaque) составляет всё большую долю от всех сетевых потоков данных. Это является следствием большого числа факторов, таких как:

- рост популярности онлайн видеосервисов, использующие сжатие видеопотоков,
- распространённость P2P-сервисов, которые в большинстве

своёмиспользуют шифрование,

- использование шифрованного соединения (HTTPS) по-умолчаниюна многих популярных сайтах,
- внедрение сжатия в HTTP протоколе на многих Web-серверах.

Проблема классификации этих видов трафика имеет несколько аспектов.

- Для корректной классификации такого трафика требуется дополнительный функционал.
- Попытка классифицировать такой трафик «в лоб» существенно снижает общую производительность классификатора, так как приходится просматривать все данные пакетов, проходя по большой части автомата и при этом результат почти наверняка будет отрицательным. То есть такой трафик представляет собой

«худший случай», характеристики работы на котором алгоритмов классификации существенно хуже средних (см. табл.2).

Для решения первого аспекта проблемы используются несколько подходов:

- Генерация копий сигнатур, которые подвергаются различным видам сжатия и кодирования. Данный метод ограничен только некоторыми алгоритмами сжатия и кодирования, а также плохо масштабируется с учётом роста количества алгоритмов сжатия и ихколичества их параметров.
- Использование модулей, осуществляющих разжатие/перекодировку данных перед их классификацией. Этот метод имеет такие же ограничения, как и предыдущий и также плохо масштабируется. Кроме того этот метод увеличивает

уязвимость системы к атакам типа zip bomb, при которых размер разжимаемых данных превосходит размер сжатых на несколько порядков.

- Установка системы анализа на месте или после средства, осуществляющего разжатие/расшифрование данных. Пример такого средства - прокси-сервер.

Для устранения второго аспекта, требуется подавать на модуль классификации трафика только «прозрачный» трафик, для чего из всего трафика требуется предварительно отфильтровать «непрозрачную» его часть. Для решения этой задачи разработаны алгоритмы, большая часть которых использует характерное свойство «непрозрачного» трафика — повышенную энтропию значений его отдельных байт.

Классификация угроз

В реализациях DPI, связанных с безопасностью (например, IDS/IPS), где классификация применяется не для идентификации протоколов, а для классификации атак и угроз, разработаны подходы, специализированные под соответствующие задачи. Одним из таких подходов является статистическое выявление аномалий, когда вначале производится обучение системы на трафике, не содержащем атак («нормальном»), а затем на реальном трафике детектируются отклонения от «нормальной» картины. Такие подходы называют «статистическое детектирование аномалий» (statistical anomaly-based detection). На основе этой техники работают многие IDS и защиты от DDoS атак.

Анализ сетевого трафика в режиме реального времени: обзор прикладных задач, подходов и решений

Требования, предъявляемые к современным средствам анализа содержимого сетевого трафика

Принимая во внимание приведённый обзор основных алгоритмов и схем анализа сетевого трафика можно сформулировать ряд функциональных и нефункциональных требований, предъявляемых к современным системам анализа сетевого трафика. Все требования можно разделить по подсистемам, к которым они применяются и отдельно выделить те, которые относятся ко всей системе в целом:

4. Система в целом.

- Поддержка масштабирования по пропускной способности анализируемого канала передачи данных.
- Минимизация числа перестановок пакетов в рамках отдельных потоков.
- Возможность встраивания дополнительных средств предобработки сетевых пакетов перед их передачей подсистеме классификации (перекодировка, разжатие).

5. Подсистема перехвата данных.

- Поддержка разбора всех протоколов ниже сетевого уровня, встречающихся в контролируемом канале (MPLS, VLAN и т.д.) Это необходимо, для обеспечения попадания всех пакетов одного потока в одну очередь обработки при выполнении балансировки нагрузки (хеширование должно выполняться на уровне IP-пакета).
- Использование кольцевого буфера для хранения обрабатываемых пакетов и режима zero-copy, при наличии поддержки со стороны сетевой карты, или 1-copy, при отсутствии такой поддержки для экономии ресурсов центрального

процессора.

- Для эффективного использования ресурсов многопроцессорных имногоядерных машин требуется поддержка того или иного вида RSS-технологии (управления прерываниями и их распределения по разным ядрам).

6. Подсистема агрегации пакетов в потоки.

- Поддержка возможности задания типа ключевой информации, по которой определяется принадлежность пакета к потоку, для обеспечения гибкости при использовании подсистемы для решения различных прикладных задач.

- Максимизация количества одновременно обрабатываемых потоков и времени жизни каждого отдельного потока в условиях ограниченных ресурсов памяти.

- Для обработки сжатых данных необходима возможность одновременного отслеживания потока, который представлен как всжатом, так и в разжатом виде.

- Встроенная защита от атак типа «zip-бомба».

- Отслеживание факта связанности потоков (например, потока управления и потока данных в случае FTP), в частности, для уточнения классификации.

7. Подсистема классификации.

- Сложность алгоритма поиска сигнатур должна быть не хуже чем линейной по входным данным «в среднем», а желательно и «в худшем» случае для устойчивости системы к целенаправленным атакам.

- Расширяемый набор «сигнатур» для поддержки

новых протоколов, их групп и сетевых приложений.

- Хорошая масштабируемость по памяти при росте количества «сигнатур».
- Возможность предварительного разделения трафика на «прозрачный» и «непрозрачный» с целью снижения нагрузки на данную подсистему.
- Возможность анализа данных, представленных в различных кодировках.

Прежде чем перейти к рассмотрению конкретных реализаций отдельных компонентов систем анализа и того, насколько они удовлетворяют перечисленным выше требованиям, будут рассмотрены существующие способы подключения этих систем к сетям передачи данных и влияние конкретных видов подключения на точность работы систем анализа и ограничение применимости этих систем.

Классификация систем анализа по способу подключения к сети передачи данных

Одна из важных характеристик систем анализа сетевого трафика — способ получения и конкретный вид данных для анализа, то есть вопрос подключения к некоторому «каналу» связи. С точки зрения особенностей подключения, подсистемы получения данных можно разделить на следующие классы:

8. Распределённые системы. Схема представляет собой набор сборщиков данных о сетевом трафике (probes) и его накопителях и набор его анализаторов (collectors), которые получают данные от сборщиков. При такой схеме наиболее актуальным становится вопрос

знания топологии сети и точек получения трафика в рамках этой топологии системой анализа и учёт этих знаний при выполнении анализа. К такимсистемам относятся системы:

- системы пассивного сетвого мониторинга (FlowMon, Ntop),
- системы анализа поведения (Network Behaviour Analysys, NBA),
- системы обнаружения аномального поведения (Network Behaviour Anomaly Detection, NBAD),
- системы управления событиями информационной безопасности (Security Information and Event Managment, SEM, SIM, SIEM).

9. Беспроводные системы, в том числе мобильные. Особенностью данных систем является то, что «канал» типа точка-точка отсутствует и при достаточном уровне сигнала можно перехватывать беспроводные коммуникации в достаточно большом радиусе. Названия систем анализа, подключаемых таким образом, обычно начинаются с префикса «wireless» — например Wireless firewall и Wireless intrusion prevention system (WIPS).

10. Локальные системы. В данном классе подключение осуществляется к конкретному каналу передачи данных (сетевому кабелю). В зависимости от места сетевого канала, к которому осуществляется подключение, в общей топологии сети можно выделить следующие подклассы:

- Конечный пользователь — подключение осуществляется на уровне сетевой карты конкретного пользователя. Названия систем анализа, подключаемых таким образом, обычно начинаются с префикса

«host-based» — например host-based application firewall и host-basedintrusion prevention system (HIPS). В свою очередь сам перехват данных в этом случае может осуществляться как на уровне сетевого стека с помощью соответствующего низкоуровнего API (PF_RING, NAPI), так и путём перехвата (hook) системных вызовов. Последний вариант является более медленным и используется в т.н. software application firewalls, к которым относятся большинство встроенных в ОС межсетевых экранов. Их отличием является то, что анализ

выполняется не по потокам, а по процессам, которые участвуют в сетевом обмене. Минусом таких решений является их уязвимость из-за неполной изоляции процессов и возможности переполнения памяти. На данный момент более продвинутой технологией является использование «песочниц» (sandbox) с дополнительной изоляцией процессов, к котором относятся например AppArmor и TrustedBSD MAC framework.

- Шлюз — подключение осуществляется в точке, которая является единственным выходом в глобальную сеть (WAN) для некоторой локальной подсети (LAN). В случае установки на одном из нескольких шлюзов может наблюдаться ситуация с

«односторонними потоками», когда, например, исходящий трафик некоторого соединения идёт через один шлюз, а входящий — через другой. Подобная ситуация, называется «сетевой ассиметрией» (network assymetry) и также может иметь место при работе со спутниковыми каналами связи (см. рис. 12). Такие условиязначительно повышают требования к системе анализа и применяемым алгоритмам (в частности усложняется TCP- нормалиация в отсутствие ACK-пакетов). Названия систем анализа, подключаемых таким образом, обычно начинаются с префикса

«network-based» — например network-based application firewall и network-based intrusion prevention system (NIPS). Подключение при этом

может выполнено по одной из трёх схем:

- Зеркалирование (mirroring), при котором весь сетевой поток канала дублируется — первая копия идёт непосредственно в сеть, а вторая отправляется на вход системы анализа (см. рис 7, левая часть). При такой схеме в отсутствие обратной связи от системы анализа может выполняться только пассивный анализ, то есть система не может влиять на трафик, попадающий в сеть (например, путём фильтрации или приоритезации). Само зеркалирование также может технически выполняться тремя способами:
  - Кабельный сплиттер или сплиттер волокна, в случае оптики, позволяет дублировать пакеты, без какого либо влияния на них (в том числе и на задержку).
  - Пассивное оборудование, выполняющее т.н. зеркалирование портов (port mirroring) при котором данные с одного (или более) входных портов копируются на несколько выходных портов. Данный метод может вносить незначительные задержки. Также могут возникать потери пакетов из-за недостаточной производительности оборудования.
  - Активное оборудование, также выполняющее зеркалирование портов, но также реализующее некоторое решение, выполняющее анализ трафика, например

межсетевой экран или приоритезацию. Основной недостаток

— вносимая задержка.

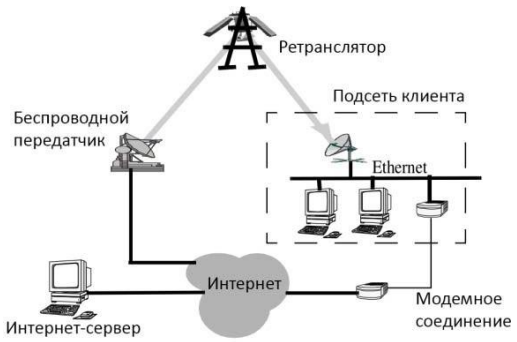


Рис. 12 - Пример «ассиметричных соединений» при использовании спутниковых каналов связи.

- Проксирование (proxy), при котором весь поток направляется на систему, которая становится посредником во всех взаимодействиях WAN и LAN (см. рис. 2). Плюсом такой схемы является

«прозрачность» (transparency) трафика для системы анализа — например, система может проводить декомпрессию и расшифрование сжатого и шифрованного трафика, соответственно с последующей компрессией и отправкой получателю. Минусом является большая ресурсоёмкость проксирование и сравнительно большая задержка в доставке сетевых пакетов, вносимая анализом. Также такое подключение вносит дополнительные риски — при выходе системы анализы из строя в результате поломки или целенаправленной сетевой атаки локальная сеть остаётся без связи с глобальной.

- Байпас (bypass). Гибридный подход между зеркалированием и проксированием, решающий проблему выхода из строя системы анализа. В нормальных условиях система работает в режиме проксирования, однако при её выходе из строя или получении от неё соответствующего сигнала, соединение с LAN выполняется напрямую. На данный момент такой вид подключения наиболее

распространён для решений, которым требуется активный анализ, то есть возможность изменять содержимое передаваемого трафика в соответствии с некоторыми политиками.

В следующем разделе будут подробно рассмотрены особенности реализации программных и программно-аппаратных систем анализа сетевого трафика.

Классификация высокоскоростных средств анализа содержимого сетевого трафика

Основные виды обрабатывающих элементов приведены на рис. 13.

Рис. 13 - Классификация основных видов высокоскоростных DPI-систем.

Как видно из рисунка, все решения можно разделить на три большие группы.

11. Специализированные аппаратные решения, представляющие собой

«чёрный ящик», являющийся, как правило, системой на чипе (System- on-a-Chip, SoC). Такое решение может содержать самые различные компоненты, позволяющие ускорить анализ сетевого трафика — ПЛИС (Field-Programmable Gate Array, FPGA), интегральные схемы специального назначения (Application-Specific

Integrated Circuit, ASIC), сетевые процессоры (Network Processors, NP), бинарную (content addressable memory, CAM) и троичную (ternary content addressable memory, TCAM) ассоциативную память или их комбинации, например. Особенности конкретных аппаратных систем следующие.

- ASIC — одно из наиболее высокоскоростных решений. Минусом является то, что вся программа обработки, включая сигнатуры, закладывается в устройство на этапе производства и впоследствии не может быть изменена.
- FPGA — устройства с достаточно высокой пропускной способностью, которые могут перепрограммироваться при необходимости изменить алгоритм обработки или набор сигнатур, хотя процесс обновления прошивки может занимать значительное время. Данные устройства достаточно неудобны в непосредственном программировании, поэтому существует некоторое количество промежуточных представлений и средств

трансляции/компиляции, генерирующих на выходе FPGA-программу. К их числу можно отнести, прежде всего, OpenCL и более высокоуровневые среды: NetCOPE от InveaTech и G от NetFPGA. При реализации классификации на FPGA устройствах обычно используют алгоритм КМР.

- CAM — специальный вид ассоциативной памяти, который выполняет параллельное сравнение всего своего содержимого с поступившим на вход значением и возвращает адрес, значение которого совпало с входным. Скорость доступа достаточно высока и составляет 4 нс, а сложность поиска составляет *O*. Однако данный вид памяти не может выполнять поиск наибольшего префикса, что существенно для большинства решений DPI, поэтому она подходит

только для поиска строк фиксированной длины. Существуют различные реализации CAM, в том числе bitwise CAM (BCAM) — ассоциативная память на основе дерева, в которой строки представляются в виде булевских формул.

- TCAM — также специальный вид ассоциативной памяти, который помимо данных хранит три вида логических значений (0, 1 и ? - «не важно»). Также, в случае сигнатур, они упорядочены по убыванию. В результате над данной памятью можно эффективно проводить операцию поиска наибольшего префикса. Данная память получила широкое распространение в сетевых устройствах, в частности на её основе в роутерах и свитчах выполняется трансляция между именами и IP-адресами (IP address lookup или DNS lookup). Пример работы этого вида памяти приведён на рис. 14. Несмотря на общую эффективность, данный вид памяти имеет ряд недостатков:
  - высокая стоимость (в десятки раз дороже SRAM);
  - неэффективность хранения (меньшая ёмкость);
  - большое энергопотребление (до 180 раз по сравнению с SRAM);
  - плохая масштабируемость на длинные входные данные.
- На данный момент существуют реализации, эмулирующие функционал TCAM на SRAM памяти E-CAM, Z-CAM и др. (полный обзор таких реализаций приведён в [78]), преодолевающие часть из перечисленных недостатков ценой незначительного снижения производительности.
- Также ряд недостатков связан напрямую с применением в DPI

решениях.

- Т.н. «Range Representation Problem» - в TCAM легко хранить префикс, который требуется искать только в начале «слов», однако чтобы эффективно искать тот же префикс в середине

«слов» требуется гораздо больше ячеек TCAM.

- Т.н. «Multi-match Classification Problem» - в результате сравнения возвращаются все ячейки, с которыми сравнение дало

положительный результат, а не только более приоритетные, что создаёт дополнительную вычислительную нагрузку.

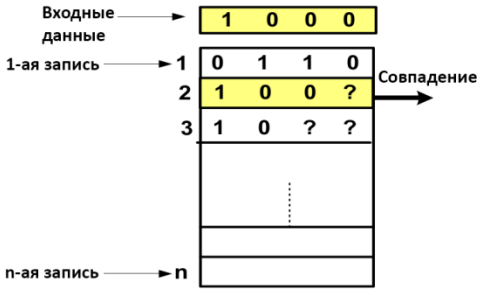


Рис. 14 - Пример работы TCAM-памяти.

- NP — специализированный вид процессоров для обработки сетевого трафика. Основные функции включают: поиск шаблонов, поиск по ключам, обработка полей протоколов, управление очередями, быстрые операции выделения и изменения буферов памяти. Архитектура использует конвейерную и суперскалярную обработку, многоядерность, специализированный набор инструкций (микрокод). Примером может служить IXP1200, использовавшийся для реализации

первой IDS CardGuard на базе сетевых процессоров.

12. Чисто программные решения, использующие в качестве аппаратной платформы, стандартные высокопроизводительные (обычно- многопроцессорные) серверы со стандартными серверными сетевыми картами. Для обеспечения полноты перехвата на высоких скоростях используются:

- Проприетарные разработки, использующие высокопроизводительные реализации сетевого стека, напримерSniffer10G в случае Emulex и Myricom.
- Открытые разработки, такие как Intel DPDK и Ntop PF_RING (Intel,Mellanox, Chelsio, Qlogic\Broadcom).

13. Программно-аппаратные решения, которые с одной стороны, также как и чисто программные решения, используют в качестве основного

«вычислителя» стандартную серверную платформу, но некоторые вычисления реализуют на специализированных устройствах. К таким устройствам относятся.

- Сетевые карты на базе технологий FPGA (Napatech, Endace, InveaTech, Accolade, Fiberblaze, Telesoft Technologies), ASIC (titanic systems, Hitech global) или многопроцессорных систем (Tillera).
- Специализированные вычислители на базе технологий FPGA,

устанавливаемые в PCI-слот.

- Использование GPU-карт для отдельных видов вычислений. Известны решения для задачи маршрутизации, т.н. Software Routers и для задачи классификации по регулярным

выражениям. Основой проблемой при использовании GPU, является «узкое горлышко» при передаче данных из основной памяти в память GPU для обработки. Для решения этой проблемы со стороны производителей GPU были предложены решения на основе прямого отображения памяти компьютера в память GPU с использованием DMA, что позволяет освободить ресурсы CPU от обработки операций копирования. У разных производителей эти технологии имеют разные названия — GPUDirect у Nvidia и DirectGMA у AMD. Известны примеры использования этих технологий для ускорения обработки сетевого трафика. Примерная схема работа данных технологий приведена на рис. 15 справа, по сравнению с режимом по умолчанию (слева).

Рис. 15 - Схема потока сетевого трафика без применения (слева) и с применением (справа) технологий GPUDirect и DirectGMA.

Масштабирование системы анализа

Важным практическим вопросом при использовании любого средства анализа трафика является вопрос масштабирования при увеличении ширины анализируемого канала. Распространённый способ решения - добавление в схему ещё одного устройства обработки трафика. Однако при этом также требуется решать задачу распределения сетевых

пакетов из исходного канала по устройствам обработки трафика. Для решения этой задачи в случае широких исходных каналов используются специальные сетевые устройства — пакетные брокеры или балансировщики, основной задачей которых является распределение сетевых пакетов, полученных из набора входных сетевых интерфейсов по набору выходных сетевых интерфейсов. В зависимости от логики, применяемой при этом выборе, можно выделить статическую и динамическую балансировку.

При статической балансировке стандартными схемам коммутации являются:

- One-to-One – каждый входной порт отображается на отдельный выходной;
- Any-to-Any - любой входной порт на любой выходной;
- Many-to-One - агрегация данных с нескольких входных портов в один выходной;
- One-to-Many - зеркалирование трафика на несколько потребителей; При динамической балансировке, важной функциональностью является возможность разбора заголовков различных протоколов низкого уровня (таких как Ethernet, VLAN, MPLS и др.). Это важно прежде всего для того чтобы была возможность передавать пакеты, относящиеся к одному потоку транспортного

уровня на один сетевой интерфейс, то есть к одному устройству обработки трафика. В частности, это позволяет корректно выполнять TCP-нормализацию. Также это позволяет уменьшить эффект перестановки пакетов в рамках одного потока, что, в свою очередь, повышает эффективность алгоритмов IP- дефрагментации и TCP-нормализации. Наличие такого функционала говорит о том, что сами балансировщики являются аппаратным средством сетевого анализа. Кроме того, многие из них также поддерживают аппаратную фильтрацию по содержимому

пакетов (обычно поиск осуществляется по префиксу данных пакетов некоторой фиксированной длины), что говорит о том, что балансировщики потенциально относятся к DPI решениям.

Пропускная способность пакетных брокеров варьируется от десятков до сотен гигабит в секунду. Среди производителей пакетных брокеров можно упомянуть cPacket, netoptics, arista.

Выводы

Из приведённого обзора можно сделать ряд выводов. Можно констатировать как количественный (в связи с ростами объёмов трафика и ширины каналов связи), так и качественный рост (в связи с новыми прикладными задачами) потребностей в средствах анализа трафика. При этом, несмотря на огромное многообразие конкретных решений, реализующих различные виды анализа, в основе большинства этих решений лежит примерно одинаковая схема, что хорошо видно на примере внедрения концепции «DPI как сервис». В вопросах реализации систем анализа можно отметить следующий ряд тенденций:

- Развитие специализированных аппаратных средств (NP, TCAM и т.д.), позволяющих обрабатывать потоки данных в каналахмаксимально достижимой пропускной способности.
- Развитие программных и программно-аппаратных технологий (NAPI, RSS, MSI-X, DCA и т.д.), позволяющих обрабатывать потоки данных порядка 10 Гбит/с на стандартных серверных платформах.
- Перенос (offloading) значительной части сетевой обработки непосредственно на сетевые карты.
- Появление специализированных сетевых стеков (в том числе с открытым исходным кодом), позволяющих

осуществлять перехват без потерь на каналах 10 и более Гбит/с. Это, в свою очередь, позволяет реализовывать достаточно мощные сетевые анализаторы на базе стандартных компонент, без использования дорогостоящих специализированных сетевых карт на базе FPGA и ASIC.

- Активные исследование в области переноса задачи классификации сетевого трафика на GPU в связи с её высокой ресурсоёмкостью и ограниченным количеством вычислительных ресурсов центрального процессора, даже на многопроцессорных системах.

Также можно видеть, что для каждого элемента общей схемы анализа в научном, техническом и IT-сообществах осуществляется поиск оптимальных решений под конкретные прикладные задачи. В научном сообществе решаются задачи поиска оптимальных алгоритмов, например для решения задачи классификации, являющейся наиболее ресурсоёмкой частью любой системы анализа. В техническом сообществе осуществляется разработка аппаратных средств, позволяющих обеспечить возможность решения прикладных задач на каналах с постоянно растущей пропускной способностью. В IT-сообществе осуществляется синтез решений, предлагаемых двумя другими сообществами, подбор конкретных параметров алгоритмов для отдельных прикладных задач и поиск баланса, который сводится, по сути, к решению оптимизационных задач в условиях большого числа переменных, среди которых можно упомянуть:

- необходимую полноту анализа;
- необходимую точность анализа;
- требуемую глубину анализа;
- цена конкретного программно-аппаратного решения;
- производительность этого решения;

- гибкость решения и его возможности по масштабированию инаращиванию функционала.

Squid - прокси сервер

Squid - это полнофункциональное приложение кэширующего прокси сервера, которое предоставляет сервисы кэширования и прокси для HTTP, FTP и других популярных сетевых протоколов. Squid может осуществлять кэширование и проксирование SSL запросов и кэширование результатов DNS поиска, а также выполнять прозрачное кэширование. Squid также поддерживает широкий набор кэширующих протоколов, таких как ICP (кэширующий интернет протокол), HTCP (гипертекстовый кэширующий протокол), CARP (протокол кэширования маршрутизации) и WCCP (кэширующий протокол перенаправления контента).

Прокси сервер Squid - это великолепное решение широких требований к кэширующему и прокси серверу, которое масштабируется для сетей от уровня регионального офиса до корпорации, когда обеспечивается расширяемый разделяемый механизм контроля доступа и отслеживания критических параметров через протокол SNMP. Когда выбираете компьютерную систему для использования в качестве Squid прокси или кеширующего сервера, убедитесь, что ваша система оснащена большим количеством оперативной памяти, поскольку Squid поддерживает кэш в памяти для увеличения производительности.

Используется в UNIX-подобных системах и в ОС семейства Windows NT. Имеет возможность взаимодействия с Active Directory Windows Server путём аутентификации через LDAP, что позволяет использовать разграничения доступа к интернет ресурсам пользователей, которые имеют учётные записи на Windows Server, также позволяет организовать «нарезку» интернет трафика для различных пользователей.

Используется вместе с движками Mediawiki на wiki хостингах. Использование кэширующего прокси-сервера для сайтов становится выгодно примерно с 2000 посетителей в сутки.

Совместимость

Сервер Squid развивается в течение уже многих лет. Обеспечивает совместимость с большинством важнейших протоколов Интернета, а также с операционными системами:

AIX

BSDI

FreeBSD

Linux

HP-UX

IRIX

Mac OS X

Microsoft Windows

NetBSD

NeXTStep

OSF и Digital Unix

OpenBSD

SCO Unix

SunOS/Solaris

Squid используется и в системе Wikimedia.

Описание архитектуры

Схема комплекса LAMP, работающего вместе с сервером Squid. Высокопроизводительное и отказоустойчивое решение для веб-сервера во враждебном окружении.

Списки контроля доступа

Для контроля доступа к ресурсам и определения ряда действий используются списки контроля доступа. Каждый ACL может состоять из нескольких критериев (но только одного типа):

- адрес (сеть) источника запроса, цели запроса
- имя (доменное имя) источника запроса, имя цели запроса
- части URL запроса
- протокол
- порт (получателя, отправителя, самого squid’a)
- метод (POST или GET) при передаче данных по HTTP
- браузер (User-agent)
- ident (запрос к рабочей станции)
- Номер автономной системы отправителя/получателя (не для всех случаев)
- Авторизация на прокси-сервере
- Номер соединения (чаще всего используется для ограничения количества соединений)
- SNMP
- сертификаты пользователя
- параметры запроса
- внешние обработчики

Идентификация

Squid поддерживает несколько видов идентификации пользователей:

- По IP-адресу (или доменному имени узла)
- По переданным реквизитам (логин/пароль)
- По идентификатору пользовательского агента (браузера)

Для идентификации по логину/паролю возможно использовать:

- Обычные логин/пароль
- NTLM-авторизацию
- Внешние программы авторизации (определяющие формат авторизации)

Squid имеет возможность переписывать запрашиваемые URL. Squid может быть сконфигурирован так, чтобы пропускать входящие URL через процесс редиректора выполняемого как внешний процесс (подобно dnsserver), который возвращает новый URL или пустую строку, обозначающую отсутствие изменений.

Редиректор не является стандартной частью пакета Squid. Редиректор предоставляет администратору контроль за передвижениями пользователей. Использование редиректора в сочетании с прозрачным проксированием даёт простой, но эффективный контроль, над доступом к нежелательным ресурсам (например к развлекательным ресурсам и социальным сетям в корпоративной сети, порнографии). Программа-редиректор должна читать URL (один на строку) со стандартного входа и записывать изменённые URL или пустые строки на стандартный выход. Нужно заметить, что программа-редиректор не может использовать буферизированный ввод-вывод. Squid дописывает дополнительную информацию после URL, которую редиректор может использовать для принятия решения.

SAMS (SQUID Account Management System) — программное средство для администрирования доступа пользователей к прокси-серверу Squid.

На данный момент SAMS настраивает работу редиректоров:

Редиректор SAMS — редиректор, работающий напрямую с базами SAMS

SquidGuard — очень мощный редиректор.

Стандартный SQUID — простейший редиректор, описанный в документации к SQUID

Виды редиректоров:

Редиректор SAMS

Написан специально для SQUID, напрямую использует информацию, содержащуюся в базе данных. Позволяет включить различное перенаправление запросов для пользователей (регулируется шаблонами пользователей). Редиректор SAMS обеспечивает:

- ограничение доступа пользователей к SQUID
- контроль времени доступа пользователей к SQUID
- ограничение доступа пользователей к запрещённым ресурсам (или доступ пользователей только к разрешённым ресурсам)
- перенаправление запросов пользователей к баннерам, счётчикам и т. п.

Редиректор SquidGuard

Мощный редиректор с большими возможностями. В состав редиректора входят списки баннерных, порно и пр. доменов. SAMS добавляет в файл конфигурации SquidGuard squidguard.conf настройки на списки запрещённых доменов и перенаправления доступа SAMS. Настройки на списки, идущие с SquidGuard не изменяются и не удаляются.

При использовании редиректора SquidGuard в файл squid.conf заносятся acl, разрешающие доступ всех пользователей к SQUID. Ограничение доступа пользователей организовано средствами редиректора.

Стандартный SQUID

Этот редиректор описан в документации на SQUID. Написан на perl. Редиректор создаётся после подачи команды на реконфигурирование SQUID, на основе списков перенаправления запросов. Быстрый и лёгкий редиректор, но не различает пользователей.

При использовании этого редиректора, ограничение доступа пользователей по спискам запрета доступа организовано с использованием ACL SQUID. При использовании редиректора SQUID или если редиректор не используется вовсе, то существует возможность — при отключении пользователей за превышение трафика у них остаётся доступ к URL и IP адресам, прописанным в списке «Локальные домены».

Ограничение максимальной скорости соединения

Ограничение максимальной скорости получения пользователем (пользователями) в squid реализовано с помощью механизма англ. delay pools (дословно — «пулы задержки»). Механизм ограничения скорости работает по принципу бассейна (откуда и название pool (бассейн)), в который «втекает» и «вытекает» информация. Отдельные конфигурируемые подобным образом области памяти называются англ. bucket (ведро). У ведра есть параметры: «ёмкость», «скорость наполнения». Если пользователь (пользователи) получают информацию на скорости ниже, чем «скорость наполнения», то ведро всегда полно. Если пользователь кратковременно поднимает скорость получения информации выше скорости наполнения, то до момента, пока ведро не пусто, он не ограничивается по скорости, как только ведро становится пустым, клиент получает информацию со скоростью наполнения ведра. В случае наличия групповых и индивидуальных вёдер, они включаются последовательно.

Существует три типа (класса) delay pools:

Единое ведро (англ. aggregate bucket, class 1) ограничение на общую потребляемую полосу для всей группы. (параметры: ёмкость бассейна, скорость наполнения).

Единое ведро с автоматическим формированием индивидуальных вёдер (англ. single aggregate bucket as well as an "individual" bucket, class 2). Индивидуальные вёдра формируются из битов IP-адреса (с 25 по 32).

Единое ведро, сетевые вёдра и индивидуальные вёдра (англ. single aggregate bucket as well as a "network" bucket and a "individual" bucket, class 3). Сетевое ведро формируется по битам 17-24 IP-адреса.

Для каждого ведра указываются два параметра: ёмкость и скорость наполнения. −1 означает «без ограничения».

Попадание пользователей в то или иное ведро определяется списками доступа к вёдрам, они просматриваются в порядке упоминания в файле конфигурации до первого совпадения. Пользователи, не попадающие ни в одно из вёдер, в скорости не ограничиваются.

Обратное кэширование

Одной из особенностей squid является возможность работать в режиме обратного прокси (reverse proxy), также известного как «ускоритель» («HTTP accelerator»). В этом случае вместо кэширования запросов нескольких пользователей к множеству сайтов, кешируются запросы множества пользователей к нескольким сайтам. В этом режиме принятый запрос проверяется на «динамичность» (нужно ли каждый раз обрабатывать запрос с нуля) и «возраст» (актуальны ли ещё данные). Если данные ещё актуальны и не поменялись, то запрос не передаётся серверу, а отдаётся из кеша squid’a. Таким образом существенно снижается нагрузка на серверы (например, в Википедии запросы к страницам кешируются, так как от просмотра их

содержимое не меняется, при этом нагрузка на серверы существенно меньше — обработка запроса к кешу много проще, чем обработка запроса к базе данных SQL, обработка вики-разметки и формирование веб-страницы).

Кроме того, «обратный прокси» способен распределять запросы между несколькими серверами, балансируя нагрузку и/или обеспечивая отказоустойчивость, то есть фактически предоставляет функциональность, аналогичную кластеру.

Режим прозрачного прокси-сервера

В сочетании с некоторыми межсетевыми экранами и маршрутизаторами squid может работать в режиме прозрачного прокси (англ. transparent proxy). В этом режиме маршрутизатор вместо того, чтобы сразу пересылать HTTP-запросы пользователя HTTP-серверу в Интернете, перенаправляет их прокси-серверу, который может работать как на отдельном хосте, так и на самом маршрутизаторе. Прокси-сервер обрабатывает запрос (с возможной отдачей содержимого из кеша), это содержимое направляется к запросившему пользователю, для которого оно выглядит как «ответ» сервера, к которому адресовался запрос. Таким образом, пользователь может даже не знать, что все запросы и ответы прошли через прокси-сервер.

При таком подходе проксирования аутентификация не предусмотрена, так как прозрачность проксирования это и подразумевает.

Достоинства

Администратору прозрачного прокси-сервера не нужно настраивать каждую клиентскую машину для работы с прокси.

Недостатки

В режиме прозрачности не проксируются FTP- и HTTPS-запросы.

Для возможности проксирования в прозрачном режиме HTTPS-запросов необходимо собрать с поддержкой SslBump.

Установка

В терминале введите следующую команду для установки сервера Squid:

sudo apt-get install squid

Настройка

Squid настраивается редактированием директив, содержащихся в конфигурационном файле */etc/squid/squid.conf.* Следующие примеры иллюстрируют некоторые директивы, которые могут быть изменены для воздействия на поведение сервера Squid. Для более глубокой настройки Squid смотрите раздел Ссылки.

Прежде, чем редактировать конфигурационный файл, вам стоит сделать копию оригинального файла и защитить ее от перезаписи, чтобы у вас всегда оставались оригинальные настройки в качестве справочника и для повторного использования при необходимости.

Скопируйте файл */etc/squid/squid.conf* и защитите его от записи следующими командами в терминале:

sudo cp /etc/squid/squid.conf /etc/squid/squid.conf.original

sudo chmod a-w /etc/squid/squid.conf.original

1. Для настройки вашего сервера Squid на прослушивание порта 8888 вместо стандартного 3128, измените директиву http_port как показано здесь:

http_port 8888

2. Измените директиву visible_hostname для того, чтобы присвоить серверу Squid определенное имя хоста (hostname). Это имя необязательно должно быть именем хоста компьютера. В примере оно определено как weezie:

visible_hostname weezie

3. Используя контроль доступа Squid, вы можете настроить, чтобы использование интернет сервиса прокси было доступно только пользователям

с определенных IP адресов. Например, мы проиллюстрируем доступ пользователей только из подсети 192.168.42.0/24:

Добавьте следующее в конец секции ACL вашего файла */etc/squid/squid.conf:*

acl fortytwo_network src 192.168.42.0/24

Затем добавьте следующее в начало секции http_access вашего файла */etc/squid/squid.conf:*

http_access allow fortytwo_network

4. Используя великолепные возможности контроля доступа Squid, вы можете настроить возможность использования интернет сервиса прокси только в обычные рабочие часы. Например, мы покажем, как настроить доступ сотрудников, которые работают с 9:00 до 17:00 с понедельника по пятницу из подсети 10.1.42.0/24:

Добавьте следующее в конец секции ACL вашего файла */etc/squid/squid.conf:*

acl biz_network src 10.1.42.0/24

acl biz_hours time M T W T F 9:00-17:00

Затем добавьте следующее в начало секции http_access вашего файла */etc/squid/squid.conf:*

http_access allow biz_network biz_hours

После внесения изменений в файл /etc/squid/squid.conf сохраните его и перегрузите приложение сервера squid, чтобы изменения вступили в силу, следующей командой в терминале:

sudo /etc/init.d/squid restart

Почтовые серверы

Виды почтовых серверов

1. Тяжелые корпоративные продукты
2. Коммерческие СМБ-продукты (для среднего и малого бизнеса)

3. Некоммерческие Linux/Unix-продукты

Сегодня каждая компания использует электронную почту как одно из основных средств коммуникаций в бизнесе. Необходимым условием эффективного применения почты является наличие специальной программы - почтового сервера. Создавать корпоративные ящики с помощью публичных сервисов в Интернете уже давно считается своеобразным дурным тоном. К тому же, такой подход совершенно небезопасен и сопровождается трудностями в управлении.

Несмотря на то, что объем продаж почтовых серверов нельзя назвать огромным, этот рыночный сегмент весьма интересен. Во-первых, он многогранен - на нем можно увидеть около десятка серьезных игроков, среди которых присутствуют крупные транснациональные корпорации, небольшие стартапы и даже разработчики-энтузиасты. На фоне всеобщей тенденции к легализации программного обеспечения, рынок почтовых серверов стремительно развивается и наращивает собственные объемы.

Разработчики программного обеспечения часто действуют по принципу "чем больше, тем лучше", и их можно понять: покупателей того или иного программного продукта - великое множество, а требования к продукту у каждого различны. Создавая универсальный продукт, разработчики целятся на как можно более широкую аудиторию - и зачастую, сами того не осознавая, обеспечивают проблемы большей части этой широкой аудитории: разнообразные функциональные возможности продукта утяжеляют программное обеспечение и делают его настройку проблематичной для IT-служб компаний. Ситуация только усугубляется, когда речь заходит о компаниях, работающих в сфере среднего и малого бизнеса (СМБ). Перед такими организациями встает нетривиальная задача выбора решения, обладающего гибкими возможностями настройки, масштабируемостью и привлекательной ценой.

Вариантов выбора, на самом деле, множество: от решений с мировым именем до open-source продуктов. Вопрос в том, каковы ежедневные задачи небольших и средних компаний, а также насколько каждое из этих решений удовлетворяет потребностям среднестатистической компании.

Краткий обзор почтовых серверов

Рынок почтовых серверов весьма интересен и его игроков можно разделить на три основные группы:

1. тяжелые корпоративные продукты,

2. коммерческие продукты и

3. некоммерческие Linux/Unix-продукты.

Тяжелые корпоративные продукты

Поставщиками являются крупные ИТ-компании, предлагающие почтовые сервера, как часть платформы для построения корпоративной ИТ-инфраструктуры.

Таким образом, покупатель получает функционально богатый продукт, как правило, способный решать множество корпоративных задач не только из сферы применения почтовых программ. Кроме того, продукт из семейства "тяжелых" корпоративных систем включает в себя глубокую интеграцию с другими компонентами IT-инфраструктуры компании, как правило, приобретаемыми вместе с почтовой системой.

Решения, предназначенные для корпораций, часто находят применение и в среде с меньшим количеством рабочих станций.

Плюсами таких решений является богатый функционал, который не всегда ограничивается только почтовыми потребностями, и глубокая степень интеграции с остальными частями платформы.

Минусы - в наиболее высокой стоимости среди всех имеющихся на рынке продуктов, а богатство функционала требуется не всем заказчикам, и в некоторых случаях оно явно избыточно. Как следствие, растет сложность

поддержки и управления продуктом. Кроме того, благодаря более широкой распространенности, индустриальные системы привлекают внимание хакеров, что приводит к периодическим проблемам с безопасностью. Еще один минус - это привязка к платформе соответствующего производителя.

Примерами подобных продуктов могут служить системы с мировым именем Microsoft Exchange и IBM Lotus/Domino.

Microsoft Exchange Server позиционируется как платформа для организации корпоративной системы электронной почты, а также групповой работы. Формально существует версия для небольших предприятий, впрочем, включенная в состав пакета Microsoft Small Business Server. Главной особенностью данного продукта является тесная интеграция с инфраструктурой сетей на основе Windows и, как следствие, со службой каталогов Active Directory.

Излишним будет упоминание о том, что Exchange - система для сетей, полностью построенных под управлением Windows. Использование Exchange в смешанных сетях сопряжено с большим количеством сложностей.

IBM Lotus/Domino - еще один яркий представитель систем подобного класса. В отличие от Microsoft Exchange Server, IBM Lotus/Domino ориентирован скорее на полную организацию работы внутри компании, а функционал электронной почты в данном решении реализован скорее в качестве дополнения к корпоративной IT-платформе. Чаще всего этот программный продукт применяется в совокупности с приложениями, позволяющими автоматизировать различные процессы в компании.

Коммерческие СМБ-продукты (для среднего и малого бизнеса)

Отличие коммерческих почтовых серверов, в первую очередь, заключается в способах разработки. Коммерческий почтовый сервер разрабатывается, как правило, небольшой компанией, для которой является либо основным продуктом, либо одним из линейки смежных решений. Такой метод развития почтового решения наиболее выигрышен для компании,

поскольку позволяет сосредоточиться на разработке конкретного продукта. Целевой аудиторией таких продуктов являются компании, занятые в секторе среднего и малого бизнеса.

Kerio MailServer, решение, разработанное американской компанией Kerio Technologies, позиционируется как простое, безопасное и в то же время функциональное решение, подходящие для средних и малых компаний. Интерфейс этого решения прост в использовании, а сам продукт удобен в настройке.

MDaemon - продукт компании Alt-N Technologies - также позволяет развернуть почтовый сервер сравнительно быстро. Однако, позиция разработчиков отлична от указанного выше Kerio MailServer: Alt-N Technologies уделяют больше внимания функционалу, меньше - удобству и интерфейсу.

Представитель коммерческих серверов почты CommunigatePro не является почтовым сервером в чистом виде: это платформа для корпоративных коммуникаций, некий гибрид между крупным корпоративным решением и коммерческим почтовым сервером. В России, однако, средние и малые бизнесы используют это программное обеспечение исключительно как почтовый сервер.

Еще один продукт из этой категории - Merak Mail Server (IceWarp) - профессиональный почтовый сервер, поддерживающий все стандарты и протоколы. Продукт занимает небольшую долю рынка. Текущая версия: 9.4.0, есть локализация и поддержка на русском языке, поддерживает все распространенные ОС.

Плюсами таких почтовых решений для организаций являются доступная цена, простота в использовании и независимость от платформы, а к минусам можно отнести то, что функционал нельзя назвать исчерпывающим.

Некоммерческие Linux/Unix-продукты

Созданием таких продуктов занимаются разработчики-энтузиасты или исследователи, создающие программы для среды Unix/Linux. Четкой целевой аудитории нет; благодаря гибкости продукты могут использоваться практически везде.

Особенность почтовых решений на основе *nix в глобальной разнице в подходе к организации почтового сервера. Если в случае с коммерческими продуктами покупатель получает готовое универсальное решение, то, выбрав сервер на основе Linux/Unix, компания получает исключительно техническую организацию процесса передачи почты. Дело в том, что подобные почтовые системы представляют собой Mail Transfer Agent (MTA) - приложение, выполняющее обмен почтой между сервером и клиентом. В случае если необходим какой-то дополнительный функционал, его можно обеспечить установкой и настройкой дополнительных модулей. Таким образом, при выборе *nix решения пользователь получает конструктор, из которого нужно собрать почтовый сервер с необходимым функционалом.

В качестве примера программ данного класса стоит назвать Sendmail - старейшину рынка MTA. Это самое старое, популярное и известное MTA-приложение для Linux/Unix систем. Оригинальная версия Sendmail была написана еще в начале 80-х годов прошлого века Эриком Оллменом (Eric Allman), создателем приложения delivermail для проекта ARPANET (который впоследствии породил Интернет). Sendmail считается очень гибкой программой, которая обладает широким функционалом.

Sendmail входит в состав ряда дистрибутивов Linux/Unix систем. Продукт использует схему двойного лицензирования (по собственной лицензии SENDMAIL LICENSE), которая подразумевает бесплатное использование для собственных нужд. Имеется платный дистрибутив для требовательных корпоративных заказчиков. Текущая версия: Sendmail 8.14.

Система Postfix изначально создавалась как простой, безопасный и быстрый вариант приложения, выполняющего те же функции, что и Sendmail. Система Postfix была создана Вейтсом Венемой (Wietse Venema), разработчиком исследовательского центра IBM им. Томаса Уотсона (IBM Thomas J. Watson Research Center), в конце 90-х годов прошлого века.

Основным преимуществом Postfix является модульная архитектура: в решении применяются модули, каждый из которых отвечает за минимальный набор простейших функций. За счет этого разработчики программы смогли создать более безопасное решение, чем Sendmail.

Postfix поставляется по свободной лицензии IBM (IBM Public License 1.0).

Система Exim была создана в 1995 году сотрудником Университета Кэмбриджа (University of Cambridge) Филиппом Хейзелом (Philip Hazel). Ключевая особенность Exim заключается в логичной и прозрачной схеме обработки почты. Создатели продукта отказались от реализации ряда экзотических функций, что положительно отразилось на простоте решения. Кроме того, Exim считается быстрее и безопаснее Sendmail.

Exim входит в состав ряда дистрибутивов Linux/Unix систем. Поставляется по свободной лицензии GNU GPL.

Система Qmail является еще одним MTA-приложением, разработанным как альтернатива Sendmail. Автор Qmail, Дэниел Бернштейн (Daniel J. Bernstein), попытался создать абсолютно безопасный почтовый сервер. После создания первой версии продукта в 1996 году автор предложил награду в $500 первому человеку, который опубликует поддающуюся проверке уязвимость в последней версии программы. В прошлом году эта премия была до сих пор не востребована.

Qmail, как и Postfix, имеет модульную структуру, распространяется без лицензии, но давно не поддерживается автором - все современные варианты программы имеют дополнительные модули сторонних разработчиков.

Плюсами некоммерческих продуктов являются нулевая стоимость (как правило, эти продукты распространяются по лицензии GNU GPL) и богатые возможности по модификации кода и созданию нового функционала, а также гибкость.

Минусы - это то, что все перечисленные продукты являются не почтовыми серверами, а программами класса MTA (Mail Transfer Agent, агент передачи сообщений). Для создания на их основе полноценного почтового сервера требуется ряд дополнительных компонент. В частности, функционал для коллективной работы (groupware) в MTA абсолютно отсутствует. Некоммерческие продукты сложны в управлении и настройке, для работы с ними необходим специально обученный персонал. Как правило, отсутствуют локализованные версии и техническая поддержка на русском языке.

Таким образом, на рынке почтовых серверов существуют два уровня конкуренции. На первом уровне конкурируют подходы к реализации функционала почтовых серверов, а на втором - уже конкретные программные продукты.

Отметим, что выбор компанией того или иного подхода подвержен влиянию ряда сторонних факторов. Например, если в небольшой фирме серверы традиционно работают под Linux и поддерживаются администратором с соответствующей компетенцией, то вполне логично ожидать, что и почтовый сервер будет использовать Linux-подход.

Похожие тезисы применимы и к компаниям, развернувшим в своей ИТ-инфраструктуре платформу корпорации Microsoft.

Второй уровень конкуренции (между продуктами одного класса) от внешних факторов зависит в меньшей степени. Уместнее говорить о непосредственной конкуренции отдельно взятых продуктов. Использование почтовых серверов в организациях разных масштабов

В представленной ниже таблице приведены результаты исследования компании Security Lab по использованию почтовых серверов в организациях разных масштабов.

Количество рабочих станций в организации

	[менее 100]	[100-1000]	[более 1000]
[Microsoft Exchange]	35,48%	46,09%	35,48%
[Sendmail]	29,84%	32,03%	29,03%
[Kerio MailServer]	21,34%	13,62%	2,21%
[IBM Lotus]	8,87%	10,16%	19,35%
[Postfix]	19,36%	25,78%	19,35%
[Qmail]	7,26%	7,03%	9,68%
[Communigate Pro]	2,42%	4,69%	6,45%
[Exim]	10,48%	10,94%	6,45%
[MDaemon]	13,71%	9,37%	3,23%
[Merak Mail Server]	0,2%	0,6%	0,1%

Интересно, что организации среднего размера заметно чаще используют больше одного почтового сервера в своей деятельности. По всей видимости, эта тенденция объясняется менее строгой формализацией работы системных администраторов, которые пробуют работать сразу с несколькими системами.

В раскладке по отдельным продуктам отметим сервер Microsoft Exchange, популярный как в мелких, так и в крупных организациях. По всей видимости, это обстоятельство объясняется высоким уровнем пиратства в отношении данного продукта. Однако несомненно, что по мере легализации программного обеспечения, доля Microsoft Exchange будет неуклонно падать. Клиенты просто начнут считать деньги и переходить с "тяжелых" корпоративных решений на более доступные аналоги.

Linux/Unix-системы примерно повторяют ситуацию по рынку в целом - они успешно используются везде, и чаще всего - в организациях среднего размера.

Система IBM Lotus/Domino чаще применяется в крупных организациях, что также весьма закономерно. Не менее закономерен тот факт, что популярность Kerio MailServer и MDaemon падает по мере роста масштаба клиентов.

Независимо от размеров организации, первое место по популярности занимают некоммерческие Linux/Unix-серверы, за которыми следуют тяжелые корпоративные системы.

Протоколы POP3, SMTP, IMAP.

Электронная почта - это инструмент, без которого не могут обойтись специалисты, работающие в разных сферах. Очень неудобно постоянно проверять её через браузер. Поэтому люди, которым активно работают электронной почтой, используют почтовый клиент.

Принцип работы доступа к почтовому ящику.

Есть два основных протокола для доступа к почтовым сообщениям - POP3 и IMAP, а также протокол отправки электронной почты - SMTP. Чтобы сделать правильный выбор в пользу одного из протоколов, нужно понимать, как они работают, в чём заключаются преимущества и недостатки каждого.

Протокол POP3.

POP3 - это протокол приёма электронных сообщений. Для просмотра почты письма скачиваются на компьютер адресата. После этого они могут удаляться с сервера. Все действия с поступившей почтой происходят на компьютере пользователя.

У такого протокола есть бесспорные преимущества:

- к письмам, хранящимся на компьютере, всегда есть доступ (даже когда нет возможности выхода в интернет);
- вложение открывается вместе с сообщением, что очень удобно;
- пространство почтового ящика постоянно освобождается путём перегрузки писем на жёсткий диск компьютера.

Протокол POP3 работает на стандартном сетевом порту 110 TCL/IP, если не требуется шифрование. При необходимости шифрования (TLS или SSL) используется порт 995.

Недостаток протокола POP3 – это риск заражения локального компьютера вирусами, пришедшими вместе с почтой, неудобство одновременной работы с почтой на нескольких устройствах. Также может утомлять необходимость частого резервирования информации. Поскольку информация находится только на жёстком диске, его выход из строя грозит безвозвратной потерей всех сообщений.

POP3 это старый протокол, созданный в 1988 году. Его возможностей уже недостаточно для комфортного использования электронной почты. Ему на смену пришел протокол IMAP.

Протокол IMAP.

IMAP – это усовершенствованный протокол. Письма не скачиваются на локальный компьютер. Они остаются на сервере. Есть возможность доступа и обработки сообщений. Одновременно работать с почтой могут несколько пользователей.

Популярность протокола постоянно растёт. Это связано с тем, что увеличивается число точек доступа к интернету. Преимуществами IMAP считаются:

- есть возможность смены почтового клиента без необходимости смены электронного адреса;

- можно проверять почту не только со стационарного компьютера, но и с любого мобильного устройства, которое предоставляет выход в интернет;
- информация хранится на сервере, а компьютер не подвергается атаке вирусов.

Большим преимуществом IMAP-протокола является возможность автоматической фильтрации. Система проверяет заголовок сообщения и сравнивает его с заданными IMAP-фильтрами. Соответствие одному из заданных параметров позволяет поместить его в папку.

Для работы с IMAP используется 143 порт, а для IMAPS (шифрование поверх SSL) используется порт 993.

Недостатками протокола называют то, что отсутствие выхода в сеть не оставляет возможности даже посмотреть существующую почту. Если произойдёт сбой работы сервера, вы можете потерять часть информации.

IMAP или POP3.

Понимая особенности использования этих протоколов, можно выбирать вариант, который вам подойдёт. Есть несколько советов, которым лучше следовать.

Выбирайте IMAP, если:

- вы путешествуете по стране и имеете гарантированный доступ в интернет;
- хотите пользоваться разными устройствами без привязки к стационарному компьютеру;
- для совместной работы с электронной почтой;
- потеря информации грозит для вас многими бедами.

Отдавайте предпочтение POP3, если:

- не уверены в том, что всегда будет возможность выхода в сеть;

- просматриваете почту всегда со стационарного компьютера;
- хотите самостоятельно делать копии сообщений и резервировать их.

Протокол SMTP.

SMTP – это протокол доставки. В отличие от IMAP и POP3, он не может перемещать сообщение с сервера или управлять ящиком. Почтовый клиент отправляет SMTP-команды и получает ответы с сервера. Протокол отвечает за отправку электронных сообщений. У него есть две функции.

Проверка настроек и работа с компьютером, с которого отправляется сообщение (выдача разрешения).

Отправка сообщения по указанному адресу и подтверждение этого действия.

Отправка сообщения осуществляется непосредственно с сервера отправителя на сервер получателя. Связь между SMTP-серверами осуществляется с помощью команд, которые формируют сессию связи. Количество SMTP-операций не ограничивается. Для каждой из операций есть три главные команды:

- (MAILFROM) определить обратный адрес;
- RCPT TO) определить получателя;
- (DATA) отправить текст сообщения.

Порты SMTP:

- 25 ый порт, для соединения без шифрования;
- 465 ый порт SSL/TLS, он также называется SMTPS.

Некоторые интернет провайдеры блокируют 25ый порт, поэтому операторы, предоставляющие услуги электронной почты, обычно для SMTP открывают еще порты 250 и 2500.

Варианты работы с почтой.

Mail-клиенты - это программы, которые работают с почтой в интернете. Их существует множество. К самым популярным причисляют Apple Mail,

Outlook или Thunderbird. Они встроены в операционные системы. Есть клиенты, которые нужно скачивать на свой компьютер (например, TheBat). Все они имеют богатый функционал и удобными настройками интерфейса.

С почтой можно работать через web-интерфейс. Для входа в учётную запись используется логин и пароль. Предлагается удобный интерфейс почтового ящика. Вы можете выполнять с письмами разные действия: читать, сортировать, удалять, отвечать на пришедшие и создавать новые сообщения.

Работать с письмами можно не только на компьютере. С момента ввода на смартфоне индивидуальных данных аккаунта Google вы получаете доступ к использованию почтового сервиса Google Gmail.

Postfix

Postfix — агент передачи почты (MTA — mail transfer agent). Postfix является свободным программным обеспечением, создавался как альтернатива Sendmail.

Изначально Postfix был разработан Вейтсом Венемой в то время, когда он работал в Исследовательском центре имени Томаса Уотсона компании IBM. Первые версии программы стали доступны в середине 1999 года.

Postfix отличается продуманной модульной архитектурой, которая позволяет создать очень надёжную и быструю почтовую систему. Так, например, привилегии root требуются только для открытия порта (TCP 25 порт), а демоны, которые выполняют основную работу, могут работать непривилегированным пользователем в изолированном (chroot) окружении, что очень положительно сказывается на безопасности.

Архитектура Postfix выполнена в стиле UNIX — где простые программы выполняют минимальный набор функций, но выполняют их быстро и надежно. Во время простоя почтовой системы ненужные демоны могут прекращать свою работу, высвобождая тем самым память, а при необходимости снова запускаются master-демоном.

Также стоит отметить более простую и понятную конфигурацию по сравнению с Sendmail и меньшую ресурсоёмкость, особенно во время простоя почтовой системы.

Совместим с AIX, BSD, HP-UX, IRIX, GNU/Linux, Mac OS X, Solaris, Tru64 UNIX, фактически может быть собран на любой Unix-подобной операционной системе, поддерживающей POSIX и имеющей компилятор C. Является службой пересылки почты по умолчанию в ОС NetBSD.

Сервер электронной почты postfix позволяет организовать обмен электронной почтой с Интернет и внутри локальной сети. postfix рекомендуется к установке в любой конфигурации CentOS. Это объясняется тем, что в UNIX-подобных системах возможность отправлять почту с помощью вызова команды из командой оболочки практически обязательна. Некоторые программы (например, служба исполнения заданий по расписанию crond) пользуются этим для отправки сообщений пользователям.

Рекомендуется настраивать систему таким образом, чтобы доставкой всей электронной почты, проходящей через машину, занималась служба MTA (Mail Transport Agent), в нашем случае — postfix. Хотя многие почтовые программы способны отправлять сообщения на удалённый SMTP-сервер, имеет смысл поручить и эту задачу системному серверу электронной почты, чтобы не было необходимости следить за отсылкой отправленных писем.

Помимо postfix существуют и другие популярные реализации MTA, например, qmail, exim и даже ветеран Интернета Sendmail, хотя последний сейчас используется всё реже, поскольку неоправданно сложен в настройке. Однако по разным причинам, включая соображения безопасности, в любом дистрибутиве CentOS по умолчанию в качестве MTA предлагается postfix.

Работа в режиме SMTP-сервера

Обратите внимание, что по умолчанию после установки postfix не работает в режиме SMTP-сервера. Для того, чтобы принимать сообщения по

протоколу SMTP или ESMTP, как извне, так и локально, необходимо переключить службу postfix в режим работы server командой control postfix server. По умолчанию после установки служба postfix находится в режиме local, в котором postfix не принимает соединений из сети, ограничиваясь приёмом локальных соединений посредством сокетов семейства UNIX (UNIX-domain socket). Для большинства рабочих станций работа в режиме SMTP-сервера не требуется (и нежелательна с точки зрения безопасности), а в качестве SMTP-сервера должен использоваться какой-то внешний узел сети.

Пакеты postfix

Базовый RPM-пакет для установки сервера postfix в CentOS носит, как нетрудно догадаться, имя postfix. Ряд дополнений, расширяющих возможности postfix, выделены в отдельные пакеты, полный список которых можно получить командой:

— для CentOS:

yum search postfix-

Конфигурационные файлы

Файлы настройки postfix располагаются в каталоге /etc/postfix. Основные параметры определяются в файле main.cf; в частности, все параметры, о которых говорится далее в этом разделе, устанавливаются именно в этом файле, остальные случаи оговариваются специально.

Для простоты ориентации в настройке postfix в файле main.cf указываются только параметры, выставленные администратором системы, плюс некоторые из значений по умолчанию, которые администратору с большой вероятностью нужно будет изменить. Значения по умолчанию для всех остальных параметров перечислены в файле main.cf.default (этот файл не следует редактировать, он служит только для справок). Довольно полный список параметров с развёрнутыми комментариями и примерами приведён в

файле main.cf.dist, а наиболее полную информацию по всем возможным параметрам следует искать в руководстве postconf.

Обратите внимание, что если вы изменили конфигурацию при запущенной службе postfix, новые настройки нужно активизировать командой service postfix reload.

Доменная информация

Имя хоста и домена, которые считаются локальными при обработке email-адресов, необходимы для функционирования почтового сервера. Если эти имена для postfix должны отличаться от того, что выдаёт команда hostname, установите их с помощью параметров myhostname и mydomain.

Postfix на узлах с удалённым доступом к сети

Существует несколько проблем, возникающих при попытке отправки исходящей почты с машин, которые не являются полноценными узлами сети Интернет, например, в системах с модемным и другими непостоянными соединениями, не всегда возможно немедленно отправить сообщения удалённым адресатам по SMTP, и их приходится держать в очереди до тех пор, пока соединение не будет установлено. Для этого используется параметр defer_transports, например:

defer_transports = smtp

Доставка активизируется командой /usr/sbin/sendmail -q, которая в CentOS исполняется автоматически при установке PPP-соединения.

Как любой полноценный MTA, postfix позволяет доставлять сообщения, связываясь напрямую с сервером, обслуживающим получателя письма. Тем не менее, для dialup-машин непосредственная доставка сообщений нежелательна, поскольку время соединения ограничено. К тому же это излюбленная тактика распространителей спама, поэтому многие почтовые серверы сверяют IP-адрес отправителя с базой известных адресов

провайдерских пулов, после чего сообщения с таких адресов отвергаются. Поэтому целесообразно доверить доставку исходящей почты SMTP-серверу провайдера. Этим управляет параметр relayhost, например:

relayhost = [smtp.provider.net]

Postfix на клиентской машине локальной сети

Рабочие станции в локальной сети или машины в провайдерской сети, отделённой от Интернета с помощью межсетевого экрана/NAT, должны переправлять исходящую почту на почтовый сервер, обслуживающий данную сеть. Для этого также используется параметр relayhost, описанный выше. Если сервер задан IP-адресом, можно отключить использование DNS для ускорения работы:

disable_dns_lookups = yes

Для того чтобы в доменной части адреса отправителя фигурировал домен сети, а не имя конкретной машины в этой сети, установите параметр myorigin равным имени домена:

myorigin = $mydomain

Почтовый сервер для небольших доменов и сетей

Домены, для которых сервер получает почту, отличные от значения mydomain и не сконфигурированные как виртуальные домены postfix, нужно перечислить с помощью параметра mydestination, либо в дополнительном файле, на который ссылается этот параметр. Аналогичным образом параметр mynetworks описывает блоки IP-адресов, которые считаются внутренними и с которых разрешён приём исходящих сообщений. Не следует записывать в mynetworks блоки адресов, не принадлежащих сети, которую обслуживает сервер, поскольку этим могут воспользоваться распространители спама.

Для пользователей, желающих отправлять сообщения через данный сервер, когда они находятся вне пределов локальной сети, следует организовать SMTP-аутентификацию. В postfix для этой задачи следует использовать библиотеку SASL (начиная с версии 2.3 в postfix появилась

возможность использовать разные реализации SASL). На текущий момент доступно две реализации: Cyrus (пакет postfix-cyrus) или Dovecot (пакет postfix-dovecot). Cyrus, кроме того, можно использовать в postfix для аутентификации в качестве клиента на удалённых SMTP-серверах.

Для шифрования соединений в этой схеме используется протокол SSL/TLS, для включения поддержки которого в postfix необходимо установить пакет postfix-tls. Используя postfix-tls следует принимать во внимание, что это расширение включает в postfix значительный объём исходных текстов, в меньшей степени проверенного в плане безопасности. Настройка аутентификации посредством TLS/SASL описана в файлах TSL_README и SASL_README в документации postfix.

Преобразование глобальных почтовых адресов в локальные адреса назначения устанавливается с помощью таблиц типа virtual:

virtual_alias_maps = cdb:/etc/postfix/virtual

Пример содержимого /etc/postfix/virtual:

domain1.ru # Домен в стиле postfix (текст здесь игнорируется) name1@domain1.ru user1 name2@domain2.ru user2@otherbox @domain2.ru user3

После редактирования не забудьте активизировать изменения командой service postfix reload.

Алиасы и преобразования адресов

Имена локальных адресатов либо совпадают с именами учётных записей пользователей системы, либо подставляются из таблицы aliases:

alias_maps = cdb:/etc/postfix/aliases

alias_database = cdb:/etc/postfix/aliases

При установке postfix «с нуля» в этой таблице создаётся алиас на имя root: вся корреспонденция, предназначенная администратору и поступающая на другие системные адреса, будет доставляться на имя реального

пользователя, который осуществляет функции администратора. Изначально им становится первый зарегистрированный в системе реальный пользователь. Таблица алиасов отличается от остальных таблиц, используемых postfix; имена в поле слева (псевдонимы) отделяются от значений в поле справа (адреса доставки) двоеточиями. Адресаты в списке перечисляются через запятую. В качестве способа доставки можно использовать почтовые адреса, команды (обозначаются символом | в начале правой части; сообщение подаётся на стандартный поток ввода команды) и имена файлов:

```
John.Smith: john
chief: chief@bosscomputer
trio: stock, hausen, walkman
robot: | /usr/bin/robot --process-mail
filebox: /dir/file
```

Рабочий образ таблицы строится с помощью команды newaliases, а также при актуализации всех изменений посредством команды service postfix reload.

При отправке сообщения postfix формирует адрес отправителя автоматически из имени учётной записи пользователя и значения собственного домена (или значения myorigin, если этот параметр выставлен). Даже если почтовый клиент выставил заголовок From:, этот адрес попадает в служебные заголовки сообщения и может быть использован получателем, что не всегда желательно. Преобразование адресов отправителей к глобальным адресам можно задать в таблице типа canonical:

```
sender_canonical_maps = cdb:/etc/postfix/sender_canonical
```

Аналогичная таблица recipient_canonical и соответствующий параметр recipient_canonical_maps могут быть использованы для преобразования адресов назначения. Для актуализации изменений таблиц используйте команду service postfix reload.

Защита от нежелательной корреспонденции

Противодействие спаму (массовым рассылкам непрошеной корреспонденции) — отдельная большая тема, которую невозможно полностью раскрыть в этом руководстве; здесь даны лишь несколько практических советов применительно к конфигурации postfix. По умолчанию сервер сконфигурирован так, что отвергает попытки переслать сообщения извне на другие удалённые серверы. Со спамом, адресованным локальным получателям, дело обстоит сложнее.

В postfix предусмотрены возможности передачи почтовых сообщений на проверку на спам как внутренним фильтрам, так и внешним программам. Подробности о фильтрации почты см. в FILTER_README в документации postfix. Режим фильтрации почты включается в CentOS командой control postfix filter.

Один из возможных способов — опираться на данные служб, организованных по принципу «чёрного списка» IP-адресов (MAPS RBL и ей подобные). Чтобы задействовать эти сервисы, предварительно ознакомившись с условиями их использования, занесите имена доменов, работающих по принципу RBL, в конфигурацию:

smtpd_client_restrictions = permit_mynetworks, reject_maps_rbl

maps_rbl_domains = relays.ordb.org, blackholes.mail-abuse.org

В некоторых случаях требуется адресная работа с отдельными нарушителями почтового этикета. Адресная работа заключается в блокировании SMTP-соединений с их адресов, сетей либо доменов. Для этого предусмотрены таблицы типа access:

smtpd_client_restrictions = permit_mynetworks, cdb:/etc/postfix/access

Пример таблицы:

1.2.3.4 550 No more canned meat, please

1.2.5 REJECT

goodguy.generallybad.com OK

.generallybad.com REJECT

Как и с другими таблицами, закончив редактирование, приведите новые настройки в действие командой service postfix reload.

Прочие настройки

По умолчанию размер файла почтового ящика при локальной доставке ограничен 51200000 байтами. Это ограничение можно изменить с помощью параметра mailbox_size_limit. Установка параметра в 0 снимает ограничение.

Использование Postfix

После того, как postfix настроен и запущен как служба с предсказуемым именем postfix, в настройках почтовых клиентов можно указывать имя или адрес машины (например, localhost) в качестве SMTP-сервера. Для получения почты с удалённых серверов можно использовать программу fetchmail, которая работает в связке с postfix, опрашивая внешние почтовые ящики пользователей по протоколам POP3 или IMAP и передавая полученные сообщения системному MTA для локальной доставки. Подробнее о работе с fetchmail см. в документации соответствующего пакета. Файлы журналов postfix находятся в каталоге /var/log/mail.

ZIMBRA

Системы групповой работы по праву занимают одно центральных мест среди must have приложений в организациях разных типов. Решения класса "все в одном" дают возможность пользователям обмениваться сообщениями, документами, планировать задачи и многое другое. Продукт Zimbra

Collaboration Suite (ZCS), распространяемый под свободной лицензией, успешно конкурирует со многими проприетарными аналогами.

Возможности ZCS

Проект калифорнийской компании Zimbra Inc. громко заявившей о себе в 2007 году, сразу привлек к себе внимание. И хотя первые версии по функциональности не дотягивали до большинства имеющихся тогда OpenSource решений, заложенный в нем потенциал был огромен, а поэтому интересен многим админам. В результате в сентябре 2007 года компания была выкуплена Yahoo!, а в начале 2010 перешла к VMware.

Сегодня в Zimbra входит стандартный набор приложений, необходимых для любой системы коллективной работы. В первую очередь это почтовый сервер, позволяющий пользователям работать с почтой с помощью клиентских программ поддерживающих протоколы POP/POPS и IMAP/IMAPS или через веб-интерфейс. Обеспечивается фильтрация спама и антивирусная проверка почты при помощи ClamAV. К слову, простота развертывания почтового сервиса была оценена еще в первых релизах продукта, поэтому многие админы вместо установки разношерстной связки сервисов и обеспечения их совместной работы, сразу ставили Zimbra.

Пользователь может настроить сбор почты с других ящиков, сообщения при этом будут копироваться на сервер Zimbra, поддерживается работа с несколькими доменами.

Zimbra включает в себя также сервис мгновенного обмена сообщениями (Jabber), календарь с возможностью планирования событий, систему управления контактами, систему обмена документами с полноценным WYSIWYG редактором Zimbra Document.

Последний поддерживает форматы RTF, HTML, работу с буфером обмена, и что особенно важно редактор понимает кириллические шрифты. Предусмотрена возможность вставки таблиц и изображений, поэтому Zimbra Document может удовлетворить потребности большинства пользователей, без необходимости дополнительной установки офисного пакета. Проблем с набором или чтением документов у пользователя точно не будет. Любой документ можно расшарить, чтобы к нему могли получить доступ другие участники (для доступа нужно знать URL). Адрес документа можно отправить по e-mail или публиковать в виде RSS/Atom.

Реализован полноценный поиск по документам, вложениям и текстам e-mail. Для удобства сообщениям можно присваивать тэги, группируя их, например по назначению. Также Zimbra обеспечивает двустороннюю синхронизацию со многими мобильными устройствами (Windows Mobile, iPhone, Nokia E и так далее), среди веб-интерфейсов доступен и упрощенный вариант, предназначенный для доступа с мобильных устройств через медленные каналы.

Учетные записи пользователей можно хранить локально, а также в любом LDAP сервере, в том числе и домене ActiveDirectory.

Разработчики предоставили специальное API, позволяющее создавать дополнительные плагины, называемые zimlets существенно расширяющие возможности Zimbra. Используя зимлеты достаточно просто интегрировать в ZCS продукты и сервисы, разработанные третьими лицами или новые функции, создав единую среду, обладающую нужной функциональностью. И кстати именно благодаря зимлетам Zimbra получил такую популярность и функциональность. В стандартной поставке сервера идет несколько десятков зимлетов, по умолчанию устанавливается лишь малая часть из них.

Как водится, в таких случаях, приложение использует клиент-серверную архитектуру. Серверная часть Zimbra Server написана на Java, является POP3/IMAP сервером, и базируется на нескольких OpenSource

проектах, среди которых nginx, Apache Lucene, OpenLDAP, MySQL, Postfix, POP3/IMAP4 прокси Perdition, ClamAV, DSPAM и некоторые другие.

Веб-клиент Zimbra Web Client — обеспечивает интерактивный, удобный и что не менее важно локализованный веб-интерфейс для получения доступа к данным пользователя. Построен с применением технологии AJAX, что упрощает взаимодействие пользователя и выполнение ряда операций. Например, достаточно навести курсор на дату в календаре, как будет высвечены все события дня, если навести мышку на адрес в сообщении или контакте, сразу будет показана схема проезда, щелчок на телефонном номере запустит Skype или Ekiga, позволяя сразу поговорить с этим человеком и так далее.

И, наконец, клиент совместной работы Zimbra Desktop, который обеспечивает подключение к серверу и синхронизацию данных (почта, контакты, календарь и так далее), может использоваться в качестве почтового клиента для любого IMAP/POP3 почтового сервиса.

ZCS предлагается в трех версиях: Open Source Edition, Network Edition (Starter, Standard и Professional) и Zimbra Appliance (Basic, Standard). Первая распространяется свободно под OpenSource-подобной ZPL лицензией (Zimbra Public License). Поддерживает неограниченное количество пользователей, и хотя имеет некоторые ограничения по сравнению с платными версиями, но они никоим образом не мешают использованию Zimbra в организациях малого и среднего размера.

Несколько сокращены инструменты администратора, отсутствует возможность синхронизации с внешними устройствами MS Outlook, отсутствует встроенный механизм резервного копирования и восстановления, невозможность работы в кластере и другие. Хотя отчаиваться не стоит, некоторые "пропущенные" в OpenSource версии вопросы давно уже решены мощным комюнити проекта. Так, например в Wiki можно найти несколько

вариантов скриптов, предназначенных для резервирования текущей установки Zimbra.

С Open Source Edition мы и будем знакомиться далее.

Установка ZCS OpenSource Edition

Серверная часть доступна для x32 и x64 битных версий Linux (Red Hat Enterprise, Fedora, Ubuntu, Debian, Mandriva, SUSE Linux) и Mac OS X. Кроме этого доступны исходные тексты и последние патчи.

Перед установкой следует правильно настроить разрешение имен на DNS сервере, мастер установки будет проверять A и MX и в случае неудачи завершит работу с ошибкой.

Сам процесс установки очень прост и если все требования выполнены, займет не более 10 минут времени.

Качаем архив, под требуемую платформу.

$ wget –c http://files2.zimbra.com/downloads/6.0.8_GA/zcs-6.0.8_GA_2661.UBUNTU8_64.20100820044710.tgz

Распаковываем и запускаем установочный скрипт.

$ tar xzvf zcs-6.0.8_GA_2661.UBUNTU8_64.20100820044710.tgz

$ cd zcs-6.0.8_GA_2661.UBUNTU8_64.20100820044710

Затем важная часть проверка записи имени узла в /etc/hosts и зависимостей. Сама программа установки ничего из репозитария не ставит, это нужно сделать админу. Если какого-то пакета не будет найдено, напротив его имени выводится MISSING, а скрипт по окончании анализа заканчивает свою работу.

До устанавливаем что не хватает, и повторяем.

$ sudo apt-get install libpcre3 libgmp3c2 libgmp3-dev sysstat libexpat1 wget

Если этот шаг пройден нормально, скрипт проверяет наличие пакетов в архиве и запрашивает разрешение на установку каждого (всего их 11). По умолчанию мастер предлагает установить все компоненты, за исключением

zimbra-memcached и zimbra-proxy (прокси POP3, IMAP и HTTP). Причем, если выбран zimbra-proxy, то memcached будет установлен автоматически.

Далее выдается запрос на разрешение модификации системы.

Соглашаемся, и начинается собственно процесс установки пакетов и настройки параметров.

Теперь скрипт запросит DNS сервер на предмет имени узла Zimbra, если ответ (A и MX) не будет совпадать с записью в /etc/hosts, последует вопрос о смене.

Далее проверка конфликта портов и выводится меню установки, в котором можно откорректировать любое значение.

Особое внимание следует обратить на пункты, отмеченные несколькими звездочками, это означает не настроенный параметр. Как минимум один такой есть - "Admin Password", означающий на отсутствие пароля администратора. Для изменения нужного пункта нажимаем соответствующую ему цифру. Так чтобы установить пароль, выбираем 3, появляется еще одно меню, ищем "Admin Password" и нажимаем цифру (она опять будет подсвечена ***), после чего вводим дважды пароль.

Чтобы перейти в старшее меню, нажимаем "r", клавишей "s" или "a" сохраняем настройки (скрипт выдаст имя файла) и для выхода из меню используем "q". Вот собственно и весь процесс установки. Еще некоторое время будут настраиваться сервисы, вся информация по установке будет сохранена в /opt/zimbra/log.

К слову убрать Zimbra так же просто, как и установить. Вначале вводим команду:

$ ./install.sh --uninstall

Затем обязательно удаляем каталог /opt/zimbra, в нем даже после удаления сохраняются все настройки.

Веб-интерфейсы Zimbra

После установки серверной части будет доступно два интерфейса. Обычные пользователи для работы с почтой, документами и календарем должны набирать в браузере URL сервера без указания номера порта. При входе можно будет выбрать один из трех вариантов, который подходит для разных условий - стандартный (HTML), расширенный (AJAX) и мобильный телефон. В стандартном варианте отсутствует все, что связано с AJAX, то есть встроенная работа с документами, всплывающие подсказки и прочие удобства. Так если выбрать созданный документ находящийся в Портфеле его будет предложено сохранить на локальном диске или открыть во внешней программе, в расширенном сразу откроется редактор. В стандартном интерфейсе доступны календарь, задачи, ежедневник, работа с почтой, адресная книга и портфель.

Разобраться, как работать с пользовательским интерфейсом очень просто.

Особенно учитывая, что доступна онлайн справка на русском языке, в которой достаточно подробно описаны все тонкости работы. Собственно, из-за простоты и любят Zimbra админы, всего пара команд и толковый почтовый сервер развернут.

Управление Zimbra из консоли

Кроме веб-интерфейса, настройками Zimbra можно управлять и с

помощью большого количества команд, выполнять которые необходимо под учетной записью zimbra. Список всех команд доступен в документации на сайте проекта "Zimbra CLI Commands". Например, системная команда "service zimbra status" по сути обращается к утилите zmcontrol.

zmcontrol (status | stop | start | maintenance | startup)

Чтобы обратиться к отдельному сервису просто указываем его имя. Поумолчанию идет опрос на локальной системе, но добавив параметр -H можно указать удаленный сервер.

Утилита zmaccts позволяет получить данные об аккаунтах, zmprov модифицировать данные LDAP, с его помощью можно например настроить алиасы для домена, создать учетные данные и так далее. Плюс для каждого сервиса можно найти специфическую команду.

Веб-интерфейс администратора "находится" на 7071 порту. Набираем в браузере ссылку https://server.com:7071, для входа используем логин admin и пароль указанный во время установки.

После входа попадаем во вкладку "Состояние сервера", где показан статус всех установленных сервисов. Основное управление находится в пяти меню:

Адреса – управление аккаунтами пользователей, создание алиасов, листами распространения и ресурсами, просмотра почты пользователей и смена пароля;

Конфигурация – глобальные настройки доступных возможностей сервера, тем, глобальные настройки (квоты, длина и время жизни пароля) и класс обслуживания, включение и установка zemlets и расширений администрирования, настройки домена и серверного пула;

Мониторинг – вывод статуса сервисов и статистики сервера за разный период времени (количество сообщений, вирусная и спам активность, при необходимости легко можно добавить график загруженности CPU, отдельного сервиса и так далее);

Сервис – управление почтовыми очередями и сертификатами, обновление ПО сервера;

Профили поиска - вкладка содержит несколько шаблонов позволяющих быстро выполнить поиск с определенными условиями (неактивные, блокированные, административные учетные записи и так далее).

При таком большом количестве опций, система на самом деле очень проста в администрировании, все настройки находятся на своих местах и производятся так как ожидаешь, без каких-либо "сюрпризов". Поэтому разобраться с управлением человеку, понимающему чего он хочет в итоге получить, очень легко.

После установки в системе присутствует несколько учетных записей - администратора, wiki и обучения спам-фильтра. Чтобы добавить нового пользователя достаточно выполнить "Учетные записи – Создать – Учетная запись" и заполнить предложенные поля. Предусмотрен импорт с файла в формате CSV, для этого достаточно выбрать Другие действия - Групповая подготовка. Файл должен содержать строки - логин, имя и пароль, разделенные запятой. Например:

user@domain.com,name,password

Если пароль не указан, пользователю будет задан случайный пароль, который он должен будет сменить при первом входе. Аналогично просто создаются псевдонимы, списки рассылок и ресурсы. В интернет можно найти несколько готовых решений позволяющих произвести миграцию почтовых ящиков, календарей и т.п. в Zimbra из других систем. Чтобы их найти просто вбей в гугле, что-то вроде "to Zimbra migration". Основную информацию по переносу пользовательских аккаунтов можно получить по ссылкам на Wiki проекта.

Следует отметить удобство при администрировании большого количества серверов и доменов. Так изначально для всех серверов действуют установки, указанные в двух подпунктах Конфигурация – Глобальные настройки и Класс обслуживания. Последние представляют своего рода политики для аккаунтов пользователей, серверов и доменов. По умолчанию присутствует один класс default, который и наследует сервер или группы серверов Zimbra.

В том случае если для каждого сервера необходимо использовать разные параметры, следует создать новый класс сервера, в котором их и указать.

Настройка класса содержит 7 вкладок, в которых настраиваются возможности сервера, параметры клиента (HTML и AJAX), поиск, отправка и получение почты, отправка получение приглашений календаря, доступные темы и модули Zimlet, лимиты ресурсов и многие другие. Следует внимательно пройтись по всем вкладкам и выставить параметры в соответствии с требованиями.

Например, в классе default отключена функция Мессенджер отвечающая за работу системы мгновенного обмена сообщения, если такой сервис нужен, его следует включить.

Язык интерфейс выбирается автоматически по установкам браузера, но в большинстве случаев его лучше сразу зафиксировать во вкладке Настройки - Язык.

Здесь же в подразделе "Параметры еженедельника" указываем часовой пояс и первый день недели в календаре (по умолчанию воскресенье — это неудобно). По окончанию не забываем нажать кнопку Сохранить.

При создании нового домена или редактирования настроек имеющегося, просто выбираем нужный класс, и все настройки которого будут наследованы. Некоторые из них можно затем переопределить в индивидуальном порядке. Аналогично настройки переопределяются для каждого отдельного сервера. Например, выбрав вкладку Службы можно отключить ненужные сервисы.

В Глобальных настройках указывается максимальный размер файла в Портфеле, список запрещенных расширений файлов, которые будут блокироваться, настройки MTA, POP, IMAP, взаимодействие с Exchange и другие.

Таким образом, параметры будут применены в таком порядке – Глобальные настройки – Класс сервера и персональные настройки. В настройках сервера есть кнопки, позволяющие сбросить параметр до глобального значения.

По умолчанию вместе с сервером устанавливается только 6 zimlets. Все остальные находятся в нескольких каталогах /opt/zimbra/zimlets*. Чтобы добавить любой из доступных, следует выбрать ссылку Zimlets, нажать Инсталляция и указать на выбранный zip архив. При следующей регистрации пользователя новый zimlets (за исключением некоторых) появится в списке. В дальнейшем пользователь самостоятельно настраивает параметры зимлета при помощи контекстного меню.

Заключение

Итак, практически не прилагая особых усилий, всего лишь введя несколько команд, мы получили полнофункциональный сервер, обеспечивающий все необходимые инструменты для организации коллективной настройки. В дальнейшем управлять им может любой пользователь, не обладающий продвинутыми знаниями по администрированию *nix систем и сетевых сервисов. Используя дополнительные модули расширения можно еще больше нарастить его возможности.

PuTTY и WinSCP

Putty

PuTTY — свободно распространяемый клиент для различных протоколов удалённого доступа, включая SSH, Telnet, rlogin. Также имеется возможность работы через последовательный порт.

Описание

PuTTY позволяет подключиться и управлять удаленным узлом (например, сервером). В PuTTY реализована только клиентская сторона соединения — сторона отображения, в то время как сама работа выполняется на стороне сервера.

Изначально разрабатывался для Microsoft Windows, однако позднее портирован на Unix. В разработке находятся порты для Mac OS и macOS. Сторонние разработчики выпустили неофициальные порты на другие платформы: мобильные телефоны под управлением Symbian OS, коммуникаторы с Windows Mobile, а также устройства с iOS и Android.

PuTTY входит в репозитории практически всех популярных дистрибутивов Linux (в т.ч. Ubuntu, Debian, ALT Linux).

Исходный код PuTTY полностью разработан на C. PuTTY не зависит от DLL, других приложений, пакетов обновлений ОС. Пакет состоит только из исполняемых файлов, которые могут быть установлены в любом месте. PuTTY и большинство утилит запускаются только в одном потоке ОС. Программа является свободно распространяемым приложением с открытым исходным кодом и выпускается под Open Source лицензией MIT.

Некоторые возможности программы:

- Сохранения списка и параметров подключений для повторного использования.
- Работа с ключами и версиями протокола SSH.
- Клиенты SCP и SFTP (соответственно программы pscp и psftp).
- Возможность перенаправления портов через SSH, включая передачу X11.
- Поддержка большей части управляющих последовательностей xterm, VT-102, а также значительная эмуляция терминала ECMA-48.
- Поддержка IPv6.
- Поддержка 3DES, AES, Arcfour, Blowfish, DES.

- Поддержка аутентификации с открытым ключом, в том числе и без ввода пароля.
- Поддержка работы через последовательный порт (начиная с версии 0.59).
- Возможность работы через прокси-сервер.

Применение PuTTY:

- удаленное администрирование Linux.
- подключение к виртуальным серверам по протоколу SSH.
- настройка сетевых маршрутизаторов через последовательный порт.
- соединение с удаленными Telnet-терминалами и пр.

Состав

В состав PuTTY входят утилиты:

- PuTTY — сам клиент для Telnet и SSH (putty.exe)
- PSCP — клиент для SCP (удаленное копирование файлов по шифрованному протоколу scp с управлением из командной строки) (pscp.exe)
- PSFTP — клиент SFTP (psftp.exe)
- PuTTYtel — клиент для Telnet
- Plink — интерфейс командной строки к PuTTY (plink.exe)
- Pageant — агент SSH-аутентификации для PuTTY, PSCP и Plink (pageant.exe)
- PuTTYgen — утилита для генерации RSA- и DSA-ключей (puttygen.exe)
- pterm — автономный эмулятор терминала (только для Unix-версии)
- Пользователь может установить как полный пакет, так и каждый модуль по отдельности.

КАК ПОЛЬЗОВАТЬСЯ PUTTY

1. ИНТЕРФЕЙС ПРОГРАММЫ

Сразу же после запуска программы из меню пуск вы увидите графический интерфейс ее настройки. После подключения вы будете видеть только терминал, но настроить программу можно через удобный интерфейс.

Рассмотрим за что отвечают те или иные вкладки программы, чтобы вы ориентировались что и где искать. У нас есть четыре вкладки:

Session - отвечает за подключение удаленному серверу, тут мы вводим параметры подключения, порт, адрес, а также можем сохранить все настройки putty, чтобы не настраивать каждый раз заново.

Terminal - позволяет включать или отключать возможности терминала;

Window - настройка внешнего вида окна, цвет, шрифт, кодировка;

Connection - настройка параметров подключения, алгоритма шифрования, сжатия, ключей аутентификации, X11 и других параметров.

Каждая вкладка имеет несколько подразделов.

2. ПОДКЛЮЧЕНИЕ К УДАЛЕННОМУ КОМПЬЮТЕРУ PUTTY

Чтобы подключиться к удаленному компьютеру по SSH перейдите на вкладку "Session", здесь, в поле "Host Name" необходимо прописать ip адрес или имя хоста, компьютера, к которому вы хотите подключиться, в поле порт - нужно указать порт, на котором запущен SSH сервер, по умолчанию используется порт 22:

Далее, нажмите кнопку "Open". После этого появится запрос на добавление ключа сервера в список доверенных ключей, нажмите "Да":

Затем вам будет нужно ввести логин пользователя и пароль. Важно заметить, что скопировать логин или пароль у вас не получится, необходимо только вводить вручную.

Теперь авторизация прошла успешно, и вы можете выполнять нужные действия на сервере.

3. СОХРАНЕНИЕ СЕССИИ PUTTY

Чтобы не вводить каждый раз ip и порт можно сохранить эти данные в виде сессии, для этого пропишите новое имя в поле "Saved Sessions", а затем нажмите кнопку "Save":

Теперь вы сможете загрузить сохраненную сессию, нажав кнопку "Load".

После того как будет завершена настройка putty и все параметры будут выставлены правильно вы можете сохранить настройки и не вводить их несколько раз.

4. ИМЯ ПОЛЬЗОВАТЕЛЯ ПО УМОЛЧАНИЮ

Вы можете не вводить имя пользователя каждый раз, для этого перейдите на влкадку "Connection", затем "Data" и в поле "Auto-login Username" пропишите имя пользователя, например, root:

Теперь подключение putty будет выполняться от имени этого пользователя.

5. АВТОРИЗАЦИЯ ПО КЛЮЧУ SSH В PUTTY

Чтобы не вводить каждый раз пароль можно настроить авторизацию по ключу. В Linux такая возможность используется очень широко потому что это удобно. Первым делом необходимо создать ключ. Для этого запустите утилиту PuTTYgen и установите переключатель в положение "SSH-2 RSA" нажмите "Generate".

Обязательно ключ должен быть SSH-2 RSA, если в главном окне нет, выберите в меню "Key". Подвигайте мышкой, чтобы создать достаточное количество энтропии:

Ключ готов, затем, с помощью кнопок "Save Public Key" и "Save Private Key" сохраните оба ключа.

Далее, откройте PuTTY, перейдите на вкладку "Connection", затем "SSH", затем "Auth":

Здесь необходимо нажать кнопку "Browse" и добавить недавно сохраненный приватный ключ:

Далее, возвращаемся на вкладку "Session", выбираем наше сохранение и нажимаем "Save" чтобы сохранить настройки. Осталось только отправить наш открытый ключ на сервер. Для этого авторизуйтесь на нем с помощью пароля и открытый ключ вставьте ключ в конец файла /root/.ssh/authorized_keys.

Ключ можно брать прямо из окна PuTTYgen "Public key for pasting" или из файла открытого ключа:

Все, теперь можно выходить и авторизоваться снова. На этот раз подключение по ssh putty будет выполняться с помощью нашего ключа. Не забывайте сохранять настройки сессии, чтобы не выбирать ключ каждый раз.

WinSCP

WinSCP – это, пожалуй, одна из наиболее популярных программ для обмена файлами между узлами с операционными системами Linux, Windows или MacOS по протоколам FTP, SFTP, FTPS, SCP, WebDAV и Amazon S3. Позволяет выполнять типовые операции с файлами и папками, такие как загрузка с удаленного узла и выгрузка на удаленный узел, переименование, перенос, удаление и создание файлов и папок в локальной или удаленной файловой системе. Он также позволяет просматривать и изменять свойства файлов и папок, а также создавать символьные ссылки и ярлыки. Программа имеет многоязычную поддержку и предоставляет пользователю возможность выбора интерфейса в стиле проводника или файлового менеджера Total Commander. Кроме того, в пакет WinSCP включена специальная утилита командной строки winscp.com, позволяющая автоматизировать процессы обмена файлами и папками с помощью заранее подготовленных сценариев.

Кроме того, WinSCP предоставляет пользователю массу дополнительных возможностей:

- Возможность импорта настроек соединений из установленной в системе программы PuTTY. - Интеграция с Pageant (PuTTY Agent) с поддержкой авторизации по открытым ключам.

- Интеграция с операционной системой Windows (поддержка Drag&Drop, ярлыков, поддержка схем URL).

- Поддержка работы с любыми версиями протокола SSH (Secure Shell)

- Поддержка различных типов авторизации, как по паролю, так и с использованием ключей.

- Встроенный текстовый редактор.

- Возможность сохранять настройки соединений.

- Возможность работы с использованием файла конфигурации вместо хранения настроек в реестре, что позволяет использовать ее в переносимом варианте (Portable WinSCP).

- Плагин для поддержки протокола SFTP в программе FAR Manager.

- Возможность синхронизации каталогов локального и удаленного узлов.

Программа WinSCP абсолютно бесплатна и распространяется с открытым исходным кодом.

Установка и настройка WinSCP

Скачать актуальную версию программы можно на странице загрузки WinSCP

Установка программы выполняется стандартным образом с рекомендуемыми параметрами или с возможностью выбора параметров. Выбираемые параметры можно изменить в любой момент времени после установки. Настройки WinSCP выполняются через меню Параметры - Настройки. Множество различных параметров настройки внешнего вида и поведения программы позволяют легко адаптировать ее под предпочтения конкретного пользователя.

Настройки программы WinSCP.

Настройки по умолчанию, как правило, соответствуют предпочтениям обычного пользователя Windows, но при необходимости, можно, например, изменить поведение программы при обрыве связи, отклонении сервером подключения по протоколу SFTP и т.п.

Подключение к серверам

Программа WinSCP позволяет создавать и, по желанию пользователя – сохранять, профили подключений к различным серверам, поддерживающим

соответствующие прикладные протоколы FTP, FTPS, SFTP, SCP, WebDAV, Amazon S3.

Для создания нового подключения используется кнопка Новое соединение - Новое подключение или комбинация клавиш CTRL+N. Далее необходимо выбрать параметры соединения – протокол передачи данных, наличие и свойства шифрования, имя сервера (IP-адрес) и номер порта, а также - учетную запись пользователя, используемую при подключении к указанному серверу.

Настройки подключений в программе WinSCP.

После завершения настройки подключения, можно выполнить его сохранение нажав кнопку Сохранить Сохраненное соединение можно использовать в качестве соединения по умолчанию, которое будет выбираться при запуске программы WinSCP.

В нижней части окна со списком соединений размещены кнопки Инструменты и Действия предоставляющие пользователю дополнительные возможности по сохранению и восстановлению конфигурации подключений, взаимодействию с программой PuTTY, стиранию следов работы программы и вызову ее настроек.

Передача данных.

Двухпанельный интерфейс по образу Total Commander, как правило, удобнее интерфейса в стиле Проводника Windows. В левом окне отображается локальная файловая система, в правом – файловая система сервера, к которому выполнено подключение.

Интерфейс в стиле Total Commander программы WinSCP.

Настройки WinSCP позволяют поменять панели местами - Настройки - Внешний вид - Коммандер - Панели - включить Поменять панели местами (локальная справа, сервер слева)

Для передачи файлов на сервер производится их выделение в левом окне и отправка нажатием кнопки Отправить. Для приема файлов от удаленного сервера производится их выделение в правом окне и получение нажатием кнопки Получить. Выделение файлов и каталогов выполняется стандартным для ОС семейства Windows способом. Перед началом обмена данными с сервером, отображается диалоговое окно, позволяющее изменить некоторые параметры передачи и выполнить ее настройки:

Настройка параметров передачи данных в программе WinSCP.

В процессе передачи данных отображается статистическая информация:

Отображение процесса передачи данных в программе WinSCP.

Важной особенностью Winscp является возможность синхронизации локальных и удаленных данных. Режим синхронизации включается через меню Команды - Синхронизация .

Режим синхронизации в программе WinSCP.

В настройках синхронизации можно выбрать 3 направления:

Компьютер - изменения в удаленном каталоге применяются по отношению к локальному.

Сервер - изменения в локальном каталоге применяются по отношению к удаленному.

В обе стороны - изменения выполняются как в удаленном, так и в локальном каталогах.

Кроме направления синхронизации, можно выбрать также и ее режим:

Синхронизация файлов - основной режим синхронизации в WinSCP. Реализуется алгоритм:

- Файлы, более новые в каталоге источника передаются в каталог приемника.

- Файлы, присутствующие в каталоге источника, но отсутствующие в каталоге приемника передаются в каталог приемника, если не включен режим Лишь имеющиеся файлы в настройках Параметры синхронизации.

- Файлы, существующие в каталоге приемника, но отсутствующие в каталоге источника могут быть удалены, если включен режим Удалять файлы в настройках Параметры синхронизации.

- При направлении синхронизации В обе стороны файлы, отсутствующие в противоположном каталоге считаются новыми и передаются, если не включен режим Лишь имеющиеся файлы. Удаление файлов при использовании данного направления синхронизации не выполняется.

Зеркальные файлы - различающиеся файлы (более новые и более старые) в каталоге источника передаются в каталог приемника.

Синхр. штампов времени - режим активен, если в удаленной и локальной системе существует возможность определения времени модификации файлов. Обычно, это подключение по протоколу SFTP. Никакие файлы не изменяются и не удаляются. Если один и тот же файл существует в каталоге источника и в каталоге приемника, можно изменить его отметку времени на время модификации либо источника, либо приемника по выбору пользователя. При направлении синхронизации В обе стороны обновляется отметка времени более старых файлов.

Автоматизация обмена данными с применением WinSCP.

Программа WinSCP имеет поддержку командной строки и развитую систему создания и выполнения сценариев, что позволяет легко реализовать автоматизацию повторяющихся процессов приема и передачи данных.

Для работы со сценариями можно использовать утилиту командной строки winscp.com, либо запустить программу с параметром /console:

"C:\Program Files (x86)\WinSCP\WinSCP.exe" /console

Для удобства работы с командной строкой WinSCP можно добавить путь к каталогу программы в переменную PATH стандартными средствами Windows либо средствами самой WinSCP: Параметры - Настройки - Интеграция - Добавить папку WinSCP в путь поиска. Поскольку добавление пути поиска WinSCP выполняется в системную переменную PATH, требуется запуск программы от имени администратора и перезагрузка Windows для применения изменений.

После выполнения команды winscp.com или winscp.exe /console , откроется стандартное консольное окно с приглашением к вводу команд WinSCP:

winscp >

По умолчанию, в окне командной строки WinSCP используется кодировка UTF-8. При желании, внешний вид и поведение консоли WinSCP можно настроить под свои предпочтения, например, включив выделение текста мышью и сменив цветовую палитру на черные символы на белом фоне.

Получение справки по командам WinSCP

Большинство внутренних команд WinSCP имеют синтаксическое и смысловое сходство с командами командной строки Linux. Для получения встроенной справки используется команда help:

Подсказка по командам WinSCP.

call - Выполняет заданную команду на сервере

cd - Изменяет папку на сервере

checksum - Вычисляет контрольную сумму файлов на сервере

chmod - Изменить права доступа к файлу на сервере

close - Закрывает соединение

cp - Дупликация/дубль удалённого файла

echo - Выводит свои аргументы в виде сообщения

exit - Закрывает все соединения и завершает программу

get - Загрузить файл с сервера

help - Отображает справку

keepuptodate - Постоянно отслеживать изменения в локальной папке

lcd - Изменить локальную папку

lls - Отобразить содержимое локальной папки

ln - Создать ссылку на сервере

lpwd - Отобразить содержимое локальной папки

ls - Отобразить содержимое папки на сервере

mkdir - Создать папку на сервере

mv - Переименование/перенос файла на сервере

open - Соединение с сервером

option - Просмотреть/задать параметры сценария

put - Выгрузить файл на сервер

pwd - Печатает имя папки сервера

rm - Удалить файл с сервера

rmdir - Удалить папку с сервера

session - Показать список активных соединений или выбрать активное соединение

stat - Запрашивает атрибуты файла на сервере

synchronize - Синхронизировать папку на сервере с локальной

Получение справки по конкретной команде WinSCP

Для получения дополнительной справочной информации по отдельным командам используется синтаксис:

help команда

help synchronize - получить подсказку по команде синхронизации. Результат выполнения:

synchronize local|remote|both [< локальный каталог > [< каталог на сервере >]]

Если первый параметр — 'local', синхронизирует локальный каталог с удалённым Если первый параметр — 'remote', синхронизирует удалённый каталог с локальным. Если первый параметр — 'both', взаимно синхронизирует каталоги. Если каталоги не указаны, синхронизируются текущие рабочие каталоги.

Примечание: подтверждения перезаписи для этой команды всегда отключены.

BR> параметры:

-preview - Только просмотр различий, не синхронизировать

-delete - Удалить устаревшие файлы

-mirror - Зеркалирование (синхронизирует старые файлы тоже). Игнорируется в режиме 'both'.

-criteria=< критерии > - Критерии сравнения. Возможные значения: 'none', 'time', 'size' и 'either'. Игнорируется в режиме 'both'.

-permissions=< код > - Установить права

-nopermissions - Сохранить права по умолчанию

-speed=< кбит/с > - Ограничить скорость передачи

-transfer=< режим > - Режим передачи: binary, ascii, automatic

-filemask=< маска > - Задаёт файловую маску.

-resumesupport=< состояние > - Настраивает поддержку возобновления. Возможные значения: 'on', 'off' или пороговая величина

действующие параметры: reconnecttime

примеры:

synchronize remote -delete

synchronize both d:\www /home/martin/public_html

Использование собственных сценариев WinSCP .

WinSCP позволяет пользователю получить уже готовые сценарии для выполнения большинства операций приема и передачи файлов. Например, при приеме файла от удаленного сервера в локальный каталог, после нажатия кнопки Получить можно открыть выпадающее меню Настройки передачи

Настройка передачи в WinSCP.

Пункт меню Сформировать код… позволяет сформировать готовый сценарий для выполнения в виде обычного командного файла Windows, сценарий для выполнения в командной строке WinSCP, а также коды сборки .NET для C#, VB.NET и Power Shell. В открывшемся окне Сформировать код передачи нужно переключиться на вкладку Сценарий и выбрать нужный Формат, например Файл сценария

Файл сценария для выполнения в командной строке WinSCP.

Выбранный сценарий нужно сохранить в каком-нибудь файле в кодировке UTF-8 (UTF-16), например C:\Scripts\s1.txt и передать его на выполнение в виде параметра командной строки winscp:

winscp.com /script=c:\Scripts\s1.txt

Если каталог WinSCP не добавлен в пути поиска переменной PATH, по нужно указать полный путь к исполняемому файлу:

"C:\Program Files (x86)\WinSCP\inscp.com" /script=c:\Scripts\s1.txt /ini=nul

Очень желательно, в параметрах командной строки указать параметр /ini=nul. Это делается для того, чтобы обеспечить выполнение winscp.com с конфигурацией по умолчанию и запрещает сохранение текущей конфигурации при завершении программы.

В процессе выполнения сценария, на экране отображается справочная информация:

Соединяюсь с site.com…

Соединение установлено

Открываю соединение…

Соединение открыто.

Активные соединения:

/utility/Backplane_Utility

C:\Backplane_Utility

backplane.zip | 1728 KB | 248,6 KB/s | binary | 100%

При необходимости, можно включить протоколирование выполнения сценария в журнале, например, C:\Scripts\s1.log :

winscp.com /script=c:\Scripts\s1.txt /log=C:\Scripts\s1.log /ini=nul

Для получения командного файла CMD Windows, необходимо выбрать Формат - Пакетный файл:

Командный файл, созданный WinSCP.

Содержимое командного файла копируется в буфер обмена и затем в командный файл, например, C:\Scripts\s1.bat. В полученный таким образом командный файл нужно внести некоторые изменения, указав действительные путь для файла журнала или отключить его ведение:

```
@echo off
"C:\Program Files (x86)\WinSCP\WinSCP.com" ^
/log="C:\Scripts\s1.log" /ini=nul ^
/command ^
```

```
"open ftp://anonymous@site.com/" ^
"cd /utility/Backplane Utility" ^
"lcd C:\Backplane_Utility" ^
"get backplane.zip" ^
"exit"
REM Ниже пример анализа кода возврата для оценки результата выполнения сценария set WINSCP_RESULT = %ERRORLEVEL%
if %WINSCP_RESULT% equ 0 (
echo Success
) else (
echo Error
)
exit /b %WINSCP_RESULT%
```

WinSCP возвращает ERRORLEVEL равный нулю, если сценарий выполнен успешно. Команда exit /b %WINSCP_RESULT% формирует ERRORLEVEL для данного командного файла, который может быть использован в других сценариях для анализа результатов его выполнения.

Если выбрать Формат - Командная строка, то в качестве кода передачи будут сформированы параметры командной строки winscp.com для выполнении в среде командного процессора Windows. Как и в случае с командным файлом, потребуется некоторая правка пути журнала или его исключение. Например:

winscp.com /command "open ftp://anonymous@site.com/" "cd /Backplane_Utility" "lcd C:\Backplane_Utility" "get backplane.zip" "exit"

Для получения сценариев на языках C#, VB.NET и PowerShell, используется вкладка Код сборки .NET

Код сборки .NET, создаваемой в среде WinSCP.

Полученный код для выполнения в среде PowerShell копируется в файл с расширением .ps1, например - C:\Scripts\s1.ps1 и запускается на выполнение командой:

powershell -file C:\Scripts\s1.ps1

По умолчанию, в Power Shell включена максимальная политика безопасности, которая позволяет выполнять команды PowerShell в командной строке, но не позволяет выполнять в ней заранее подготовленные сценарии. Поэтому, если на экране отобразится сообщение о том, что невозможно загрузить файл сценария, так как выполнение скриптов запрещено для данной системы, нужно выполнить команду:

powershell -Command Set-ExecutionPolicy RemoteSigned

После выполнения данной команды, выполнение сценариев в среде PowerShell будет разрешено.

Полученный код сборки для PowerShell, может потребовать некоторой правки, в частности - указания полного пути для загрузки динамической библиотеки WinSCPnet.dll.

Загрузить сборку .NET WinSCP

Add-Type -Path "C:\Program Files (x86)\WinSCP\WinSCPnet.dll"

. . .

Использование планировщика заданий для автоматизации выполнения сценариев

Для запуска Планировщика заданий можно воспользоваться поиском в Windows, перейти в “Панель управления” - “Администрирование” - “Планировщик заданий”, либо выполнить команду taskschd.msc.

Библиотека планировщика заданий, отображаемая в левой части окна оснастки планировщика, имеет довольно непростую иерархическую структуру, поэтому, можно создать отдельную папку, с использованием контекстного меню, вызываемого правой кнопкой мышки и пункта Создать

папку, ввести имя папки, и в дальнейшем, именно в ней создавать свои тестовые или рабочие задания.

Для создания задач планировщика могут использоваться два мастера, вызываемые в режимах Создать простую задачу и Создать задачу. При создании простой задачи используется минимальный набор параметров, не предусматривающий наличие множественных условий выполнения и множественных действий. Для запуска сценариев WinSCP по расписанию, вполне достаточно создания простой задачи.

Новую задачу можно создать с использованием пункта меню Действие - Создать простую задачу либо через контекстное меню, вызываемое правой кнопкой мышки на уровне созданной папки в библиотеке планировщика, либо на уровне "Библиотека планировщика", если такая папка не нужна. После чего запускается мастер создания задачи:

Мастер создания задачи для планировщика заданий Windows

На шаге Общие введите имя задания и его описание. Имя, для примера - DownloadFiles, а описание "Download files from site.com". Описание может быть произвольным текстом, но желательно, чтобы оно отражало суть создаваемой задачи.

На шаге Триггер выполняется настройка условий, при возникновении которых, будет запущена создаваемая задача.

Триггер задачи для планировщика заданий Windows

На шаге Действия в качестве действия выбираем Запуск программы, в виде которой будет выступать созданный ранее командный файл Script1.bat, который нужно выбрать с использованием кнопки Обзор….

Действия задачи для планировщика заданий Windows

На шаге Завершение нажмите кнопку Готово и задание будет создано. Настройки созданного задания можно изменить в любое время при наличии у пользователя соответствующих прав.

Протокол FTP

Когда интернет только зарождался, но уже были компьютерные сети, возникла потребность передавать файлы от одного компьютера к другому. В 1971 году каналы передачи данных были не такие надёжные (и не такие быстрые), как сейчас, поэтому нужен был инструмент, который поможет обмениваться документами друг с другом на расстоянии.

Основные требования были такие: простота работы и надёжность при отправке и получении. Таким инструментом стал FTP-протокол.

Принцип работы

FTP расшифровывается как File Transfer Protocol — протокол передачи файлов. Он отличается от других протоколов тем, что если в процессе передачи возникает какая-то ошибка, то процесс останавливается и выводится сообщение для пользователя. Если ошибок не было, значит, пользователь получил именно тот файл, который нужен, в целости и без недостающих элементов.

По FTP-протоколу можно скачивать что угодно: фильмы, музыку, документы, программы, драйверы и картинки. Сейчас многие производители железа выкладывают драйверы от устройств на FTP-серверы, чтобы их могли скачать все желающие.

В корпоративной среде FTP используется для организации локального хранилища внутренних документов и файлов для работы. Например, там могут храниться видеолекции или архивные сканы документов. Ещё FTP позволяет загружать свои файлы на сервер, чтобы их мог скачать любой желающий.

Программисты иногда используют такие серверы для обмена файлами и для бэкапов кода, хотя многие для этого предпочитают GIT. Про него ещё поговорим отдельно.

Клиент и сервер

Для работы по FTP нужны двое: FTP-сервер и FTP-клиент. Что делает сервер:

обеспечивает доступ по логину и паролю к нужным файлам;

показывает пользователю только те файлы и папки, которые он может просматривать или загружать в них;

следит за качеством передачи и смотрит, чтобы не было ошибок;

управляет параметрами соединения в пассивном режиме.

Так как FTP пришёл к нам из времён UNIX-систем, то любое соединение требует логина и пароля. Если у пользователя его нет, сервер его не пропустит. Но чтобы сделать файлы доступными для всех, используют анонимный режим. В нём логином будет слово anonymous, а паролем — любой адрес электронной почты. Современные браузеры умеют сами заходить на анонимные FTP-серверы и подставлять почту. Со стороны это выглядит так, как будто никакого логина и пароля нет, но они есть.

Когда запускается FTP-сервер, ему говорят: «Уважаемый сервер, вот список файлов и папок, которые нужно показывать на сервере. Если к тебе постучится пользователь с таким-то логином и паролем, то покажи ему всё, а если с вот таким логином — то дай ему одну только эту папку. Анонимов не пускать». Ещё один обязательный параметр — адрес сервера и порт, по которому будет идти передача файлов.

Чтобы подключиться к серверу, нужна специальная программа, их ещё называют FTP-клиентами. Для каждой операционной системы есть много своих клиентов, например, FileZilla или CuteFTP. Те, кто работает в Linux-подобных системах, часто используют командную строку.

Для FTP не нужен сайт, то есть веб-интерфейс. Не нужно запускать веб-сервер, настраивать шаблоны вывода списка файлов и поднимать отдельную программу, которая будет нам отдавать эти файлы (типа Вордпресса). FTP — это как доступ к удаленной папке: ты сразу видишь файлы и можешь их качать, без посредников. А в вебе нужна какая-то программа, которая «нарисует» тебе файловую систему и поставит ссылки на файлы.

В FTP уже реализованы вопросы авторизации и прав. А в вебе их нужно создавать: например, ставить тот же Вордпресс и к нему прикручивать плагины с системой доступа. Или настраивать Apache, генерировать ключи доступа, раскладывать конфигурационные файлы по папкам — это гораздо менее элегантно, чем настройка FTP.

В FTP можно разрешить или запретить отдельным пользователям загружать файлы на FTP-сервер. В вебе загрузка файлов от пользователя на сервер — это на порядок более сложная задача.

Уязвимости и надёжность

Сам по себе FTP-протокол надёжен и гарантированно доставляет пользователю нужные файлы, если с соединением всё в порядке.

Проблема в том, что протокол изначально был незащищённый, и предполагалось, что канал передачи данных всегда надёжен. Поэтому в FTP всё передаётся в открытом виде: файлы, пароли, имена пользователей и любые данные.

Сейчас по умолчанию предполагается, что каждый канал — ненадёжный, и что данные нужно дополнительно шифровать. FTP этого не поддерживает. Если кто-то будет перехватывать ваш Wi-Fi-трафик или подключится к вашей локальной сети, то он сможет перехватить все эти данные и скачать их себе, параллельно с вами.

Ещё есть вопрос безопасности входа: по умолчанию у FTP-протокола нет защиты от подбора пароля и попыток входа, поэтому кто-то может просто перебрать доступные пароли, чтобы получить доступ к папкам. Если вы видели в фильмах про хакеров, как они там перебирают пароли при входе — это вполне вероятная ситуация для FTP.

С точки зрения современной безопасности правильным решением будет использовать одну из реализаций шифрованного FTP (FTPS, SFTP) или пользоваться FTP через VPN

Протокол TFTP

TFTP (англ. Trivial File Transfer Protocol – простой протокол передачи файлов) используется главным образом для первоначальной загрузки бездисковых рабочих станций. TFTP, в отличие от FTP, не содержит возможностей аутентификации и основан на транспортном протоколе UDP.

Основное назначение TFTP – обеспечение простоты реализации клиента. В связи с этим, он используется для загрузки бездисковых рабочих станций, загрузки обновлений и конфигураций в «умные» сетевые устройства, записи статистики с мини-АТС (CDR) и аппаратных маршрутизаторов/межсетевых экранов.

Поскольку протокол не поддерживает аутентификации, единственный метод идентификации клиента – это его сетевой адрес, который может быть подделан. Обычно, в Unix-системах, tftpd доступен только каталог /tftpboot. Однако в старых TFTP-серверах было

возможным получить файл паролей командой «RRQ ../etc/passwd».

Демон tftpd (одна из реализаций tftp-сервера) отказывается обрабатывать файлы, содержащие в своём имени комбинацию «/../» или начинающуюся с «../». Запись разрешается только в файлы, которые уже существуют, любого размера, и доступны для публичной записи права доступа: -rw-rw-rw-.

Дополнительная защита от доступа к произвольным файлам осуществляется с помощью смены корневого каталога на каталог tftpd.

Сначала в TFTP-пакете идет поле размером в 2 байта, определяющее тип пакета:

- Read Request (RRQ, #1) – запрос на чтение файла;
- Write Request (WRQ, #2) – запрос на запись файла;
- Data (DATA, #3) – данные, передаваемые через TFTP;
- Acknowledgment (ACK, #4) – подтверждение пакета;
- Error (ERR, #5) – ошибка.
- Option Acknowledgment (OACK, #6) – подтверждение опций.

Для начала передачи данных, клиент должен послать серверу WRQ или RRQ-пакет. У обоих пакетов формат одинаковый:

- 0x01/0x02 (тип пакета) – Имя файла;
- 0x00 (конец строки) – Режим передачи;
- 0x00 (конец строки) – Опции.

После получения RRQ-пакета сервером, он сразу начинает передачу данных. В случае с WRQ-запросом – сервер должен прислать ACK-пакет с номером пакета 0.

После получения запроса RRQ, сервер сразу посылает, в качестве подтверждения пакет с данными и с ID пакета, равным единице. В WRQ в качестве подтверждения используется ACK с ID, равным нулю. Всего по TFTP можно передать 32Мб, однако, из-за использования знакового int вместо без знакового, размер подтверждения ограничен 16 мегабайтами. Однако, если клиент и сервер поддерживают расширения протокола RFC 2347 и RFC 2348, то максимальный размер передаваемого файла увеличивается до 4Gb.

ОБЗОР ПОПУЛЯРНЫХ СУБД

СУБД (системы управления базами данных) - программное обеспечение, облегчающее использование баз данных.

Функции и классификации СУБД

СУБД выполняют следующие функции:

- управляют данными, размещенными на дисковых носителях;
- управляют данными в оперативной памяти с задействованием дискового кэша;

- сохраняют историю изменений (проводят журнализацию), создают резервные копии и восстанавливают содержимое БД, поврежденное в результате некорректного завершения работы;
- поддерживают используемые в БД языки, определяющие типы данных и манипулирующие ими.

В зависимости от того, какие способы представления и обработки данных выбирают основой для СУБД, эти системы могут относиться к:

- ***иерархическим системам управления***. Для этой модели характерно построение древовидной структуры данных, разделенной на различные уровни;
- ***сетевым СУБД***. Эта модель представляет собой расширенную версию иерархической модели не с одной записью - «предком» строго для каждого «потомка», а с несколькими, размещенными в одной сети;
- ***реляционным СУБД***. Эти системы управления используются в БД, представленных в виде двумерных таблиц с размещенными в них атрибутированными записями;
- ***объектно-ориентированным СУБД***. Эти системы управления работают с БД, в которых все данные сложно структурированы по классам и типам;
- ***объектно-реляционными СУБД***. Данные СУБД представляют собой комплексы, способные дополнительно выполнять объектно-ориентированные операции.

Наибольшее распространение среди существующих типов СУБД получили СУБД, работающие с реляционными базами данных. Они применяются преимущественно при создании различных web-продуктов.

Обзор самых популярных СУБД

Самыми популярными при решении задач Data Mining являются СУБД:

- ***MySQL*** - реляционная СУБД, имеющая открытый исходный код, позволяющая поддерживать табличные БД с простой структурой и сложными условиями запросов. Она отличается гибкостью и высокой скоростью обработки информации, простотой интерфейса и способностью синхронизации с другими БД и используется для построения прогностических моделей в e-commerce, IT и финтехе (то есть в сферах, где наиболее активно применяется Data Mining);
- ***Microsoft SQL Server***. Эта фирменная разработка Microsoft, подходящая к установке в ОС Windows и Linux. Ее характеризуют простой интерфейс, надежность сохранности данных и совместимость с с различными программными продуктами Windows. В интеллектуальном анализе данных эти СУБД используются главным образом для обработки данных из Microsoft Excel;
- ***Oracle*** - объектно-реляционная СУБД с простой установкой и первоначальной настройкой, с возможностью расширения функционала. Ей присущи высокие надежность, практичность и быстродействие. В частности, в Data Mining для e-commerce эта СУБД позволяет решать задачи определения эффективности различных видов продаж и построения моделей потребностей клиентов в зависимости от ситуации на рынках;
- ***PostgreSQL*** - объектно-реляционная СУБД, предназначенная для работы с базами данных различных сайтов и web-сервисов. Она подходит практически ко всем популярным платформам и используется в облачных сервисах.

Самостоятельно прочитать про каждую СУБД подробнее.

PostgreSQL

Поддержка стандартов, возможности, особенности

PostgreSQL базируется на языке SQL и поддерживает многие из возможностей стандарта SQL:2011.

В PostgreSQL версии 12 есть следующие ограничения

- Максимальный размер базы данных Нет ограничений
- Максимальный размер таблицы 32 Тбайт
- Максимальный размер поля 1 Гбайт
- Максимум записей в таблице Ограничено размерами таблицы
- Максимум полей в записи 250—1600, в зависимости от типов полей
- Максимум индексов в таблице Нет ограничений

Сильными сторонами PostgreSQL считаются:

- высокопроизводительные и надёжные механизмы транзакций и репликации;
- расширяемая система встроенных языков программирования: в стандартной поставке поддерживаются PL/pgSQL, PL/Perl, PL/Python и PL/Tcl; дополнительно можно использовать PL/Java, PL/PHP, PL/Py, PL/R, PL/Ruby, PL/Scheme, PL/sh и PL/V8, а также имеется поддержка загрузки модулей расширения на языке C[10];
- наследование;
- возможность индексирования геометрических (в частности, географических) объектов и наличие базирующегося на ней расширения PostGIS;
- встроенная поддержка слабоструктурированных данных в формате JSON с возможностью их индексации;

- расширяемость (возможность создавать новые типы данных, типы индексов, языки программирования, модули расширения, подключать любые внешние источники данных).

История

PostgreSQL создана на основе некоммерческой СУБД Postgres, разработанной как open-source проект в Калифорнийском университете в Беркли. К разработке Postgres, начавшейся в 1986 году, имел непосредственное отношение Майкл Стоунбрейкер, руководитель более раннего проекта Ingres, на тот момент уже приобретённого компанией Computer Associates. Название расшифровывалось как «Post Ingres», и при создании Postgres были применены многие ранние наработки.

Стоунбрейкер и его студенты разрабатывали новую СУБД в течение восьми лет с 1986 по 1994 год. За этот период в синтаксис были введены процедуры, правила, пользовательские типы и другие компоненты. В 1995 году разработка снова разделилась: Стоунбрейкер использовал полученный опыт в создании коммерческой СУБД Illustra, продвигаемой его собственной одноимённой компанией (приобретённой впоследствии компанией Informix), а его студенты разработали новую версию Postgres — Postgres95, в которой язык запросов POSTQUEL — наследие Ingres — был заменен на SQL.

Разработка Postgres95 была выведена за пределы университета и передана команде энтузиастов. Новая СУБД получила имя, под которым она известна и развивается в текущий момент — PostgreSQL.

Основные возможности

Этот раздел имеет чрезмерный объём или содержит маловажные подробности.

Если вы не согласны с этим, пожалуйста, покажите в тексте существенность излагаемого материала. В противном случае раздел может быть удалён. Подробности могут быть на странице обсуждения.

Функции

Функции являются блоками кода, исполняемыми на сервере, а не на клиенте БД. Хотя они могут писаться на чистом SQL, реализация дополнительной логики, например, условных переходов и циклов, выходит за рамки SQL и требует использования некоторых языковых расширений. Функции могут писаться с использованием одного из следующих языков:

Встроенный процедурный язык PL/pgSQL, во многом аналогичный языку PL/SQL, используемому в СУБД Oracle;

Скриптовые языки — PL/Lua, PL/LOLCODE, PL/Perl, PL/PHP, PL/Python, PL/Ruby, PL/sh, PL/Tcl, PL/Scheme, PL/v8 (Javascript);

Классические языки — C, C++, Java (через модуль PL/Java);

Статистический язык R (через модуль PL/R).

PostgreSQL допускает использование функций, возвращающих набор записей, который далее можно использовать так же, как и результат выполнения обычного запроса.

Функции могут выполняться как с правами их создателя, так и с правами текущего пользователя.

Иногда функции отождествляются с хранимыми процедурами, однако между этими понятиями есть различие. С девятой версии возможно написание автономных блоков, которые позволяют выполнять код на процедурных языках без написания функций, непосредственно в клиенте.

Триггеры

Триггеры определяются как функции, инициируемые DML-операциями. Например, операция INSERT может запускать триггер, проверяющий добавленную запись на соответствия определённым условиям. При написании функций для триггеров могут использоваться различные языки программирования (см. выше).

Триггеры ассоциируются с таблицами. Множественные триггеры выполняются в алфавитном порядке.

Правила и представления

Механизм правил (англ. rules) представляет собой механизм создания пользовательских обработчиков не только DML-операций, но и операции выборки. Основное отличие от механизма триггеров заключается в том, что правила срабатывают на этапе разбора запроса, до выбора оптимального плана выполнения и самого процесса выполнения. Правила позволяют переопределять поведение системы при выполнении SQL-операции к таблице. Хорошим примером является реализация механизма представлений (англ. views): при создании представления создается правило, которое определяет, что вместо выполнения операции выборки к представлению система должна выполнять операцию выборки к базовой таблице/таблицам с учётом условий выборки, лежащих в основе определения представления. Для создания представлений, поддерживающих операции обновления, правила для операций вставки, изменения и удаления строк должны быть определены пользователем.

Индексы

В PostgreSQL имеется поддержка индексов следующих типов: B-дерево, хеш, GiST, GIN, BRIN, Bloom. При необходимости можно создавать новые типы индексов. Индексы в PostgreSQL обладают следующими свойствами:

- возможен просмотр индекса не только в прямом, но и в обратном порядке — создание отдельного индекса для работы конструкции ORDER BY ... DESC не нужно;
- возможно создание индекса над несколькими столбцами таблицы, в том числе над столбцами различных типов данных;

- индексы могут быть функциональными, то есть строиться не на базе набора значений некоего столбца/столбцов, а на базе набора значений функции от набора значений;
- индексы могут быть частичными, то есть строиться только по части таблицы (по некоторой её проекции); в некоторых случаях это помогает создавать намного более компактные индексы или достигать улучшения производительности за счёт использования разных типов индексов для разных (например, с точки зрения частоты обновления) частей таблицы;
- планировщик запросов может использовать несколько индексов одновременно для выполнения сложных запросов.

Многоверсионность (MVCC)

PostgreSQL поддерживает одновременную модификацию БД несколькими пользователями с помощью механизма Multiversion Concurrency Control (MVCC). Благодаря этому соблюдаются требования ACID и практически отпадает нужда в блокировках чтения.

Типы данных

PostgreSQL поддерживает большой набор встроенных типов данных:

- Численные типы
  - Целые
  - С фиксированной точкой
  - С плавающей точкой
  - Денежный тип (отличается специальным форматом вывода, а в остальном аналогичен числам с фиксированной точкой с двумя знаками после запятой)
- Символьные типы произвольной длины
- Двоичные типы (включая BLOB)

- Типы «дата/время» (полностью поддерживающие различные форматы, точность, форматы вывода, включая последние изменения в часовых поясах)
- Булев тип
- Перечисление
- Геометрические примитивы
- Интервалы (RANGE)
- Сетевые типы
  - IP и IPv6-адреса
  - CIDR-формат
  - MAC-адрес
- UUID-идентификатор
- XML-данные
- Массивы
- JSON
- Идентификаторы объектов БД
- Псевдотипы

Более того, пользователь может самостоятельно создавать новые требуемые ему типы и программировать для них механизмы индексирования с помощью GiST.

Пользовательские объекты

PostgreSQL может быть расширен пользователем для собственных нужд практически в любом аспекте. Есть возможность добавлять собственные:

- Преобразования типов
- Типы данных
- Домены (пользовательские типы с изначально наложенными ограничениями)

- Функции (включая агрегатные)
- Индексы
- Операторы (включая переопределение уже существующих)
- Процедурные языки

Наследование и партицирование

Таблицы могут наследовать характеристики и наборы полей от других таблиц (родительских). При этом данные, добавленные в порождённую таблицу, автоматически будут участвовать (если это не указано отдельно) в запросах к родительской таблице.

В PostgreSQL 10 был добавлен механизм партицирования таблиц. Партицирование предназначено для разделения одной таблицы на несколько, так называемые партиции. Партицирование схоже с наследованием, но имеет более дружелюбный к пользователю синтаксис и более строгие ограничения, что позволяет выполнять дополнительные оптимизации при планировании запросов.

Прочие возможности

- Соблюдение принципов ACID
- Соответствие стандартам ANSI, SQL-92, SQL-99, SQL:2003, SQL:2011
- Поддержка запросов с OUTER JOIN, UNION, UNION ALL, EXCEPT, INTERSECT и подзапросов
- Последовательности
- Контроль целостности
- Репликация
- Общие табличные выражения и рекурсивные запросы
- Аналитические функции

- Поддержка Юникода (UTF-8)
- Поддержка регулярных выражений в стиле Perl
- Встроенная поддержка SSL, SELinux и Kerberos
- Протокол разделяемых блокировок
- Подгружаемые расширения, поддерживающие SHA1, MD5, XML
- Расширения для написания сложных выборок, отчётов и т. д. (API открыт)
- Средства для генерации совместимого с другими системами SQL-кода и импорта из других систем
- Автономные блоки на доступных языках, а не только SQL
- Качество исходного кода

Согласно результатам автоматизированного исследования, различного ПО на предмет ошибок, проведённом в 2005 году, в исходном коде PostgreSQL было найдено 20 проблемных мест на 775 000 строк исходного кода (в среднем, одна ошибка на 39 000 строк кода). Для сравнения: MySQL — 97 проблем, одна ошибка на 8 000 строк кода; FreeBSD (целиком) — 306 проблем, одна ошибка на 2 500 строк кода; Linux (только ядро) — 950 проблем, одна ошибка на 800 строк кода.

Производные продукты

Лицензия PostgreSQL позволяет на его основе создавать различные, в том числе коммерческие, форки. Их известно несколько десятков.

На базе PostgreSQL компанией EnterpriseDB были разработаны другие варианты этой СУБД, являющиеся платными для коммерческого использования — Postgres Plus (состоит целиком только из продуктов с открытыми исходными кодами; плата требуется только при необходимости приобретения коммерческой поддержки продукта) и Postgres Plus Advanced Server (расширение PostgreSQL специальными возможностями для обеспечения совместимости с Oracle Database). В комплекте поставки данных

продуктов содержится набор ПО для разработчиков и администраторов баз данных:

- Postgres Studio — аналог phpPgAdmin;
- Postgres Plus Debugger — отладчик для кода на PL/pgSQL, интегрированный с предыдущим пакетом;
- Migration Studio — инструмент для автоматического преобразования баз данных из MySQL/Oracle в PostgreSQL

Существуют и другие коммерческие продукты, созданные на базе PostgreSQL и дополняющие её различными функциями:

- 2ndQPostgres компании 2nd Quadrant
- Postgres Pro компании Postgres Professional
- Fujitsu Enterprise Postgres компании Fujitsu

Самостоятельное изучение документации по ссылке:

Документация к PostgreSQL 13.3 (postgrespro.ru)

Архитектура клиент-сервер

Веб-приложение – это клиент-серверное приложение, в котором клиентом выступает браузер, а сервером – веб-сервер (в широком смысле).

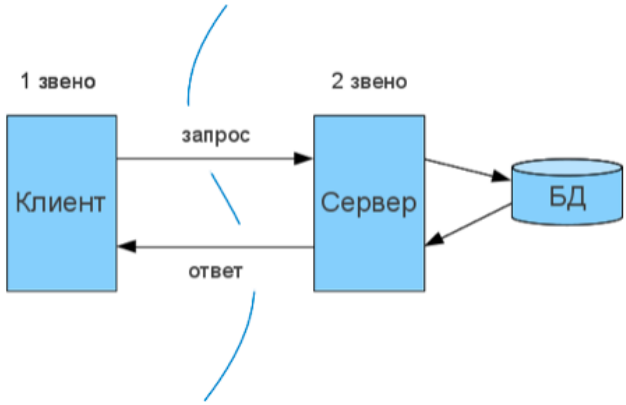
Основная часть приложения, как правило, находится на стороне веб-сервера, который обрабатывает полученные запросы в соответствии с бизнес-логикой продукта и формирует ответ, отправляемый пользователю. На этом этапе в работу включается браузер, именно он преобразовывает полученный ответ от сервера в графический интерфейс, понятный пользователю.

Архитектура «клиент-сервер» определяет общие принципы организации взаимодействия в сети, где имеются серверы, узлы-поставщики

некоторых специфичных функций (сервисов) и клиенты (потребители этих функций).

Практические реализации такой архитектуры называются **клиент-серверными технологиями.**

Двухзвенная архитектура - распределение трех базовых компонентов между двумя узлами (клиентом и сервером). Двухзвенная архитектура используется в клиент-серверных системах, где сервер отвечает на клиентские запросы напрямую и в полном объеме.



Расположение компонентов на стороне клиента или сервера определяет следующие основные модели их взаимодействия в рамках двухзвенной архитектуры:

- **Сервер терминалов** — распределенное представление данных.
- **Файл-сервер** — доступ к удаленной базе данных и файловым ресурсам.
- **Сервер БД** — удаленное представление данных.
- **Сервер приложений** — удаленное приложение.

Клиент – это браузер, но встречаются и исключения (в тех случаях, когда один веб-сервер (ВС1) выполняет запрос к другому (ВС2), роль клиента играет веб-сервер ВС1). В классической ситуации (когда роль клиента выполняет браузер) для того, чтобы пользователь увидел графический интерфейс приложения в окне браузера, последний должен

обработать полученный ответ веб-сервера, в котором будет содержаться информация, реализованная с применением HTML, CSS, JS (самые используемые технологии). Именно эти технологии «дают понять» браузеру, как именно необходимо «отрисовать» все, что он получил в ответе.

Веб-сервер – это сервер, принимающий HTTP-запросы от клиентов и выдающий им HTTP-ответы. Веб-сервером называют как программное обеспечение, выполняющее функции веб-сервера, так и непосредственно компьютер, на котором это программное обеспечение работает. Наиболее распространенными видами ПО веб-серверов являются Apache, IIS и NGINX. На веб-сервере функционирует тестируемое приложение, которое может быть реализовано с применением самых разнообразных языков программирования: PHP, Python, Ruby, Java, Perl и пр.

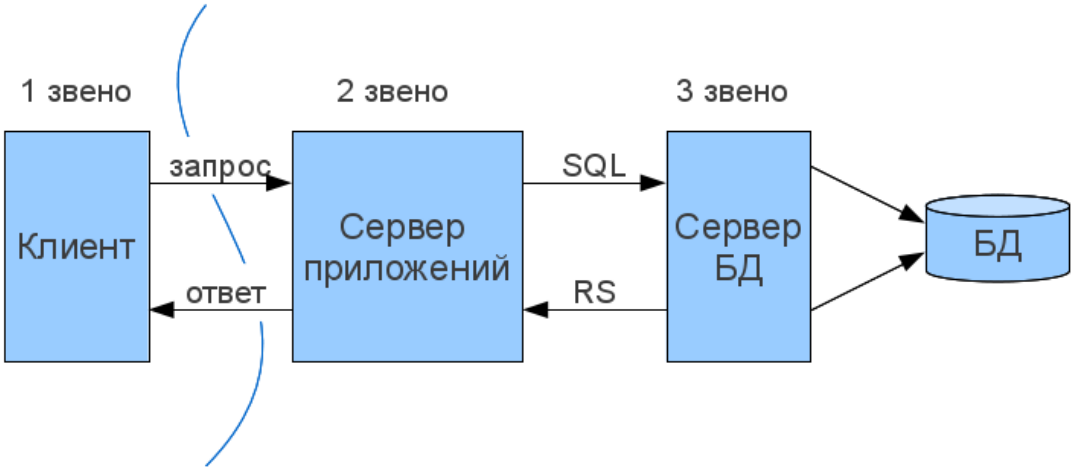
База данных фактически не является частью веб-сервера, но большинство приложений просто не могут выполнять все возложенные на них функции без нее, так как именно в базе данных хранится вся динамическая информация приложения (учетные, пользовательские данные и пр).

База данных - это информационная модель, позволяющая упорядоченно хранить данные об объекте или группе объектов, обладающих набором свойств, которые можно категоризировать. Базы данных функционируют под управлением так называемых систем управления базами данных (далее – СУБД). Самыми популярными СУБД являются MySQL, MS SQL Server, PostgreSQL, Oracle (все – клиент-серверные).

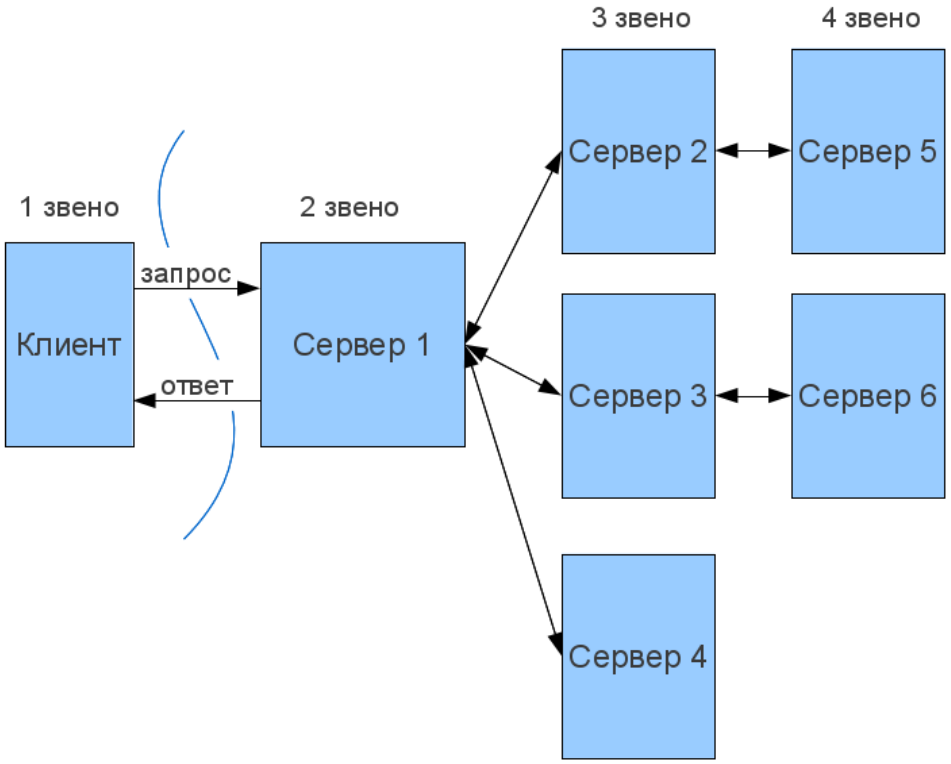
Трехзвенная архитектура - сетевое приложение разделено на две и более частей, каждая из которых может выполняться на отдельном компьютере. Выделенные части приложения взаимодействуют друг с другом, обмениваясь сообщениями в заранее согласованном формате.

Третьим звеном в трехзвенной архитектуре становится сервер приложений, т.е. компоненты распределяются следующим образом:

1. Представление данных — на стороне клиента.
2. Прикладной компонент — на выделенном сервере приложений (как вариант, выполняющем функции промежуточного ПО).
3. Управление ресурсами — на сервере БД, который и представляет запрашиваемые данные.



Трехзвенная архитектура может быть расширена до **многозвенной (N-tier, Multi-tier)** путем выделения дополнительных серверов, каждый из которых будет представлять собственные сервисы и пользоваться услугами прочих серверов разного уровня.



Двухзвенная архитектура проще, так как все запросы обслуживаются одним сервером, но именно из-за этого она менее надежна и предъявляет повышенные требования к производительности сервера.

Трехзвенная архитектура сложнее, но, благодаря тому, что функции распределены между серверами второго и третьего уровня, эта архитектура предоставляет:

1. Высокую степень гибкости и масштабируемости.
2. Высокую безопасность (т.к. защиту можно определить для каждого сервиса или уровня).
3. Высокую производительность (т.к. задачи распределены между серверами).

Клиент-серверные технологии

Архитектура клиент-сервер применяется в большом числе сетевых технологий, используемых для доступа к различным сетевым сервисам.

Типы сервисов:

- **Web-серверы**

Изначально предоставляли доступ к гипертекстовым документам по протоколу HTTP (Hyper Text Transfer Protocol). Сейчас поддерживают расширенные возможности, в частности, работу с бинарными файлами (изображения, мультимедиа и т.п.).

- **Серверы приложений**

Предназначены для централизованного решения прикладных задач в некоторой предметной области. Для этого пользователи имеют право запускать серверные программы на исполнение. Использование серверов приложений позволяет снизить требования к конфигурации клиентов и упрощает общее управление сетью.

- **Серверы баз данных**

Серверы баз данных используются для обработки пользовательских запросов на языке SQL. При этом, СУБД находится на сервере, к которому и подключаются клиентские приложения.

- **Файл-серверы**

Файл-сервер хранит информацию в виде файлов и предоставляет пользователям доступ к ней. Как правило, файл-сервер обеспечивает и определенный уровень защиты от несанкционированного доступа

- **Прокси-сервер**

Во-первых, действует как посредник, помогая пользователям получить информацию из Интернета и, при этом, обеспечивая защиту сети.

Во-вторых, сохраняет часто запрашиваемую информацию в кэш-памяти на локальном диске, быстро доставляя ее пользователям, без повторного обращения к Интернету.

- **Файрволы** (брандмауэры)

Межсетевые экраны, анализирующие и фильтрующие проходящий сетевой трафик, с целью обеспечения безопасности сети.

- **Почтовые серверы**

Предоставляют услуги по отправке и получению электронных почтовых сообщений.

- **Серверы удаленного доступа (RAS)**

Эти системы обеспечивают связь с сетью по коммутируемым линиям. Удаленный сотрудник может использовать ресурсы корпоративной ЛВС, подключившись к ней с помощью обычного модема.

Для доступа к тем или иным сетевым сервисам используются клиенты, возможности которых характеризуются понятием «толщины». Оно определяет конфигурацию оборудования и программное обеспечение, имеющиеся у клиента. Рассмотрим возможные граничные значения:

«Тонкий» клиент

Этот термин определяет клиента, вычислительных ресурсов которого достаточно лишь для запуска необходимого сетевого приложения через web-интерфейс. Пользовательский интерфейс такого приложения формируется средствами статического HTML (выполнение JavaScript не предусматривается), вся прикладная логика выполняется на сервере. Для работы тонкого клиента достаточно лишь обеспечить возможность запуска web-браузера, в окне которого и осуществляются все действия. По этой причине web-браузер часто называют "универсальным клиентом".

«Толстый» клиент

Таковым является рабочая станция или персональный компьютер, работающие под управлением собственной дисковой операционной системы и имеющие необходимый набор программного обеспечения. К сетевым серверам «толстые» клиенты обращаются, в основном, за дополнительными услугами (например, доступ к web-серверу или корпоративной базе данных).

Так же под «толстым» клиентом подразумевается и клиентское сетевое приложение, запущенное под управлением локальной ОС. Такое приложение совмещает компонент представления данных (графический пользовательский интерфейс ОС) и прикладной компонент (вычислительные мощности клиентского компьютера).

В последнее время все чаще используется еще один термин:«rich»-client. «Rich» -клиент, своего рода, компромисс между «толстым» и «тонким» клиентом. Как и «тонкий» клиент, «rich»-клиент также представляет графический интерфейс, описываемый уже средствами XML и включающий некоторую функциональность толстых клиентов (например, интерфейс drag-and-drop, вкладки, множественные окна, выпадающие меню и т.п.)

Прикладная логика «rich»-клиента также реализована на сервере. Данные отправляются в стандартном формате обмена, на основе того же XML (протоколы SOAP, XML-RPC) и интерпретируются клиентом.

Некоторые основные протоколы «rich»-клиентов на базе XML приведены ниже:

- XAML (eXtensible Application Markup Language) — разработан Microsoft и используется в приложениях на платформе .NET.
- XUL (XML User Interface Language) — стандарт, разработанный в рамках проекта Mozilla, используется, например, в почтовом клиенте Mozilla Thunderbird или браузере Mozilla Firefox.
- Flex — мультимедийная технология на основе XML, разработанная Macromedia/Adobe.

Протокол передачи данных — набор соглашений интерфейса логического уровня, которые определяют обмен данными между различными программами. Эти соглашения задают единообразный способ передачи сообщений и обработки ошибок при взаимодействии ПО.

Сетевой протокол — набор правил и действий (очерёдности действий), позволяющий осуществлять соединение и обмен данными между двумя и более включёнными в сеть устройствами.

Сетевые протоколы:

TCP/IP — набор протоколов передачи данных, получивший название от двух принадлежащих ему протоколов: TCP (англ. Transmission Control Protocol) и IP (англ. Internet Protocol).

Наиболее известные протоколы, используемые в сети Интернет:

- **HTTP (Hyper Text Transfer Protocol)** — это протокол передачи гипертекста.

- **HTTPS (HyperText Transfer Protocol Secure)** - расширение протокола HTTP для поддержки шифрования, в целях повышения безопасности. Данные в протоколе HTTPS передаются поверх криптографических протоколов SSL или TLS.
- **SSL (Secure Sockets Layer — уровень защищённых сокетов)** — криптографический протокол, который подразумевает более безопасную связь.
- **FTP (File Transfer Protocol)** — это протокол передачи файлов со специального файлового сервера на компьютер пользователя.
- **POP3 (Post Office Protocol)** — это стандартный протокол почтового соединения.
- **SMTP (Simple Mail Transfer Protocol)** — протокол, который задает набор правил для передачи почты.
- **TELNET** — это протокол удаленного доступа.
- **DTN** — протокол, предназначенный для сетей дальней космической связи IPN, которые используются NASA.

Всё ПО для работы с протоколом HTTP разделяется на три большие категории:

1.Серверы - основные поставщики услуг хранения и обработки информации (обработка запросов).

2.Клиенты - конечные потребители услуг сервера (отправка запроса).

3.Прокси (посредники) - для выполнения транспортных служб.

Прокси-сервер (proxy — «представитель, уполномоченный») - промежуточный сервер (комплекс программ) в компьютерных сетях, выполняющий роль посредника между пользователем и целевым сервером (при этом о посредничестве могут как знать, так и не знать обе стороны), позволяющий клиентам как выполнять косвенные запросы (принимая и передавая их через прокси-сервер) к другим сетевым службам, так и получать ответы.