

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1. ПОСТАНОВКА ЗАДАЧИ	5
2. ОБЗОР СОВРЕМЕННЫХ МЕТОДОВ И ТЕХНОЛОГИЙ СЕРВЕРНОЙ ВИРТУАЛИЗАЦИИ	7
3. СИСТЕМНЫЙ АНАЛИЗ ВИРТУАЛЬНОЙ ИНФРАСТРУКТУРЫ . . .	25
3.1. Принцип конечной цели	25
3.2. Принцип единства	26
3.3. Принцип связности	27
3.4. Принцип модульности	29
3.5. Принцип функциональности	29
3.6. Принцип иерархии	30
3.7. Принцип сочетания централизации и децентрализации	31
3.8. Принцип развития	32
3.9. Принцип учета случайностей	33
4. ОПИСАНИЕ ВИРТУАЛЬНОЙ ИНФРАСТРУКТУРЫ	35
4.1. Общая схема инфраструктуры	35
4.2. Схема мониторинга	36
4.3. Схема репликации и балансировки нагрузки DNS-серверов	37
4.4. Используемые технологии виртуализации	38
4.5. Алгоритмы функционирования инфраструктуры	38
4.5.1 Схема взаимодействия пользователя с инфраструктурой	38
4.5.2 Алгоритм взаимодействия компонентов инфраструктуры	40
5. РУКОВОДСТВО АДМИНИСТРАТОРА	42
5.1. Расположение серверов в сетевой инфраструктуре	42
5.2. Сервер shared-хостинга	42
5.3. Сервер виртуализации OpenVZ	47
5.4. Сервер виртуализации KVM	54

5.5. Дополнительные виртуальные сервера	56
5.6. Защита от DDoS-атак	56
5.7. Общие рекомендации по администрированию инфраструктуры . .	57
6. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ	59
7. РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ	60
7.1. Нагрузка сети во время резервного копирования	61
7.2. Состояние сервера в период атаки на отказ	63
7.3. Отказ одного из DNS-серверов	66
8. БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ	67
8.1. Краткая характеристика помещения	67
8.2. Микроклимат	68
8.3. Освещение	68
8.4. Шум и вибрация	69
8.5. Пожаробезопасность	69
8.6. Электробезопасность	70
8.7. Эргономика и техническая эстетика	71
8.8. Режим труда и отдыха	72
8.9. Выводы	73
ЗАКЛЮЧЕНИЕ	74
БИБЛИОГРАФИЧЕСКИЙ СПИСОК	75
ПРИЛОЖЕНИЕ А	78
ПРИЛОЖЕНИЕ Б	85

ВВЕДЕНИЕ

Привет мир!!!

Облачные услуги — это способ предоставления, потребления и управления технологией. Данный тип услуг выводит гибкость и эффективность на новый уровень, путем эволюции способов управления, таких как непрерывность, безопасность, резервирование и самообслуживание, которые соединяют физическую и виртуальную среду. В данной сфере возрастает потребность в качественно продуманной архитектуре, позволяющей надежно и правильно организовать облачную инфраструктуру.

Для эффективной работы облачной инфраструктуры требуется эффективная структура и организация. Небольшая команда из специалистов и бизнес-пользователей может создать обоснованный план и организовать свою работу в инфраструктуре. Данная выделенная группа может намного эффективнее построить и управлять нестандартной облачной инфраструктурой, чем если компании будут просто продолжать добавлять дополнительные сервера и сервисы для поддержки центра обработки данных (ЦОД).

IaaS (Infrastructure as a Service) — это предоставление пользователю компьютерной и сетевой инфраструктуры, их обслуживание как услуги в форме виртуализации, то есть виртуальной инфраструктуры. Другими словами, на базе физической инфраструктуры дата-центров (ДЦ) провайдер создает виртуальную инфраструктуру, которую предоставляет пользователям как сервис. Стоит отметить, что IaaS не предполагает передачи в аренду программного обеспечения, а всего лишь предоставляет доступ к вычислительным мощностям.

Технология виртуализации ресурсов позволяет физическое оборудование (сервера, хранилища данных, сети передачи данных) разделить между пользователями на несколько частей, которые используются ими для выполнения текущих задач. К примеру, на одном физическом сервере можно запустить

сотни виртуальных серверов, а пользователю для решения задач выделить время доступа к ним.

1 ПОСТАНОВКА ЗАДАЧИ

Конечная цель проектирования — разработка виртуальной инфраструктуры для реализации облачных услуг.

Виртуальная инфраструктура должна обладать следующими характеристиками:

- использование, по возможности, продуктов, распространяющихся под свободной лицензией (GNU GPL) для организации инфраструктуры;
- устранение единой точки отказа при проектировании инфраструктуры;
- защита от распределенных атак на отказ (DDoS);
- использование инфраструктуры в бизнесе для предоставления облачных услуг клиентам.

Аппаратное обеспечение должно базироваться в дата-центре ориентированным на требования стандарта Tier III, состоящего из информационной, телекоммуникационной и инженерной инфраструктуры, с возможностью аренды выделенных серверов и аппаратной защиты от DDoS-атак, а также поддержкой аппаратного RAID. Рекомендуемые характеристики выделенных серверов виртуализации:

- 2xIntel® Xeon E5430 @ 2.66GHz;
- минимальный объем ОЗУ 32 Гб;
- минимум 960 Гб места на жестком диске (SSD);
- поддержка аппаратного RAID;
- операционная система не ниже Debian 7 GNU/Linux или CentOS 6.5;
- интернет-канал с пропускной способностью не менее 100 Мб/с;
- возможность добавления дополнительных IP-адресов к серверам;
- возможность удаленного доступа к серверам посредством IPMI (Intelligent Platform Management Interface).

Следует предусмотреть возможность развертывания дополнительных виртуальных серверов для организации DNS-серверов, системы мониторинга, управления адресным пространством и платежной системы. Также необходимо

предусмотреть наличие системы хранения данных (СХД) для резервных копий, а также расширение дискового пространства на выделенных серверах.

Для разработки виртуальной инфраструктуры необходима реализация следующих этапов:

- выбор технологий виртуализации;
- выбор физических серверов на основе услуги IaaS в ДЦ;
- приобретение лицензий на программное обеспечение;
- приобретение подсетей IP-адресов;
- создание и подключение инфраструктуры мониторинга, платежной системы (биллинга), резервного копирования;
- реализация аппаратной защиты от DDoS-атак;
- выбор перечня PaaS-услуг;
- внедрение перечня предоставляемых PaaS-услуг;
- создание руководства администратора по виртуализации;
- создание руководства для клиентов по часто задаваемым вопросам;
- внедрение инфраструктуры на рынке PaaS-услуг.

2 ОБЗОР СОВРЕМЕННЫХ МЕТОДОВ И ТЕХНОЛОГИЙ СЕРВЕРНОЙ ВИРТУАЛИЗАЦИИ

Понятие облачных вычислений на сегодняшний день уже достаточно хорошо известно и в информационных технологиях (ИТ) и бизнесе. Облачные вычисления являются новой сервисной моделью предоставления вычислительных услуг. Еженедельно появляются новые статьи, презентации, книги об облачных вычислениях, что говорит о заинтересованности сообщества в данных технологиях.

За время существования информационных технологий создавались и изменялись подходы к построению информационных систем. Первой моделью информационной системы была монолитная архитектура. В данной модели на одном компьютере работали и приложения и база данных (БД), а пользователи сидели у «тонких» терминалов которые отображали информацию с компьютера. У архитектуры было большое количество недостатков, поэтому впоследствии ее сменила более перспективная клиент-серверная архитектура. В этом случае на компьютере располагался выделенный сервер баз данных, а пользователи с «толстых клиентов» разгружали сервер БД. Затем появилась еще более современная многоуровневая архитектура, у которой логика приложений вынесена на отдельный компьютер, который называется сервер приложений, а пользователи работали на «тонких» клиентах через веб-браузеры. В современном информационном мире большинство приложений выполнено именно в многоуровневой архитектуре. Она подразумевает развертывание всей ИТ-инфраструктуры на территории заказчика [1].

Облачные вычисления являются следующим шагом в эволюции архитектуры построения информационных систем. Благодаря преимуществам данного подхода вполне очевидно, что многие информационные системы в ближайшее время переносятся или будут перенесены в облако. Процесс уже идет полным ходом и его игнорирование или недооценка может привести к поражению в конкурентной борьбе на рынке. В данном случае имеется в виду не только отставание ИТ или неоправданные затраты на него, но и отставание в развитии

бизнеса компании, которая зависит от гибкости информационной инфраструктуры и скорости вывода новых сервисов и продуктов на рынок.

В феврале 2011 года ИТ-директор американского правительства Вивек Кундра опубликовал стратегию переноса части информационных систем в облако. Стратегия была опубликована в документе под названием «Federal Cloud Computing Strategy», который четко описывает порядок и сроки переноса. Целью работ является уменьшение сложности и повышение управляемости ИТ, увеличение нагрузки оборудования до 70-80%, а также уменьшение количества центров обработки данных.

Дата-центр, иначе именуемый центром обработки данных, является специализированным зданием, в котором размещается серверное и сетевое оборудование, подключаемое абонентов к сети Интернет. Основное требование, предъявляемое к центрам обработки данных, отказоустойчивость. При этом подразумевается отключение дата-центра как на время планово-предупредительных работ и профилактики оборудования, так и внеплановых аварийных ситуаций.

Классификация Tier описывает надежность функционирования ЦОД и является необходимой для компаний, как желающих построить свой ЦОД, так и для арендующих чужие вычислительные мощности. В зависимости от критичности бизнеса и потерь, которые понесет компания в случае остановки бизнес-процессов, выбирается тот или иной уровень надежности. В свою очередь, высокий уровень надежности требует высоких материальных и эксплуатационных затрат, поэтому и стоимость вычислительных мощностей зависит от уровня надежности ЦОД [2].

На сегодняшний день существует четыре уровня надежности ЦОД названные Tier I, Tier II, Tier III и Tier IV, которые были введены организацией Uptime Institute (Институт бесперебойных процессов, США):

- Tier I: время простоя 28.8 часов в год, коэффициент отказоустойчивости 99.671%;
- Tier II: 22.0 часа в год, 99.749%;
- Tier III: 1.6 часа в год, 99.982%;

- Tier IV: 0.4 часа в год, 99.995%.

ЦОД уровня Tier I (базовый уровень) подвержен нарушениями работы как от плановых, так и от внеплановых действий. Применение фальшпола, источников бесперебойного питания (ИБП), дизель-генераторных установок (ДГУ) не обязательно. Если ИБП и ДГУ используются, то выбираются более простые модели, без резерва, с множеством точек отказа. Возможны самопроизвольные отказы оборудования. К простоям ЦОД также приведут ошибки в действиях обслуживающего персонала. В таких ЦОД отсутствует защита от случайных и намеренных событий, обусловленных действиями человека.

В ЦОД уровня Tier II (с резервированными компонентами) время простоя возможно в связи с плановыми и внеплановыми работами, а также аварийными ситуациями, однако оно сокращено благодаря внедрению одной резервной единицы оборудования в каждой системе. Таким образом, системы кондиционирования, ИБП и ДГУ имеют одну резервную единицу, тем не менее, профилактические работы требуют отключения ЦОД. В центрах обработки данных с резервированными компонентами требуется наличие минимальных защитных мер от влияния человека.

Третий уровень надежности (уровень с возможностью параллельного проведения ремонтных работ) требует осуществления любой плановой деятельности без остановки ЦОД. Под плановыми работами подразумевается профилактическое и программируемое техническое обслуживание, ремонт и замена компонентов, добавления или удаление компонентов, а также их тестирование. В таком случае необходимо иметь резерв, благодаря которому можно пустить всю нагрузку по другому пути во время работ на первом. Для реализации Tier III необходима схема резервирования блоков схем кондиционирования, ИБП, ДГУ N+1, также требуется наличие двух комплектов трубопроводов для системы кондиционирования, построенной на основе чиллера (холодильной машины). Строительные требования обязывают сохранять работоспособность ЦОД при большинстве случаев намеренных и случайных вмешательств человека. Также следует предусмотреть резервные входы, дублирующие подъездные

пути, контроль доступа, отсутствие окон, защиту от электромагнитного излучения.

Четвертый уровень надежности ЦОД (отказоустойчивый) характеризуется безостановочной работой при проведении плановых мероприятий и способен выдержать один серьезный отказ без последствий для критически важной нагрузки. Необходим дублированный подвод питания, резервирование системы кондиционирования и ИБП по схеме $2(N+1)$. Для дизель-генераторных установок необходима отдельная площадка с зоной хранения топлива. Tier IV требует защиту от всех потенциальных проблем в связи с человеческим фактором. Регламентированы избыточные средства защиты от намеренных или случайных действий человека. Учтено влияние сейсмоявлений, потопов, пожаров, ураганов, штормов, терроризма.

Дата-центры по виду использования подразделяют на корпоративные и коммерческие (аутсорсинговые). Корпоративные ДЦ предназначены для обслуживания конкретной компании, коммерческие, в свою очередь, предоставляют услуги всем желающим.

Некоторые ДЦ предлагают клиентам дополнительные услуги по использованию оборудования, по автоматическому уходу от различных видов атак. Команда специалистов круглосуточно производит мониторинг серверов. Для обеспечения сохранности данных используются системы резервного копирования. Для предотвращения кражи данных, в дата-центрах используются различные системы ограничения физического доступа, системы видеонаблюдения.

Дата-центры предоставляют несколько основных типов услуг, среди которых:

- виртуальный хостинг (shared hosting);
- виртуальный сервер (virtual private/dedicated server);
- выделенный сервер (dedicated server);
- размещение сервера (colocation);
- выделенная зона (dedicated area).

Виртуальный хостинг используется для размещения большого количества сайтов на одном веб-сервере. В основном для построения веб-сервера

используется типичный стек технологий LAMP, где в качестве операционной системы выступает GNU/Linux, http-сервер Apache (зачастую в связке с nginx), сервер баз данных MySQL, интерпретируемые скриптовые языки PHP, Perl, Python. Существует решение на базе ОС Windows Server, где в качестве http-сервера используется IIS, в качестве СУБД выступает MS SQL, а также существует поддержка платформы ASP.NET. Разделение ресурсов на виртуальном хостинге основывается на ограничении дискового пространства, сетевого трафика, количества используемых доменов, почтовых ящиков, баз данных, FTP-аккаунтов, ограничение на использование процессорного времени, памяти для PHP-скриптов и так далее.

Виртуальный выделенный сервер эмулирует работу отдельного физического сервера. На одной физической машине может быть запущено несколько виртуальных серверов, при этом каждый виртуальный сервер имеет свои процессы, ресурсы и отдельное администрирование. Для реализации виртуальных машин используются технологии виртуализации, как системы с открытым исходным кодом, так и коммерческие.

В случае выделенного сервера клиенту целиком предоставляется отдельная физическая машина. Владелец сервера имеет возможность смены конфигурации оборудования, установки любой операционной системы. Такой тип хостинга подходит для высоконагруженных проектов.

Размещение сервера отличается от услуги предоставления выделенного сервера тем, что ДЦ размещает у себя сервер, который заранее подготовил клиент. Дата-центр подключает его в общую инфраструктуру ЦОДа, обеспечивает бесперебойное электропитание, охлаждение, доступ к сетевому каналу, удаленный доступ к серверу, охрану, мониторинг и другие услуги.

Выделенная зона предоставляется в основном для специальных клиентов, имеющих строгие нормы безопасности. В этом случае дата-центр предоставляет выделенную зону, обеспеченную электроснабжением, холодоснабжением и системами безопасности, а клиент сам создает свой дата-центр внутри этого пространства.

Также можно выделить такую услугу, как аренда телекоммуникационных стоек, которая является частным случаем размещения сервера, с отличием в том, что арендаторами в основном являются юридические лица.

При построении облачной инфраструктуры важную роль играет виртуализация.

Виртуализация — абстракция вычислительных ресурсов и предоставление пользователю системы, которая инкапсулирует (скрывает в себе) собственную реализацию. Проще говоря, пользователь работает с удобным для себя представлением объекта, и для него не имеет значения, как объект устроен в действительности. Термин «виртуализация» появился в шестидесятых годах XX века, а в девяностых — стали очевидны перспективы подхода: с ростом аппаратных мощностей, как персональных компьютеров, так и серверных решений, вскоре представится возможность использовать несколько виртуальных машин на одной физической платформе.

Понятие виртуализации можно условно разделить на две категории [3]:

- виртуализация платформ, продуктом этого вида виртуализации являются виртуальные машины — некие программные абстракции, запускаемые на платформе реально аппаратно-программных систем;
- виртуализация ресурсов преследует целью комбинирование или упрощение представления аппаратных ресурсов для пользователя и получение неких пользовательских абстракций оборудования, пространств имен, сетей.

Когда производится виртуализация, существует множество способов ее осуществления. Фактически есть несколько путей, с помощью которых достигаются одинаковые результаты через разные уровни абстракции [4]:

- эмуляция аппаратных средств;
- полная виртуализация;
- паравиртуализация;
- виртуализация уровня ОС.

Эмуляция аппаратных средств является одним из самых сложных методов виртуализации. В то же время главной проблемой при эмуляции аппа-

ратных средств является низкая скорость работы, в связи с тем, что каждая команда моделируется на основных аппаратных средствах. В эмуляции оборудования используется механизм динамической трансляции, то есть каждая из инструкций эмулируемой платформы заменяется на заранее подготовленный фрагмент инструкций физического процессора [5]. Архитектура процесса эмуляции представлена на рис. 2.1.



Рис. 2.1 – Эмуляция аппаратных средств

Примерами виртуализации посредством эмуляции являются программные платформы QEMU и Bochs.

Система QEMU поддерживает два режима эмуляции: пользовательский и системный. Пользовательский режим эмуляции позволяет процессу, созданному на одном процессоре, работать на другом (выполняется динамический перевод инструкций для принимающего процессора и конвертация системных вызовов Linux). Системный режим позволяет эмулировать систему целиком, включая процессор и разнообразную периферию. Достоинством QEMU является его быстрый и компактный динамический транслятор. Динамический транслятор позволяет во время исполнения переводить инструкции целевого (гостевого) процессора в инструкции центрального процессора хоста для обеспечения эмуляции. QEMU обеспечивает динамическую трансляцию преобразованием целевой инструкции в микрооперации. Эти микрооперации представляют собой элементы С-кода, которые компилируются в объекты. Затем

запускается основной транслятор, который отображает целевые инструкции на микрооперации для динамической трансляции. Такой подход не только эффективен, но и обеспечивает переносимость.

Использование QEMU в качестве эмулятора персонального компьютера обеспечивает поддержку разнообразных периферийных устройств. Сюда входят стандартные периферийные устройства — эмулятор аппаратного видеоадаптера (VGA), мыши и клавиатуры PS/2, интерфейс IDE для жестких дисков, интерфейс CD-ROM и эмуляция дисководов. Кроме того, QEMU имеет возможность эмуляции сетевых адаптеров NE2000 (PCI), последовательных портов, многочисленных звуковых плат и контроллера PCI Universal Host Controller Interface (UHCI), Universal Serial Bus (USB) (с виртуальным USB концентратором). Также поддерживается до 255 процессоров с поддержкой симметричной многопроцессорности (SMP).

Полная виртуализация использует гипервизор, который осуществляет связь между гостевой ОС и аппаратными средствами физического сервера. В связи с тем, что вся работа с гостевой операционной системой проходит через гипервизор, скорость работы данного типа виртуализации ниже чем в случае прямого взаимодействия с аппаратурой. Основным преимуществом является то, что в ОС не вносятся никакие изменения, единственное ограничение — операционная система должна поддерживать основные аппаратные средства. Архитектура полной виртуализации представлена на рис. 2.2.

Полная виртуализация возможна исключительно при условии правильной комбинации оборудования и программного обеспечения. Например, она была невозможной ни в серии IBM System/360, за исключением IBM System/360-67, ни в ранних IBM System/370, пока IBM не добавила оборудование виртуальной памяти в своих System/370 в 1972 г. Аналогичная ситуация и с платформой x86: полная виртуализация была возможна не в полной мере, до добавления технологий AMD-V и Intel VT.

KVM (Kernel-based Virtual Machine) — программное решение, обеспечивающее виртуализацию в среде Linux, которая поддерживает аппаратную виртуализацию на базе Intel VT (Virtualization Technology) либо AMD SVM (Secure

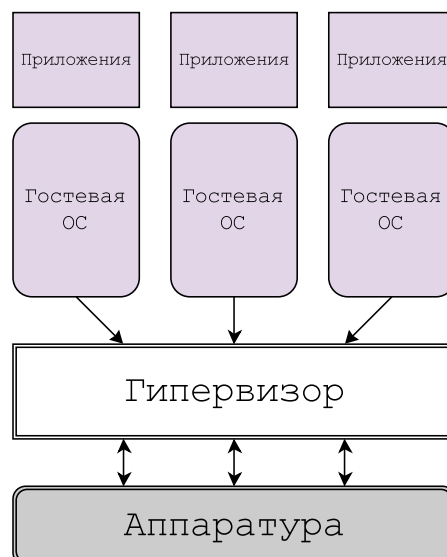


Рис. 2.2 – Архитектура полной виртуализации

Virtual Machine) [6]. KVM не выполняет никакой самоэмуляции, вместо этого, программа, работающая в пользовательском пространстве, применяет интерфейс `/dev/kvm` для настройки адресного пространства гостевого виртуального сервера, берет его смоделированные ресурсы ввода/вывода и отображает его образ на образ хоста.

В архитектуре KVM виртуальная машина выполняется как обычный Linux-процесс, запланированный стандартным планировщиком Linux. На самом деле, виртуальный процессор представляется как обычный Linux-процесс, это позволяет KVM пользоваться всеми возможностями ядра Linux. Эмуляцией устройств управляет модифицированная версия QEMU, которая обеспечивает эмуляцию BIOS, шины PCI, шины USB, а также стандартный набор устройств таких, как дисковые контроллеры IDE и SCSI, сетевые карты и другие.

Hyper-V является технологией виртуализации на основе гипервизора для 64х-разрядной версии операционной системы Windows Server 2008. Платформа позволяет нескольким изолированным операционным системам разделять одну аппаратную платформу. Hyper-V оперирует понятием «раздел». Стек виртуализации работает в родительском разделе и имеет прямой доступ к аппаратным устройствам. Корневой раздел создает дочерние разделы, в которых распола-

гаются гостевые ОС. Корневой раздел создает дочерние разделы с помощью интерфейса программирования приложений гипервызовов (API).

Разделы не имеют прямого доступа к физическому процессору и они не могут обрабатывать прерывания. Вместо этого они имеют доступ к виртуальному процессору и работают в виртуальной памяти. Гипервизор обрабатывает прерывания процессора и перенаправляет их к соответствующему разделу. Hyper-V позволяет ускорить трансляцию адресов между различными гостевыми виртуальными адресными пространствами с помощью IOMMU (Input Output Memory Management Unit), который работает независимо от аппаратного управления памятью используемой процессором.

Дочерние разделы также не имеют прямого доступа к аппаратным ресурсам, ресурсы в них представлены в качестве виртуальных устройств (VDevs). Запросы к виртуальным устройствам перенаправляются либо через VMBus, либо через гипервизор. Hyper-V требует процессор, которые поддерживает аппаратную виртуализацию (Intel VT или AMD-V).

Паравиртуализация имеет некоторые сходства с полной виртуализацией. Этот метод использует гипервизор для разделения доступа к основным аппаратным средствам, но объединяет код, касающийся виртуализации, в непосредственно операционную систему, поэтому недостатком метода является то, что гостевая ОС должна быть изменена для гипервизора. Но паравиртуализация существенно быстрее полной виртуализации, скорость работы виртуальной машины приближена к скорости реальной, это осуществляется за счет отсутствия эмуляции аппаратуры и учета существования гипервизора при выполнении системных вызовов в коде ядра. Вместо привилегированных операций совершаются гипервызовы обращения ядра гостевой ОС к гипервизору с просьбой о выполнении операции. Архитектура паравиртуализации представлена на рис. 2.3.

Для организации паравиртуализации используется программный продукт Xen.

Xen — это монитор виртуальных машин (VMM) или гипервизор с поддержкой паравиртуализации для процессоров архитектуры x86, распространя-

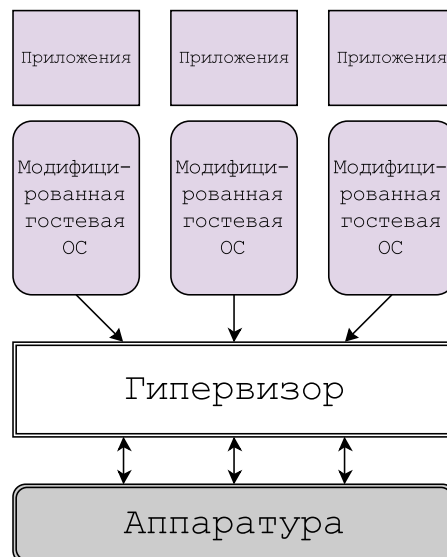


Рис. 2.3 – Архитектура паравиртуализации

ющийся с открытым исходным кодом. Xen может организовать совместное безопасное исполнение нескольких виртуальных машин на одной физической системе с производительностью, близкой к той, которая была бы непосредственно на физическом оборудовании. Он перекладывает большинство задач по поддержке аппаратуры на гостевую ОС, работающую в управляющей виртуальной машине, также известной как домен 0 (dom0) [7]. Сам Xen содержит только код, который необходим для обнаружения и запуска остальных процессоров системы, настройки обработки прерываний и нумерации шины PCI. Драйверы устройств работают не в Xen, а внутри привилегированной гостевой операционной системы. Такой подход обеспечивает совместимость с большинством устройств, поддерживаемых Linux. Сборка XenLinux по умолчанию содержит поддержку большого количества серверного сетевого и дискового оборудования, однако при необходимости можно добавить поддержку других устройств, скомпилировав ядро Linux обычным способом.

В паравиртуальном режиме (PV) оборудование не эмулируется, и гостевая операционная система должна быть специальным образом модифицирована для работы в таком окружении. Начиная с версии 3.0, ядро Linux поддерживает запуск в паравиртуальном режиме без перекомпиляции со сторонними патчами. Преимущество режима паравиртуализации состоит в том, что он не

требует поддержки аппаратной виртуализации со стороны процессора, а также не тратит вычислительные ресурсы для эмуляции оборудования на шине PCI. Режим аппаратной виртуализации (HVM), который появился в Xen, начиная с версии 3.0 гипервизора требует поддержки со стороны оборудования. В этом режиме для эмуляции виртуальных устройств используется QEMU, который весьма медлителен несмотря на паравиртуальные драйвера. Однако со временем поддержка аппаратной виртуализации в оборудовании получила настолько широкое распространение, что используется даже в современных процессорах лэптопов.

Виртуализация уровня операционной системы отличается от других. Она использует технику, при которой сервера виртуализируются непосредственно над ОС. Недостатком метода является то, что поддерживается одна единственная операционная система на физическом сервере, которая изолирует контейнеры друг от друга. Преимуществом виртуализации уровня ОС является «родная» производительность. Виртуализация уровня ОС — метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного. Эти экземпляры с точки зрения пользователя полностью идентичны реальному серверу. Для систем на базе UNIX эта технология может рассматриваться как улучшенная реализация механизма chroot. Ядро обеспечивает полную изолированность контейнеров, поэтому программы из разных контейнеров не могут воздействовать друг на друга. Архитектура виртуализации уровня ОС представлена на рис. 2.4.

Для реализации виртуализации уровня операционной системы часто используется продукт OpenVZ.

OpenVZ разрабатывается как патч к исходным текстам ядра Linux. В модифицированном ядре добавлен массив дополнительных сущностей — виртуальных окружений (VE) или контейнеров (CT), а для всех имеющихся объектов (процессы, сокеты и прочие) введены дополнительные поля — номер контейнера, к которому этот объект относится, и номер объекта внутри контейнера. Каждое виртуальное окружение имеет собственный набор квот на

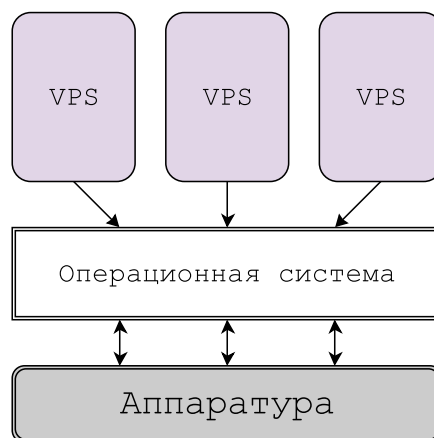


Рис. 2.4 – Архитектура виртуализации уровня ОС

потребление системных ресурсов и отдельный каталог для использования в качестве корневой файловой системы. Дополнительные модули ядра — `vzdev`, `vzmon` и прочие, отвечают за работу ограничений, мониторинг, эмуляцию сети в контейнере, сохранение и восстановление текущего состояния запущенных контейнеров. К преимуществам OpenVZ по сравнению с более универсальными инструментами виртуализации, такими, как Xen и KVM, относят прозрачный доступ из внешней системы к процессам, файлам и прочим ресурсам в контейнере.

OpenVZ разрабатывается фирмой Parallels как часть более крупного коммерческого продукта под названием Parallels Virtuozzo Containers (PVC) [8]. В число преимуществ Virtuozzo по сравнению с OpenVZ входят:

- файловая система VZFS;
- управление через графическую консоль и веб-интерфейс;
- программный интерфейс на базе XML для создания собственных инструментов управления и контроля;
- средства миграции с физической системы в контейнер и обратно;
- средства контроля за полосой и суммарным потреблением трафика;
- интеграция с Plesk, коммерческой панелью управления хостингом;
- круглосуточная техническая поддержка.

VZFS позволяет совмещать файловые системы контейнеров, при этом базовый образ используется всеми контейнерами, а изменения в нем для каждого контейнера сохраняются отдельно. Преимущества такого подхода:

- место, занимаемое программами на диске, становится фиксированным и не зависит от количества контейнеров, в которые эти программы установлены;
- уменьшается расход оперативной памяти, так как код нескольких экземпляров программы или библиотеки, запущенной из одного и того же исполняемого файла, размещается в памяти в единственном экземпляре;
- обновление программного обеспечения в группе контейнеров выполняется одной командой.

LXC (Linux Containers) — система виртуализации на уровне операционной системы. Данная система сходна с OpenVZ и Linux-VServer для Linux, а также FreeBSD jail и Solaris Containers. LXC основана на технологии cgroups, входящей в ядро Linux, начиная с версии 2.6.29. Ее нельзя рассматривать как законченный продукт, фактически это набор из нескольких совершенно самостоятельных функций ядра Linux и пользовательских утилит, которые позволяют удобно создавать и управлять изолированными контейнерами [9]. Практически вся функциональность LXC представлена известными механизмами ядра cgroups и namespaces.

CGroups (Control Groups) — позволяет ограничить аппаратные ресурсы некоторого набора процессов. Под аппаратными ресурсами подразумеваются: процессорное время, память, дисковая и сетевая подсистемы. Набор или группа процессов могут быть определены различными критериями. Например, это может быть целая иерархия процессов, получающая все лимиты родительского процесса. Кроме этого возможен подсчет расходуемых группой ресурсов, заморозка (freezing) групп, создание контрольных точек (checkpointing) и их перезагрузка. Для управления этим полезным механизмом существует специальная библиотека libcgroups, в состав которой входят такие утилиты, как cgcreate, cgexec и некоторые другие.

Namespaces — пространства имен. Это механизм ядра, который позволяет изолировать процессы друг от друга. Изоляция может быть выполнена в шести контекстах (пространствах имен):

- `mount` — предоставляет процессам собственную иерархию файловой системы и изолирует ее от других таких же иерархий по аналогии с `chroot`;
- `PID` — изолирует идентификаторы процессов (PID) одного пространства имен от процессов с такими же идентификаторами другого пространства;
- `network` — предоставляет отдельным процессам логически изолированный от других стек сетевых протоколов, сетевой интерфейс, IP-адрес, таблицу маршрутизации, ARP и прочие реквизиты;
- `IPC` — обеспечивает разделяемую память и взаимодействие между процессами;
- `UTS` — изоляция идентификаторов узла, таких как имя хоста (`hostname`) и домена (`domain`);
- `user` — позволяет иметь один и тот же набор пользователей и групп в рамках разных пространств имен, в каждом контейнере могут быть свой `root` и любые другие пользователи и группы.

Одно из главных преимуществ LXC — это присутствие его базовых блоков (`cgroups` и `namespaces`) во всех современных ядрах Linux. Это означает, что нет необходимости что-то компилировать или использовать стороннее ядро, как в случае с OpenVZ. Единственное, что необходимо установить, это пакет утилит управления (`vzctl`, `Docker`, `libvirt`, `systemd`).

К системам управления можно отнести `Docker`.

`Docker` — это программное обеспечение для автоматизации развертывания и управления приложениями в среде виртуализации на уровне операционной системы, например LXC. `Docker` позволяет «упаковать» приложение и все его окружение и зависимости в контейнер, который легко переносится на любую Linux-систему с поддержкой `cgroups` в ядре, а также предоставляет среду по управлению контейнерами.

Идея `Docker` состоит в том, что производитель сам собирает программу со всеми необходимыми зависимостями и предоставляет ее в виде контейнера,

работающего на платформе Docker, все что останется, это скачать контейнер и запустить его, а впоследствии обновлять.

Для экономии дискового пространства используется файловая система (ФС) Aufs с поддержкой каскадно-объединенного монтирования. Контейнеры используют образ базовой ОС, а изменения записываются в отдельную область. Также поддерживается размещение контейнеров в файловой системе Btrfs, в котором включен режим копирования при записи. В состав Docker входит демон, являющийся сервером контейнеров, клиентские средства, позволяющие из интерфейса командной строки управлять образами и контейнерами, а также программный интерфейс, позволяющий в стиле REST программно управлять контейнерами. Демон обеспечивает полную изоляцию запускаемых на узле контейнеров на уровне ФС (каждому контейнеру доступна собственная файловая система), на уровне процессов (процессы не имеют доступа к чужой файловой системе контейнера, а ресурсы разделены средствами LXC), на уровне сети (каждому контейнеру привязано сетевое пространство имен и соответствующие виртуальные сетевые интерфейсы).

Набор средств клиента позволяет запускать, приостанавливать и возобновлять процессы в новых контейнерах, останавливать и запускать контейнеры. Серия команд позволяет осуществлять мониторинг запущенных процессов (по аналогии с ps, top). Новые образы создаются из специального сценарного файла (dockerfile), в котором возможно записать все изменения, сделанные в контейнере в новый образ. Все команды могут работать как с docker-демоном локальной системы, так и с сервером docker, доступным по сети. В интерфейсе командной строки встроены возможности по взаимодействию с публичным репозиторием, именуемым Docker Hub, в котором размещены образы контейнеров, предварительно собранные пользователями, образы можно скачивать в локальную систему, также возможна отправка локально собранных образов в Docker Hub.

Виртуальная инфраструктура является частным облаком, размещаемом на оборудовании провайдера. Инфраструктура представляет собой динамическое распределение ресурсов в соответствии с нормами предприятия. Вирту-

альная машина использует ресурсы одного компьютера, а виртуальная инфраструктура — всей информационной среды, формируя из компьютеров, а также из подключенных к ним сетей и хранилищ единый пул ресурсов (кластер).

Виртуальная инфраструктура включает в себя следующие компоненты [10]:

- гипервизоры для одного узла для полной виртуализации каждого компьютера;
- пакет услуг инфраструктуры распределенных систем на основе виртуализации (например, управление ресурсами) для оптимального распределения доступных ресурсов между виртуальными машинами;
- решения для автоматизации, обеспечивающие особые возможности оптимизации того или иного ИТ-процесса (например инициализации или восстановления в критических ситуациях).

Благодаря отделению программной среды от исходной аппаратной инфраструктуры, виртуализация позволяет объединить ряд серверов, инфраструктур хранения данных и сетей в единый пул ресурсов, который динамически, безопасно и надежно распределяется между приложениями по мере необходимости. С помощью этого решения организации могут создать вычислительную инфраструктуру с максимальной гибкостью, эффективностью, доступностью и автоматизацией, состоящую из недорогих серверов и соответствующих отраслевому стандарту.

Предпочтение виртуальной инфраструктуре отдают по причинам:

- экономии на обслуживающем персонале, при условии сохранения полной отказоустойчивости системы;
- отсутствии необходимости выделять бюджет на обновление оборудования;
- возможности объединения в общую виртуальную среду целые офисы, которые географически находятся в разных местах мира;
- возможности быстрого масштабирования проекта;
- быстрого доступа к данным.

Под каждый тарифный план создается изолированное частное облако с фиксированным количеством выделенных ему ресурсов. Выделенное частное облако становится гибкой виртуальной оболочкой и функционирующие внутри нее виртуальные машины могут объединяться в виртуальные сети. Изменение их вычислительной мощности происходит в зависимости от решаемых задач, а смена или преобразование тарифных планов производится в режиме реального времени без перебоев в работе.

Управление ресурсами осуществляется через удобный для пользователя веб-интерфейс, что позволяет делать все необходимые операции с виртуальными машинами, такие как:

- создание арендуемых виртуальных серверов самостоятельно;
- изменение конфигурации за несколько минут, часть операций даже без остановки и перезагрузки сервера (для некоторых операционных систем);
- включение, выключение, установка, переустановка ОС и приложений самостоятельно и удаленно;
- осуществление резервного копирования и сохранение состояний (снапшотов) работающих виртуальных машин.

3 СИСТЕМНЫЙ АНАЛИЗ ВИРТУАЛЬНОЙ ИНФРАСТРУКТУРЫ

На данный момент при анализе и синтезе сложных программных и аппаратных систем все чаще используется системный подход. Важным моментом для системного подхода является определение структуры системы — совокупности связей между элементами системы, отражающих их взаимодействие.

Принципы системного анализа — это положения общего характера, являющиеся обобщением опыта работы человека со сложными системами. Пренебрежение принципами при проектировании любой нетривиальной технической системы, непременно приводит к потерям того или иного характера, от увеличения затрат в процессе проектирования до снижения качества и эффективности конечного продукта.

Системный анализ выполнен в соответствии с [11].

3.1 Принцип конечной цели

Принцип конечной цели — основополагающий принцип системного анализа. Конечная цель имеет абсолютный приоритет, в системе все должно быть подчинено достижению конечной цели. Принцип имеет несколько правил:

- для проведения системного анализа необходимо в первую очередь, сформулировать цель функционирования системы, так как расплывчатые, не полностью определенные цели влекут за собой неверные выводы;
- анализ системы следует вести на базе первоочередного уяснения основной цели исследуемой системы, что позволит определить ее существенные свойства, показатели качества и критерии оценки;
- при синтезе систем любая попытка изменения или совершенствования должна быть в первую очередь рассмотрена с позиции его полезности в достижении конечной цели;
- цель функционирования искусственной системы задается, как правило, системой, в которой исследуемая система является составной частью.

В соответствии с данным принципом должна быть четко сформулирована конечная цель — назначение проектируемой системы и сформирован список функций, которые должна выполнять система.

Цель проектирования — разработка виртуальной инфраструктуры для реализации облачных услуг. Список функций проектируемой системы:

- Ф1 — прием и регистрация обращений пользователей;
- Ф2 — идентификация и обработка инцидентов и запросов на обслуживание;
- Ф3 — создание, смена, обновление и удаление услуг по требованию;
- Ф4 — предоставление доступа к услугам;
- Ф5 — мониторинг состояния инфраструктуры.

3.2 Принцип единства

Принцип единства — это совместное рассмотрение системы как целого и как совокупности частей. Принцип ориентирован на декомпозицию с сохранением целостных представлений о системе. Система должна рассматриваться и как целое, и как совокупность элементов. Расчленение системы необходимо производить, сохраняя целостное представление о системе. Принцип подразумевает выделение подсистем, композиция которых в совокупности со связями позволяет выполнять все функции проектируемой системы, определяет ее структуру и может быть рассмотрена как единая, целостная система.

На основании функций проектируемой системы, представленных выше, в ней можно выделить следующие подсистемы:

1. подсистема взаимодействия с пользователем;
2. подсистема управления услугами;
3. подсистема управления инфраструктурой;
4. подсистема защиты и обеспечения целостности данных.

На рис. 3.1 представлена схема взаимодействия между подсистемами.

Обозначения, приведенные на рис. 3.1 требуют пояснения:

- a — информация, предоставляемая пользователем, передается на сервер;
- b — выходная информация (результат выполнения);

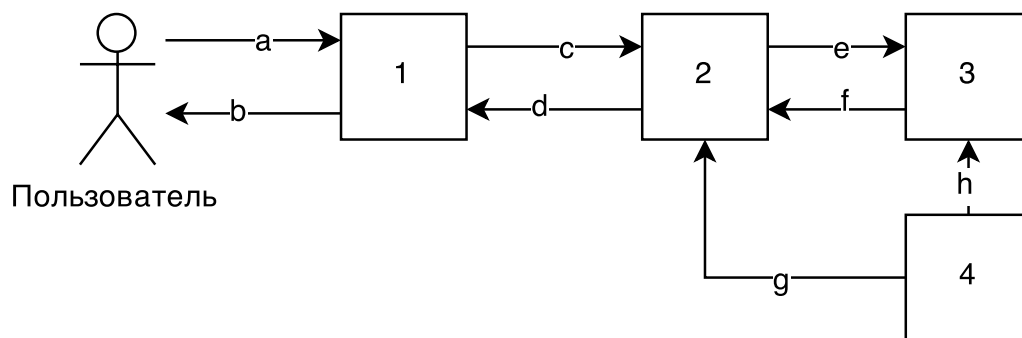


Рис. 3.1 – Взаимодействие между подсистемами и их связь с окружающей средой

c — проверка корректности переданных данных;

d — в случае неправильно введенных данных, возвращается управление к подсистеме взаимодействия с пользователем;

e — создание услуги;

f — возврат информации о созданной услуге;

g — обеспечение целостности информации об услуге;

h — обеспечение целостности данных пользователя.

3.3 Принцип связности

Любая часть системы должна рассматриваться со всеми своими связями с окружающими ее объектами, как внешними по отношению ко всей системе, так и внутренними — другими элементами системы. Если некоторая подсистема имеет связи только с внешней средой, то есть смысл реализовать ее в виде отдельной системы. Подсистема, не связанная ни с внешней средой, ни с другой подсистемой, является избыточной и должна быть удалена из системы.

Подсистема взаимодействия с пользователем представлена на рис. 3.2.

Подсистема управления услугами представлена на рис. 3.3.

Подсистема управления инфраструктурой представлена на рис. 3.4.

Подсистема защиты и обеспечения целостности данных представлена на рис. 3.5.

3.4 Принцип модульности

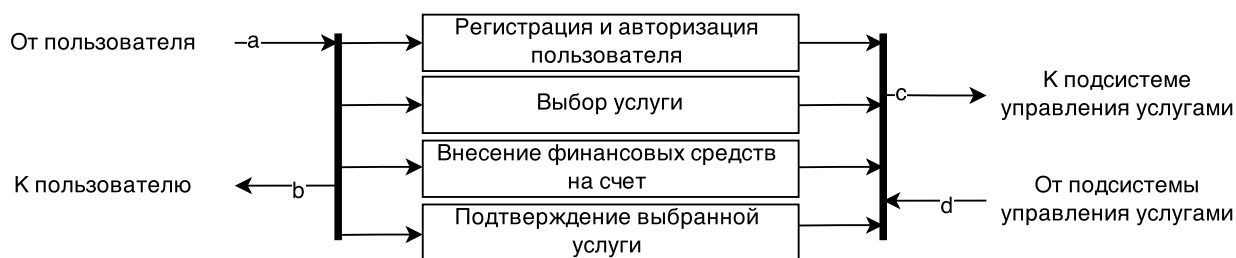


Рис. 3.2 – Внутренние и внешние связи подсистемы взаимодействия с пользователем

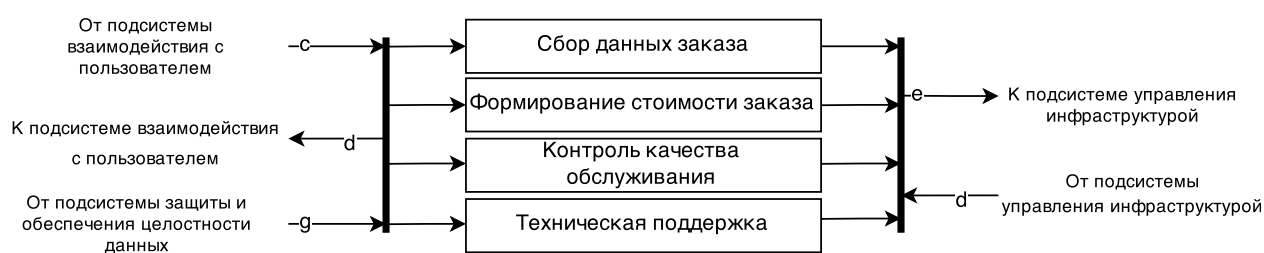


Рис. 3.3 – Внутренние и внешние связи подсистемы управления услугами



Рис. 3.4 – Внутренние и внешние связи подсистемы управления инфраструктурой

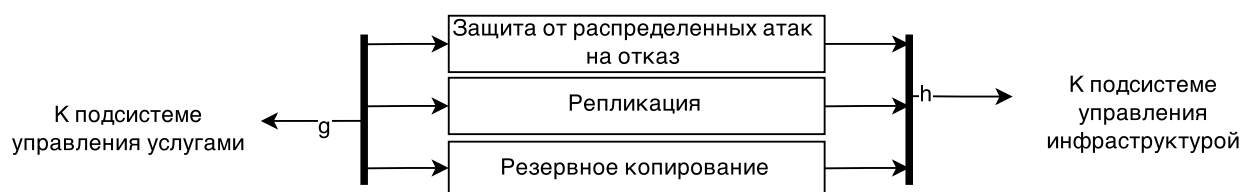


Рис. 3.5 – Внутренние и внешние связи подсистемы защиты и обеспечения целостности данных

В соответствии с этим принципом, выделение модулей системы, если таковые имеются, продуктивно и оправдано. Под модулями здесь понимаются относительно автономные и достаточно простые блоки, выполняющие ограниченный набор функций. Модуль в отличие от подсистем, которые имеют нерегулярную структуру и, как правило, несут определенную функцию, частично отражающую функцию системы, имеет регулярную структуру и характерные для него внутренние и внешние связи. Принцип указывает на возможность вместо части системы исследовать совокупность ее входных и выходных воздействий. Выделению модулей соответствует декомпозиция сложной задачи на множество более простых подзадач.

Принцип модульности для разрабатываемой системы поясняется с помощью рис. 3.6, описывающего разбиение на модули системы взаимодействия с пользователем.

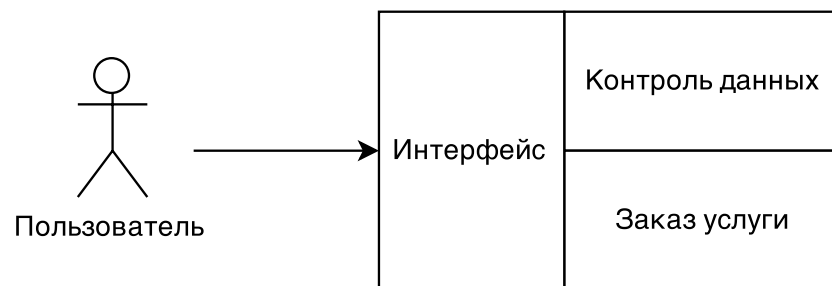


Рис. 3.6 – Принцип модульности на примере подсистемы взаимодействия с пользователем

Излишняя детализация не требуется, поэтому остальные системы на модули принято решение не разбивать.

3.5 Принцип функциональности

Принцип определяет первичность функции по отношению к структуре, так же, как цель первична для функции. Другими словами, цель определяет функции системы, а функции определяют ее структуру — совокупность элементов с их связями. Структура же, в свою очередь, определяет параметры системы. В случае придания системе новых функций полезно пересматривать ее структуру, а не пытаться втиснуть новую функцию в старую схему. Поскольку

выполняемые функции составляют процессы, то целесообразно рассматривать отдельно процессы, функции, структуры.

Функции подсистем приведены в пункте 3.1.

Матрица инцидентий функций системы и функций назначения подсистем приведена в таблице 3.1.

Таблица 3.1 – Матрица инцидентий

Функции	Подсистемы				Виртуальная инфраструктура
	1	2	3	4	
Ф1	+				+
Ф2	+				+
Ф3		+			+
Ф4		+			+
Ф5			+	+	+

В матрице инцидентий знаком «+» обозначены функции, которые реализуются для каждой из подсистем.

Детализация функций подсистемы на примере подсистемы взаимодействия с пользователем:

1. регистрация и авторизация пользователя в платежной системе;
2. обеспечение перечня услуг;
3. обеспечение возможных способов оплаты услуг;
4. возможность уточнения или изменения услуги.

Входными данными для подсистемы является информация о пользователе, а выходными — подтвержденный заказ услуг.

3.6 Принцип иерархии

Полезно введение иерархии частей и их ранжирование, что упрощает разработку системы и устанавливает порядок рассмотрения частей. Разработка иерархий классов является нетривиальной задачей. Грамотно спроектированные иерархии классов позволяют создавать высокоэффективные системы, пло-

хотел спроектированная иерархия приводит к созданию сложных и запутанных систем.

Выполнение принципа иерархичности для разрабатываемой системы на примере подсистемы защиты и обеспечения целостности данных проиллюстрировано на рис. 3.7.

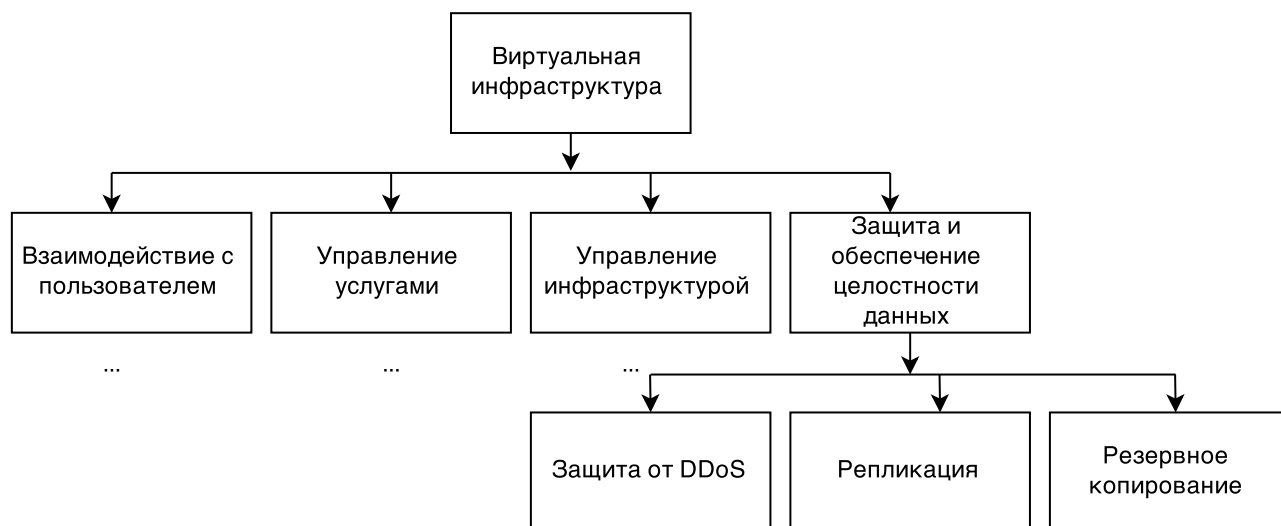


Рис. 3.7 – Принцип иерархии на примере подсистемы защиты и обеспечения целостности данных

3.7 Принцип сочетания централизации и децентрализации

Степень централизации должна быть минимальной, обеспечивающей выполнение поставленной цели. Соотношение централизации и децентрализации определяется уровнями, на которых вырабатываются и принимаются управленческие решения.

Недостатком децентрализованного управления является увеличение времени адаптации системы, которое существенно влияет на функционирование системы в быстро меняющихся средах. То, что в централизованных системах можно сделать за короткий промежуток времени, в децентрализованной системе будет осуществляться весьма медленно. Недостатком централизованного управления является сложность управления из-за огромного потока информации, подлежащей переработке в старшей системе управления. В медленно меняющейся обстановке децентрализованная часть системы успешно справ-

ляется с адаптацией поведения системы к среде и с достижением глобальной цели системы за счет оперативного управления, а при резких изменениях среды осуществляется централизованное управление по переводу системы в новое состояние.

Например, можно выполнить декомпозицию подсистемы взаимодействия с пользователем таким образом:

1. подсистема работы с пользователями;
2. подсистема работы пользователей с услугами;
3. подсистема учета финансовых средств.

Такое разбиение позволит реализовать полученные подмножества в виде отдельных модулей.

3.8 Принцип развития

Учет изменяемости системы, ее способности к развитию, адаптации, расширению, замене частей, накоплению информации. В основу синтезируемой системы требуется закладывать возможность развития, наращивания усовершенствования. Обычно расширение функций предусматривается за счет обеспечения возможности включения новых модулей, совместимых с уже имеющимися. С другой стороны, при анализе принцип развития ориентирует на необходимость учета предыстории развития системы и тенденций, имеющих в настоящее время, для вскрытия закономерностей ее функционирования.

Одним из способов учета этого принципа разработчиками является рассмотрение системы относительно ее жизненного цикла. Условными фазами жизненного цикла являются: проектирование, изготовление, ввод в эксплуатацию, эксплуатация, наращивание возможностей (модернизация), вывод из эксплуатации (замена), уничтожение.

Отдельные авторы этот принцип называют принципом изменения (историчности) или открытости. Для того чтобы система функционировала, она должна изменяться, взаимодействовать со средой.

Проектируемая система может быть развита следующим образом:

- добавлением модуля конвертации валюты для оплаты услуги;

- расширением перечня услуг;
- внедрением дополнительных технологий виртуализации;
- увеличением штата технической поддержки;
- введением расширенного мониторинга;
- многоуровневой защитой от атак на отказ;
- расширением памяти на системах хранения данных;
- многоуровневой репликацией и резервным копированием.

3.9 Принцип учета случайностей

Можно иметь дело с системой, в которой структура, функционирование или внешние воздействия не полностью определены.

Сложные системы не всегда подчиняются вероятностным законам. В таких системах можно оценивать «наихудшие» ситуации и проводить рассмотрение для них. Этот способ обычно называют методом гарантируемого результата, он применим, когда неопределенность не описывается аппаратом теории вероятностей.

При наличии информации о вероятностных характеристиках случайностей можно определять вероятностные характеристики выходов в системе.

События и действия, некорректные с точки зрения правил функционирования системы:

- попытка заказа услуги без наличия финансовых средств на счету;
- неподтвержденные заказы услуг;
- сбой в работе аппаратуры, в частности устройств хранения данных;
- сбой сети в пределах дата-центра или на магистрали;
- отсутствие своевременной работы технической поддержки;
- ошибки и уязвимости в используемых программных платформах.

Кроме того, необходимо вести контроль успешности и целостности проведения операций с компонентами системы, данными пользователей и корректно обрабатывать исключения, возникновение которых возможно в процессе работы системы.

Перечисленные выше принципы обладают высокой степенью общности. Такая интерпретация может привести к обоснованному выводу о незначимости какого-либо принципа. Однако, знание и учет принципов позволяет лучше увидеть существенные стороны решаемой проблемы, учесть весь комплекс взаимосвязей, обеспечить системную интеграцию.

4 ОПИСАНИЕ ВИРТУАЛЬНОЙ ИНФРАСТРУКТУРЫ

В разделе описания виртуальной инфраструктуры представлена общая схема инфраструктуры, схемы мониторинга, репликации и балансировки DNS, используемые технологии виртуализации, алгоритмы взаимодействия пользователя и компонентов инфраструктуры.

4.1 Общая схема инфраструктуры

Общая схема инфраструктуры представлена на рис. 4.1 и чертеже СевГУ 09.03.01.15.A1.

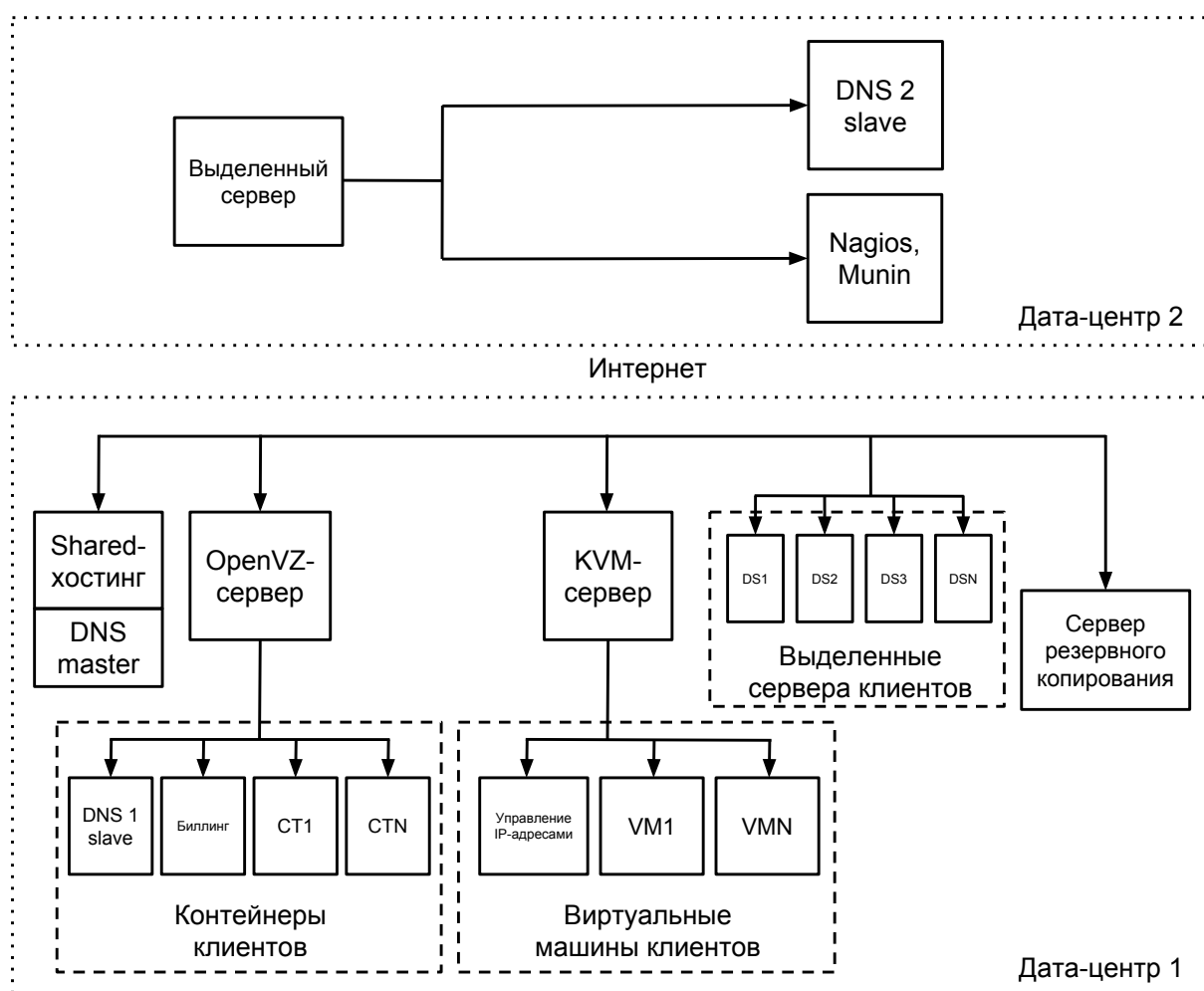


Рис. 4.1 – Общая схема инфраструктуры

В дата-центре 1 располагается основная часть инфраструктуры: сервера shared-хостинга, виртуализации OpenVZ и KVM, сервер резервного копирования и выделенные сервера клиентов.

В дата-центре 2 на двух арендованных виртуальных машинах находится один из подчиненных DNS-серверов, а также сервер мониторинга.

На физических серверах располагаются важные элементы инфраструктуры: главный и один подчиненный DNS-сервера, система биллинга и система управления IP-адресами.

4.2 Схема мониторинга

Схема мониторинга инфраструктуры представлена на рис. 4.2.

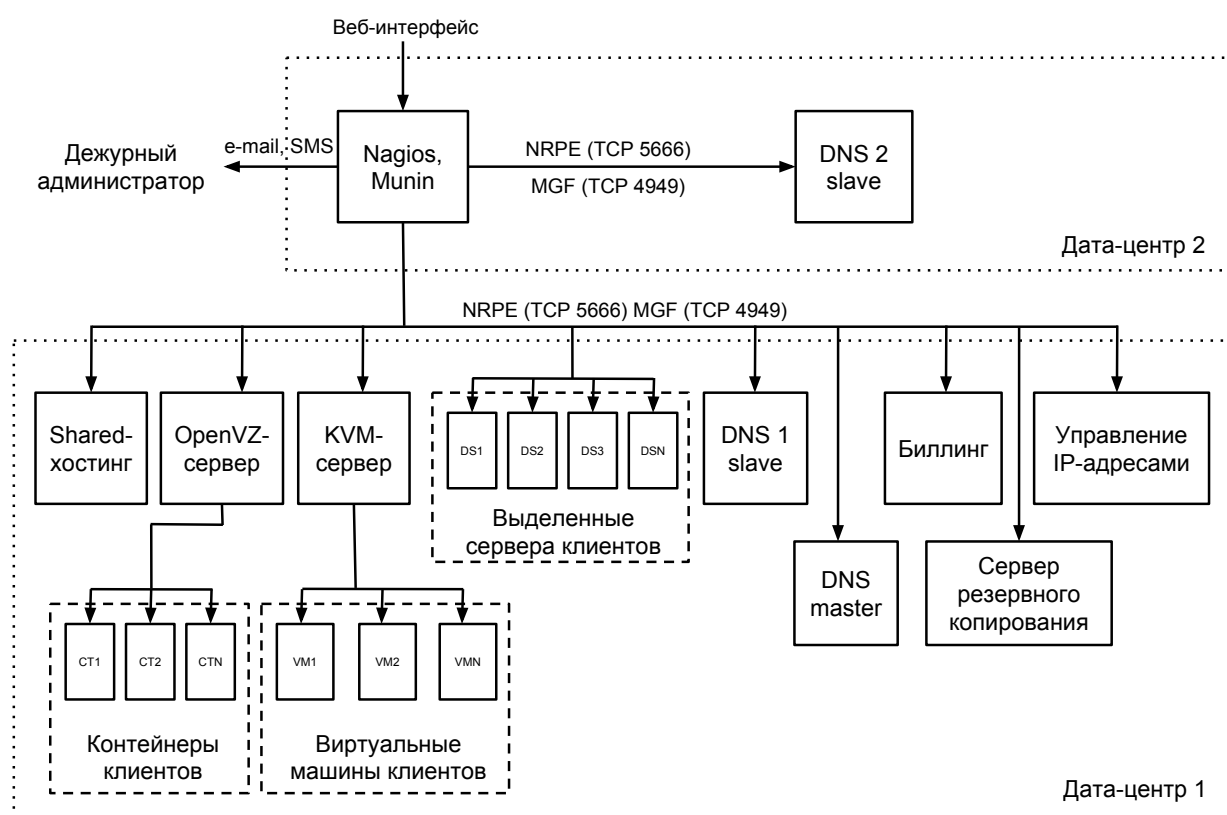


Рис. 4.2 – Схема мониторинга инфраструктуры

В случае возникновения проблемы на одном из серверов, Nagios фиксирует изменения и отправляет текстовые уведомления по SMS и электронной почте дежурному администратору, который следит за системой мониторинга. Nagios работает по протоколу NRPE (Nagios Remote Plugin Executor) и слушает порт 5666.

Munin имеет возможность визуализировать состояние ресурсов с помощью графиков, наблюдая за графиками можно делать выводы о событиях, происходящих в инфраструктуре. Munin работает по протоколу MGF (Munin Graphing Framework) на порту 4949.

Помимо физических серверов, мониторятся также виртуальные машины, на которых располагаются сервисы инфраструктуры, а также некоторые виртуальные машины клиентов, по требованию.

4.3 Схема репликации и балансировки нагрузки DNS-серверов

С целью обеспечения отказоустойчивости системы была разработана следующая схема репликации и балансировки нагрузки DNS-серверов. Схема представлена на рис. 4.3.

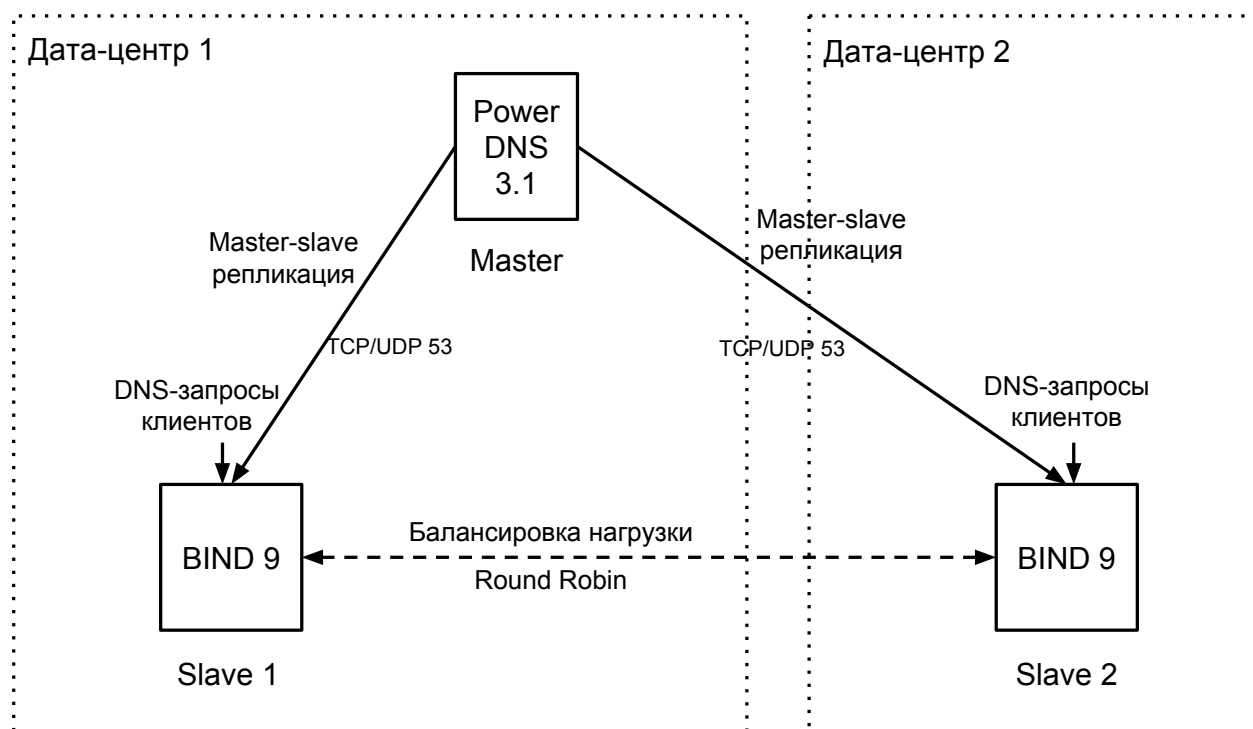


Рис. 4.3 – Схема репликации и балансировки нагрузки DNS-серверов

В каждой зоне есть один главный сервер имен на котором хранится официальная копия данных зоны. Администратор модифицирует информацию, касающуюся зоны, редактируя файлы главного сервера [12]. Подчиненный (slave) сервер копирует свои данные с главного сервера посредством операции,

называемой передачей зоны (zone transfer). В зоне имеется два подчиненных сервера, один из которых располагается в другом дата-центре.

На главном сервере установлен DNS-сервер PowerDNS. На подчиненных серверах — BIND 9.

Балансировка нагрузки осуществляется по алгоритму Round Robin (алгоритм кругового обслуживания). Алгоритм представляет собой перебор по круговому циклу: первый запрос передается одному серверу, затем следующий запрос передается другому и так до достижения последнего сервера [13].

4.4 Используемые технологии виртуализации

На основе обзора литературы, проведенного в пункте 2, выбраны и внедрены технологии виртуализации OpenVZ (для организации контейнеров) и KVM (для организации виртуальных машин).

4.5 Алгоритмы функционирования инфраструктуры

Взаимодействие пользователя с инфраструктурой происходит по сценариям, представленным в пункте 4.5.1.

В пункте 4.5.2 представлены сценарии взаимодействия компонентов инфраструктуры.

4.5.1 Схема взаимодействия пользователя с инфраструктурой

Сценарий 1. Заказ облачного сервиса:

1. пользователь обращается к инфраструктуре через веб-интерфейс с целью заказа услуги;
2. биллинг регистрирует нового пользователя;
3. заключается договор на предоставление услуги с указанием реквизитов пользователя;
4. подсистема управления услугами производит выделение физического либо виртуального сервера (контейнера, виртуальной машины), либо иной услуги с требуемыми характеристиками;
5. переход к сценарию 2.

Сценарий 2. Использование сервиса:

1. клиент пользуется предоставленным сервисом некоторый период времени, используя веб-интерфейс (VNC, SSH);
2. биллинг выставляет счет пользователю за оказанные услуги;
3. пользователь оплачивает сервис;
4. переход к пункту а, пока не наступит событие, соответствующее сценариям 3, 4, 5, 6 или 7.

Сценарий 3. Изменение условий предоставления сервиса:

1. пользователь желает изменить условия предоставления сервиса, либо отказаться от него;
2. биллинг регистрирует запрос;
3. подсистема управления услугами производит изменение условий предоставления сервиса или удаляет его;
4. переход к сценарию 2, в случае продолжения оказания сервиса, иначе к сценарию 5.

Сценарий 4. Окончание срока действия договора:

1. у пользователя окончился срок действия договора;
2. биллинг блокирует услуги;
3. если пользователь желает продлить срок действия договора, переход к сценарию 6, иначе к сценарию 5.

Сценарий 5. Окончание использования облачной услуги:

1. пользователь желает завершить использование сервиса;
2. биллинг регистрирует отмену использования сервиса;
3. подсистема управления услугами удаляет пользовательскую услугу (выделенный сервер, контейнер, виртуальная машина) из списка используемых.

Сценарий 6. Продление договора:

1. пользователь обращается к инфраструктуре через веб-интерфейс с целью продления срока действия договора;
2. биллинг регистрирует факт заключения нового договора;
3. переход к сценарию 2.

Сценарий 7. Возникновение проблем в работе услуги:

1. пользователь обращается к биллингу через веб-интерфейс с целью предоставления заявки на устранение проблемы и информации о возникшей проблеме;
2. биллинг регистрирует запрос и направляет его технической поддержке;
3. техническая поддержка первого уровня устраняет проблему, если нет, то заявка передается в отдел системного администрирования;
4. переход к сценарию 2.

Алгоритмы взаимодействия пользователем с инфраструктурой представлены на чертеже СевГУ 09.03.01.15.A2.

4.5.2 Алгоритм взаимодействия компонентов инфраструктуры

Сценарий 1. Мониторинг:

1. Nagios мониторит состояние сервисов;
2. Nagios обнаружил недоступность или сбой сервиса;
3. отправка электронного и SMS-сообщения дежурному администратору;
4. дежурный администратор исправляет проблему;
5. переход к пункту а.

Сценарий 2. Резервное копирование:

1. запуск задания инкрементального резервного копирования;
2. расчет свободного дискового пространства на удаленном хранилище для инкрементальной или полной копии;
3. в случае нехватки дискового пространства на удаленном хранилище — отправка электронного сообщения системному администратору, переход к пункту е;
4. архивация и сжатие данных;
5. передача данных по сети;
6. конец резервного копирования.

Сценарий 3. Распределение IP-адресов:

1. добавлен новый физический сервер, контейнер, виртуальная машина или клиент хостинга заказал дополнительный адрес;

2. проверка свободного IP-адреса;
3. если свободного адреса нет, то уведомление о необходимости заказа новой подсети IP-адресов у провайдера, переход к пункту е;
4. привязка нового IP-адреса к серверу;
5. удаление привязанного адреса из списка свободных;
6. конец.

Сценарий 4. Добавление нового сервера:

1. заказ услуги физического сервера, контейнера или виртуальной машины;
2. выделение IP-адресов;
3. подключение к системе мониторинга;
4. трансфер или создание зоны на главном DNS-сервере;
5. подключение к системе резервного копирования;
6. конец.

Сценарий 5. Добавление нового домена:

1. трансфер или создание зоны на главном DNS-сервере;
2. проверка доступности подчиненных DNS-серверов;
3. репликация зоны на подчиненные сервера;
4. ожидание обновления зон провайдера;
5. проверка работоспособности DNS;
6. конец.

Алгоритм взаимодействия компонентов инфраструктуры представлен на чертеже СевГУ 09.03.01.15.А3.

5 РУКОВОДСТВО АДМИНИСТРАТОРА

Руководство администратора содержит информацию о сетевой инфраструктуре, используемых технологиях виртуализации и программном обеспечении.

5.1 Расположение серверов в сетевой инфраструктуре

Во всей инфраструктуре на физических серверах первого дата-центра располагаются:

- shared-хостинг (10.0.0.100);
- виртуализация OpenVZ (10.0.1.100);
- виртуализация KVM (10.0.2.100);
- сервер резервного копирования (10.0.3.100);
- физические сервера клиентов (10.0.4.100 — 10.0.4.200).

В свою очередь, некоторые важные сервисы инфраструктуры располагаются на виртуальных машинах:

- сервер биллинга (10.0.1.101), OpenVZ;
- сервер управления IP-адресами (10.0.2.101), KVM;
- сервер DNS 1 (10.0.2.102), KVM.

Сервер мониторинга и сервер DNS 2 располагаются на виртуальных машинах KVM в другом дата-центре. Сервер shared-хостинга выполняет роль главного (master) DNS-сервера и управляется с помощью PowerDNS.

5.2 Сервер shared-хостинга

Сервер shared-хостинга является самым уязвимым местом всей инфраструктуры из-за большого количества и плотности клиентов на сервере. Распределение ресурсов между пользователями происходит за счет встроенных в ядро Linux механизмов.

Стоит отметить, что после переезда на новый сервер для хранения данных доступен RAID-массив из SSD дисков объемом 1 Тб.

На сервере установлен и настроен веб-сервер в составе следующего ПО:

- http-сервер Apache HTTP Server, кэширующий и проксирующий http-сервер nginx и обработчик динамических PHP-запросов php-fpm;
- memcached кэширует динамические запросы в оперативной памяти;
- MySQL и PostgreSQL выполняют роли серверов баз данных.

nginx принимает запросы к серверу с порта 80 и решает каким образом обрабатывать запрос. Если это запрос на получение статического файла (изображение, js-код, CSS, видео, аудио, архивы), то nginx сам его обрабатывает и в случае необходимости кэширует. Если же запрашивается динамическое содержимое, то nginx передает запрос на бэкенд, где его, в зависимости от настройки виртуального хоста сайта, обрабатывает либо php-fpm, либо встроенный в Apache модуль mod_php. nginx не может кэшировать динамические запросы, поэтому для этого используется memcached, который работает в связке с Apache и позволяет кэшировать динамические запросы. Для работы Apache с memcached необходима установка модуля PHP php5-memcached:

```
# apt-get install php5-memcached
```

Почтовой связкой на сервере shared-хостинга выступает SMTP-сервер Exim Internet Mailer, а запросы по протоколам POP3 и IMAP принимает Dovecot. В качестве антиспам-связки выступает Postgrey, Spamassassin, ClamAV и OpenDKIM. Конфигурации данных сервисов являются стандартными.

ProFTPD необходим для работы пользователей по протоколу FTP. Доступна возможность подключения пользователей по протоколу FTPS (порт 22).

Сервер управления временем NTP обращается к следующим серверам за уточнением времени на сервере:

```
# cat /etc/ntp.conf | grep ^server | awk '{print $2}'
0.debian.pool.ntp.org
1.debian.pool.ntp.org
2.debian.pool.ntp.org
3.debian.pool.ntp.org
```

Конфигурация SSH требует отдельного пояснения.

Доступ к SSH по стандартному порту закрыт, это позволяет избавиться от большинства злоумышленников, которые пытаются скомпрометировать па-

роль доступа к серверу. Также запрещен вход от пользователя root, для этого создан отдельный пользователь. Можно ограничить доступ к серверу по SSH, оставив только доверенные адреса или подсети адресов.

```
# useradd sshuser
# passwd sshuser

# cat /etc/ssh/sshd_config
Port 222
PermitRootLogin yes
Match host 10.0.0.111, 10.0.0.222
    PermitRootLogin yes

# service ssh restart
```

Подключение к серверу по SSH и получение root-прав:

```
user@10.0.0.111~$ ssh -p 222 sshuser@10.0.0.100
sshuser@10.0.0.100's password:
sshuser@10.0.0.100~$ su -
Password:
root@10.0.0.100~#
```

Разрешен вход на сервер по ключам:

```
user@10.0.0.111~$ ssh-keygen
user@10.0.0.111~$ ssh-keygen -p
user@10.0.0.111~$ ssh-copy-id -p 222 root@10.0.0.100
user@10.0.0.111~$ ssh -p 222 root@10.0.0.100
```

Резервное копирование данных пользователей осуществляется средствами панели ISPmanager 5 в ночное время, применяется инкрементальный метод резервного копирования. В инкрементальном методе копирования при первом резервном копировании будет создана полная копия, в которой будут сохранены все файлы. При последующих будут сохраняться файлы, измененные с момента предыдущей полной или инкрементальной копии. Для восстановления необходимо несколько копий — полная и все инкрементальные, сохраненные до выбранной точки восстановления. Раз в неделю (в воскресенье) запускается очередная полная резервная копия данных.

На сервере функционирует скрипт блокировки адресов, которые слишком часто подбирают пароли доступа к панели администратора наиболее популярных CMS (система управления содержимым), таких как WordPress и Joomla. Скрипт написан на языке Shell, находится в открытом доступе и имеет простое использование:

```
#!/bin/bash
# Nginx block ip more than 3k connections to WordPress \
and Joomla admin panels
# admin@amet13.name
# v0.1 29.06.2014
# v0.2 30.06.2014

# IP's > 3000 connections to WordPress and Joomla admin panels
command=$(cat /var/log/nginx/access.log | \
grep "administrator/index.php \|wp-login.php " | \
cut -f1 -d " " | \
sort | \
uniq -c | \
sort -n | \
awk '{ if ($1 > 2999) print $2}')

# Add new ip's to database
for word in $command; do
    # Is that ip already in database?
    grep $word /etc/nginx/blockips.conf > /dev/null
    if (( $? ));
    then
        sed -i "s/allow all;/deny $word;\nallow all;/" \
        /etc/nginx/blockips.conf
    fi
done
exit 0
```

Основные конфигурации настроек веб-сервера хранятся в файлах:

- /etc/nginx/nginx.conf;
- /etc/apache2/apache2.conf;

- /etc/php5/{apache2,cgi,cli,fpm}/php.ini;
- /etc/mysql/my.cnf;
- /etc/postgresql/9.1/main/postgresql.conf.

Для сканирования уязвимостей на сайтах пользователей используется утилита maldet, которая находит подозрительные файлы в системе и составляет отчет по найденным файлам.

Установка и пример использования maldet:

```
# cd /tmp/
# wget http://www.rfxn.com/downloads/maldetect-current.tar.gz
# tar xzf maldetect-current.tar.gz
# cd maldetect-*
# ./install.sh

# maldet -b --scan-all /var/www/
# maldet --report list
# maldet --report 021715-1414.3266
# maldet -q 021715-1414.3266
```

На сервере shared-хостинга на сайтах пользователей находится большое количество уязвимостей, которые часто используются для рассылки спама с сервера. Для обнаружения большого числа рассылок с сервера был написан плагин для системы мониторинга Nagios, который контролирует число писем в почтовой очереди, а также проверяет наиболее популярные организации (например Spamhaus), которые собирают данные о спам-серверах.

Для просмотра списка очереди используется команда `exim -bp` или `mailq`. Таким образом можно просмотреть количество писем для разных доменов и их количество:

```
# exim -bp | exiqsumm -c -s | head -10
Count   Volume   Oldest    Newest    Domain
-----
12151   1378KB    14h       0m       spamsite.ru > host.ru
    93     264KB    14h       5m       spamsite.ru > gmail.com
    16     154KB    13h      11h       site.ru > kk.com
     8      77KB    13h      11h       site.ru > ww.com
```

В данном случае очевидна рассылка писем с сайта spamsite.ru. Посмотреть идентификатор письма (ID), а также содержимое заголовков и тела письма можно теми же командами, но с другими параметрами:

```
# exiqgrep -b -f 'spamsite.ru' | head -1
1YyRgs-0000zD-Qy From: <fobos@spamsite.ru> To: maria@host.ru
# exim -Mvh 1YyRgs-0000zD-Qy
# exim -Mvb 1YyRgs-0000zD-Qy
```

Если сайт действительно заражен, то следует отключить его, оповестить пользователя о закрытии сайта и необходимости избавления от вредоносного кода. Также необходимо удалить спам-письма из очереди сообщений:

```
# exiqgrep -b -f 'spamsite.ru' | \
awk '{print $1}' | xargs exim -Mrm
```

В случае неработоспособности панели ISPmanager 5 возможна ее перезагрузка:

```
# pgrep -l core
4437 core
7129 core
21194 core
# killall core
```

5.3 Сервер виртуализации OpenVZ

Список всех существующих в системе контейнеров можно узнать командой `vzlist -a`:

```
# vzlist -a
```

CTID	NPROC	STATUS	IP_ADDR	HOSTNAME
145	134	running	10.0.0.245	site1.ru
146	41	running	10.0.0.246	site2.ru
205	—	stopped	10.0.0.205	site3.ru

В заголовках показывается ID контейнера, число процессов в контейнере, текущий статус, IP-адрес и имя контейнера.

Список всех доступных шаблонов операционной системы и список шаблонов установленных локально:

```
# vztmpl-dl --list-all
centos-5-x86
centos-5-x86_64
centos-5-x86_64-devel
centos-5-x86-devel
centos-6-x86
centos-6-x86_64
centos-6-x86_64-devel

# vztmpl-dl --list-local
centos-7-x86_64
debian-7.0-x86_64
ubuntu-14.04-x86_64
```

Каждый контейнер имеет свой конфигурационный файл, которые хранятся в каталоге `/etc/sysconfig/vz-scripts/`, именуются эти файлы по CTID контейнера. Например, для контейнера с CTID=101, файл будет называться `101.conf`.

При создании контейнера можно использовать типовую конфигурацию для VPS. Типовые файлы конфигураций находятся в том же каталоге:

```
# ls -l /etc/sysconfig/vz-scripts/ | grep sample
ve-basic.conf-sample
ve-ispbasic.conf-sample
ve-light.conf-sample
ve-vswap-1024m.conf-sample
ve-vswap-1g.conf-sample
ve-vswap-256m.conf-sample
ve-vswap-2g.conf-sample
ve-vswap-4g.conf-sample
ve-vswap-512m.conf-sample
```

В этих конфигурационных файлах описаны контрольные параметры ресурсов, выделенное дисковое пространство, оперативная память и прочие ресурсы. На базе уже существующего файла конфигурации создан типовый файл, подходящий для большинства контейнеров именуемый `ve-custom.conf-sample`. Таким образом, при использовании этого конфигурационного файла, будет создаваться контейнер, которому будет доступен 1 Гб выделенного дискового

пространства, 128 Мб оперативной памяти и 128 Мб SWAP. В дальнейшем, при создании контейнеров, следует использовать данный конфигурационный файл.

Создание контейнера осуществляется командой:

```
# vzctl create 101 --ostemplate debian-7.0-x86_64 --config custom
```

где 101 — CTID контейнера, --ostemplate — шаблон ОС, --config — используемый шаблон конфигурационного файла.

Перед первым запуском контейнера необходимо установить его IP адрес, hostname, указать DNS сервера и задать пароль суперпользователя. Для настройки VPS используется команда `vzctl set`.

Для того, чтобы контейнер запускался при старте хост-компьютера (например после перезагрузки), необходимо использовать команду:

```
# vzctl set 101 --onboot yes --save
```

При использовании ключа --save, сохраняются параметры контейнера в соответствующий ему конфигурационный файл.

Аналогично задается hostname и IP-адрес:

```
# vzctl set 101 --hostname site.ru --save
```

```
# vzctl set 101 --ipadd 10.0.0.210 --save
```

Адреса DNS серверов (в большинстве случаев адрес DNS совпадает с адресом хост-компьютера, поэтому можно вместо адреса указать параметр inherit):

```
# vzctl set 101 --nameserver inherit --save
```

Установка пароля суперпользователя:

```
# vzctl set 101 --userpasswd root:p@ssw0rd
```

Пароль будет установлен в VPS, в файл /etc/shadow и не будет сохранен в конфигурационный файл контейнера. Если же пароль будет утерян или забыт, то можно будет просто задать новый.

После настроек нового контейнера, его можно запустить:

```
# vzctl start 101
```

Проверка сетевых интерфейсов внутри гостевой ОС:

```
# vzctl exec 101 ifconfig | grep "lo\|venet" -A 1
lo          Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
--
venet0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00
            inet addr:127.0.0.2  P-t-P:127.0.0.2   \
            Bcast:0.0.0.0   Mask:255.255.255.255
--
venet0:0    Link encap:UNSPEC  HWaddr 00-00-00-00-00-00
            inet addr:10.0.0.210 P-t-P:10.0.0.210  \
            Bcast:10.0.0.201  Mask:255.255.255.255
```

Должны присутствовать сетевые интерфейсы:

- lo (127.0.0.1);
- venet0 (127.0.0.2);
- venet0:0 (10.0.0.210).

Если сеть в порядке, то можно соединиться к контейнеру по SSH с хост-компьютера:

```
# ssh root@10.0.0.210
root@10.0.0.210's password: p@ssw0rd
```

Вход в контейнер напрямую с хост-компьютера осуществляется командой `vzctl enter`:

```
# vzctl enter 101
entered into CT 101
root@site:/#
```

Для остановки контейнера используется команда `vzctl stop`. Для полной остановки контейнера, системе требуется немного времени. Иногда нужно выключить VPS как можно быстрее, например, если контейнер был подвержен взлому. Для того чтобы срочно выключить VPS, используется ключ `--fast`:

```
# vzctl stop 101 --fast
```

Для перезапуска контейнера можно использовать команду `vzctl restart`.

Для того чтобы удалить контейнер, его нужно сначала остановить:

```
# vzctl stop 101
```

```
# vzctl destroy 101
```

Команда выполняет удаление частной области сервера и переименовывает файл конфигурации, дописывая к нему `.destroyed`.

Иногда бывает нужно выполнить команду на нескольких VPS. Для этого можно использовать команду:

```
# for i in `vzlist -o veid -H`; do \  
echo "VPS $i"; vzctl exec $i command; done
```

Например можно узнать сколько времени работают все запущенные контейнеры:

```
# for i in `vzlist -o veid -H`; do \  
echo "VPS $i"; vzctl exec $i uptime; done  
VPS 101  
05:45:01 up 2 min, 0 users, load average: 0.01, 0.02, 0.03  
VPS 102  
05:46:01 up 1 min, 0 users, load average: 0.04, 0.05, 0.06
```

Если на хост-ноде наблюдается высокое значение Load Average, то в первую очередь следует посмотреть данные значения непосредственно у контейнеров:

```
# vzlist -o veid,laverage  
CTID      LAVERAGE  
145 0.00/0.01/0.05  
146 0.00/0.00/0.00  
148 0.60/0.51/0.34
```

В случае обнаружения нагружающего контейнера, стоит подключиться к нему и устранить причину нагрузки.

Непосредственно с хост-ноды можно узнать, какому контейнеру принадлежит процесс:

```
# vzpid 1048108  
Pid      CTID  Name  
1048108 145   php-cgi
```

Свободное место в контейнере можно узнать командой `df`:

```
# df -h
Filesystem                Size      Used Avail Use% Mounted on
/dev/mapper/lv_root        50G       5,6G   42G   12% /
tmpfs                     16G         0   16G    0% /dev/shm
/dev/cciss/c0d0p1         485M      276M   184M   61% /boot
/dev/mapper/lv_vz         489G      403G    62G   87% /vz

# df -i
Filesystem                Inodes     IUsed       IFree IUse% Mounted on
/dev/mapper/lv_root      3276800  55229   3221571     2% /
tmpfs                    4101290         1   4101289     1% /dev/shm
/dev/cciss/c0d0p1        128016         96   127920     1% /boot
/dev/mapper/lv_vz       32546816        248  32546568     1% /vz
```

Подключение модуля TUN для контейнера (необходимо для работы OpenVPN). Прежде чем запускать контейнер нужно убедиться, что модуль TUN загружен на хост-ноде:

```
# lsmod | grep tun
```

В случае, если модуль не загружен:

```
# modprobe tun
# lsmod | grep tun
tun          1957      0
```

Разрешаем использовать устройство TUN контейнеру:

```
# vzctl set 101 --devnodes net/tun:rw --save
# vzctl set 101 --devices c:10:200:rw --save
# vzctl set 101 --capability net_admin:on --save
```

Запуск контейнера:

```
# vzctl start 101
```

Создание в контейнере собственного устройства TUN:

```
# vzctl exec 101 mkdir -p /dev/net
# vzctl exec 101 mknod /dev/net/tun c 10 200
# vzctl exec 101 chmod 600 /dev/net/tun
```

Для пользователей в контейнерах доступен брандмауэр Netfilter, для его работы в файле `/etc/vz/vz.conf` должна присутствовать строка:

```
IPTABLES="ipt_owner ipt_REDIRECT ipt_recent ip_tables \  
iptables_filter iptable_mangle ipt_limit ipt_multiport \  
ipt_tos ipt_TOS ipt_REJECT ipt_TCPMSS ipt_tcpmss \  
ipt_ttl ipt_LOG ipt_length ip_conntrack ip_conntrack_ftp \  
ipt_state iptable_nat ip_nat_ftp"
```

Для того, чтобы для контейнеров был доступен FUSE, его необходимо включить на хост-ноде и проверить, что он успешно включен:

```
# modprobe fuse  
# lsmod | grep fuse  
fuse 92980 54
```

Также необходимо добавить автозагрузку модуля при перезапуске хост-ноды:

```
# vim /etc/rc.local  
#!/bin/sh  
modprobe fuse
```

Проброс FUSE для контейнера 101:

```
# vzctl stop 101  
# vzctl set 101 --devnodes fuse:rw --save  
# vzctl set 101 --devices c:10:229:rw --save  
# vzctl start 101
```

В контейнере проверяем, пробросилось ли устройство:

```
[root@site /]# ls /dev/fuse  
/dev/fuse
```

Резервное копирование контейнеров осуществляется утилитами `vzdump` и `vzrestore`:

```
# vzdump --suspend --compress --dumpdir /root/ \  
--exclude-path /tmp/ 101  
# vzrestore \  
/vz/dump/vzdump-openvz-101-2015_01_01-12_12_12.tar 103
```

Для работы OpenVZ с файлами используется `ploop`, так как `simfs` является устаревшим методом. Диски с файловыми системами контейнеров хранятся в `/vz/private` и имеют имя `root.hdd`.

Смена размера диска для контейнера:

```
# vzctl set 103 --diskspace 2G --save
```

5.4 Сервер виртуализации KVM

В целом, администрирование KVM-ноды не отличается от администрирования OpenVZ. Однако в случае OpenVZ возможно получить доступ к пользовательскому контейнеру непосредственно с хост-ноды, в случае с KVM это невозможно, это следует учесть при работе с виртуальной машиной. Изменение размера диска также является нетривиальной задачей, высок риск краха всей файловой системы без возможности восстановления данных.

Команды, которые чаще всего используются при администрировании виртуальных машин на базе KVM:

```
# virsh list --all
# virsh start lin_tmpl
# virsh shutdown lin_tmpl
# virsh destroy lin_tmpl
# virsh undefine lin_tmpl
# virsh autostart lin_tmpl
# virsh autostart --disable lin_tmpl
# virsh define /etc/libvirt/qemu/lin_tmpl.xml
# virsh vncdisplay lin_tmpl
# virsh dumpxml lin_tmpl | grep vnc
```

Для защиты сервера от брутфорса пароля SSH на серверах OpenVZ и KVM используется fail2ban.

Установка (в CentOS 6, для установки fail2ban нужно подключить репозиторий EPEL):

```
# rpm -ivh http://download.fedoraproject.org/pub/epel/6/$(arch) /
    epel-release-6-8.noarch.rpm
# yum install fail2ban
```

fail2ban блокирует с помощью IPTables на некоторое время активные адреса, которые пытаются соединиться по SSH после пяти неудачных попыток, также отправляет администратору письмо с уведомлением о блокировке:

```
# cat /etc/fail2ban/jail.local
[ssh-iptables]
enabled = true
filter = sshd
action = iptables[name=SSH, port=ssh, protocol=tcp]
sendmail-whois[name=SSH, dest=admin@domain.com, \
sender=fail2ban@example.com, sendername="Fail2Ban"]
maxretry = 5
bantime = 86400
```

Разблокировать IP адрес из бана:

```
# fail2ban-client set JAIL unbanip IP
```

Пример разблокировки:

```
# fail2ban-client set ssh-iptables unbanip 10.0.0.111
```

В виртуальных машинах пользователи зачастую склонны забывать пароль от MySQL. В данном случае существует простое решение проблемы. Остановка службы MySQL:

```
# service mysql stop
```

Запуск службы с опцией --skip-grant-tables:

```
# mysqld_safe --skip-grant-tables &
```

Подключение с серверу MySQL при помощи клиента:

```
# mysql -u root
mysql>
```

Ввод нового пароля для root:

```
mysql> use mysql;
mysql> update user set password=PASSWORD("NEW-PASS") \
where User='root';
mysql> flush privileges;
mysql> quit
```

Остановка сервера MySQL:

```
# service mysql stop
```

Запуск MySQL-сервера и вход с новым паролем:

```
# service mysql start
# mysql -u root -p
Enter password: NEW-PASS
```

5.5 Дополнительные виртуальные сервера

В качестве сервера мониторинга используется Nagios, для построения графиков — Munin. Дополнительной настройки этих сервисов не требуется. Добавление новых хостов происходит вручную, требуется лишь править адреса и сервисы, которые требуется мониторить.

Важно мониторить свободное место на жестком диске сервера резервного копирования и вовремя очищать старые резервные копии.

На DNS серверах используется master-slave репликация, причем один из DNS-серверов, равно как и сервер мониторинга расположены в другом дата-центре для обеспечения отказоустойчивости системы. Также между подчиненными серверами настроена балансировка нагрузки.

5.6 Защита от DDoS-атак

Стоит незамедлительно реагировать на DDoS-атаки. При обнаружении подозрительного трафика следует убедиться, что это действительно DDoS, а также поинтересоваться у дата-центра реакцию сетевого оборудования на нее. Определить DDoS это или нет можно с помощью утилит netstat и iftop.

Намеренную DDoS-атаку сложно остановить лишь с помощью программных средств, однако некоторые несложные скрипты позволяют отражать небольшие «любительские» атаки на отказ. Подобный скрипт блокировки DDoS-атак представлен в приложении А. Скрипт анализирует текущие подключения к серверу на основе утилиты netstat, в случае превышения определенного лимита подключений, адрес попадает в блок IPTables на некоторое время.

Для блокировки намеренных атак требуется аппаратная защита на уровне сетевого оборудования. Защита от распределенных DDoS-атак основывается на многофакторном анализе трафика, который поступает на каждый защищаемый сервер. Во время нормальной работы система защиты может самообучаться

или настраиваться, а после обнаружения атаки либо автоматически, либо по требованию, активно противодействует нелегитимному трафику.

В данном случае защитником выступает дата-центр, который имеет в распоряжении оборудование, способное осуществлять фильтрацию трафика и предотвращение атак на отказ.

5.7 Общие рекомендации по администрированию инфраструктуры

Следует следить за новостными рассылками о критических уязвимостях в операционной системе и используемом программном обеспечении.

Если работа сервера замедлилась, в диагностике проблемы помогут утилиты ps, top, atop, htop. В случае обнаружения сетевых проблем на помощь придут утилиты ping, traceroute, nmap, mtr, tcpdump, iftop.

Важно следить за состоянием RAID на сервере, в случае сбоя одного из дисков, следует обратиться к поддержке дата-центра с просьбой о замене диска.

Необходимо подбирать сложные для перебора пароли, периодически их менять и уведомлять пользователей о смене пароля. Для генерации паролей подходит утилита pwgen:

```
# pwgen 12 -slyc  
Y;Xhfu;04A{O
```

Стоит аккуратно работать на сервере под учетной записью root, не стоит запускать неизвестные скрипты или двоичные файлы. Необходимо проверять контрольные суммы загруженных пакетов и следить за цифровой подписью.

Не следует просто так перезагружать сервер, рекомендуется это делать только в случаях чрезвычайной необходимости (например обновление ядра).

Всегда стоит проверять наличие актуальных резервных копий, а также свободное место на системах хранения данных. Стоит также настроить ротацию логов, это позволит сэкономить место на диске.

На физических нодах стоит устанавливать только самый необходимый, минимальный набор программного обеспечения, для уменьшения вероятности компрометации сервера.

Всегда стоит вести документацию в том или ином виде и логировать свои действия на сервере. Логи, которые могут помочь в диагностике неисправностей:

- /var/log/{apache2,httpd}/error.log;
- /var/log/auth.log;
- /var/log/dmesg;
- /var/log/kern.log;
- /var/log/libvirt/libvirtd.log;
- /var/log/mail.log;
- /var/log/messages;
- /var/log/mysql/mysql-slow.log;
- /var/log/nginx/error.log;
- /var/log/syslog;
- /var/log/vzctl.log.

В случае обнаружения неполадок с аппаратной частью, стоит незамедлительно обратиться к провайдеру для проверки работоспособности аппаратуры. В случае недочетов в программном обеспечении стоит обратиться к технической поддержке ПО.

6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

В руководстве администратора рассмотрен процесс заказа услуги пользователем, внешний интерфейс панелей управления. Панели биллинга, панели управления хостингом, контейнерами и виртуальными машинами имеют простой интуитивный интерфейс с возможностью тонкой настройки.

Рассмотрен процесс подачи запроса в техническую поддержку. Для пользователя доступны ответы на самые часто задаваемые вопросы в разделе wiki.

Руководство пользователя представлено в приложении Б.

7 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

В режиме нормального функционирования мониторинг показывает полную работоспособность всех сервисов системы (рис. 7.1).

dams-oo-ups1	Battery expired	OK	13-12-2011 09:43:58	32d 18h 0m 3s	1/2	OK: noBatteryNeedsReplacing
	Battery status	OK	13-12-2011 09:45:32	32d 18h 0m 3s	1/2	OK: batteryNormal
	Battery temperature	OK	13-12-2011 09:44:41	32d 18h 0m 3s	1/2	SNMP OK - 28
	Input voltage	OK	13-12-2011 09:44:08	0d 14h 5m 54s	1/2	SNMP OK - 244
	Output load	OK	13-12-2011 09:45:32	32d 18h 0m 3s	1/2	SNMP OK - 35
	Output status	OK	13-12-2011 09:45:00	0d 10h 49m 2s	1/2	OK: onLine
	Temperature sensor	OK	13-12-2011 09:44:00	32d 17h 58m 40s	1/2	SNMP OK - 23
dams-oo-ups2	Battery expired	OK	13-12-2011 09:44:00	70d 2h 7m 53s	1/2	OK: noBatteryNeedsReplacing
	Battery status	OK	13-12-2011 09:44:00	70d 0h 44m 29s	1/2	OK: batteryNormal
	Battery temperature	OK	13-12-2011 09:44:00	70d 2h 6m 15s	1/2	SNMP OK - 30
	Input voltage	OK	13-12-2011 09:45:24	0d 14h 6m 38s	1/2	SNMP OK - 244
	Output load	OK	13-12-2011 09:44:00	70d 2h 6m 17s	1/2	SNMP OK - 57
	Output status	OK	13-12-2011 09:44:41	0d 10h 49m 21s	1/2	OK: onLine
dams-oo-ups3	Battery expired	OK	13-12-2011 09:44:49	70d 2h 7m 29s	1/2	OK: noBatteryNeedsReplacing
	Battery status	OK	13-12-2011 09:45:32	70d 2h 6m 32s	1/2	OK: batteryNormal
	Battery temperature	OK	13-12-2011 09:44:00	70d 2h 7m 33s	1/2	SNMP OK - 35
	Input voltage	OK	13-12-2011 09:44:00	0d 8h 2m 2s	1/2	SNMP OK - 246
	Output load	OK	13-12-2011 09:44:42	70d 2h 7m 53s	1/2	SNMP OK - 23
	Output status	OK	13-12-2011 09:45:30	0d 4h 40m 32s	1/2	OK: onLine
dams-oo-ups4	Battery expired	OK	13-12-2011 09:44:01	70d 2h 6m 8s	1/2	OK: noBatteryNeedsReplacing
	Battery status	OK	13-12-2011 09:44:00	38d 19h 7m 14s	1/2	OK: batteryNormal
	Battery temperature	OK	13-12-2011 09:44:49	70d 2h 7m 16s	1/2	SNMP OK - 26
	Input voltage	OK	13-12-2011 09:45:42	0d 8h 8m 20s	1/2	SNMP OK - 244
	Output load	OK	13-12-2011 09:45:08	70d 2h 7m 28s	1/2	SNMP OK - 45
	Output status	OK	13-12-2011 09:45:31	0d 5h 8m 31s	1/2	OK: onLine
dams-oo-ups5	Battery expired	OK	13-12-2011 09:44:00	34d 18h 39m 8s	1/2	OK: noBatteryNeedsReplacing
	Battery status	OK	13-12-2011 09:44:42	34d 18h 39m 7s	1/2	OK: batteryNormal
	Battery temperature	OK	13-12-2011 09:44:42	34d 18h 39m 7s	1/2	SNMP OK - 29
	Input voltage	OK	13-12-2011 09:44:32	0d 14h 7m 30s	1/2	SNMP OK - 243
	Output load	OK	13-12-2011 09:45:32	34d 18h 39m 8s	1/2	SNMP OK - 55
	Output status	OK	13-12-2011 09:45:49	0d 10h 52m 13s	1/2	OK: onLine
dams-oo-ups6	Battery expired	OK	13-12-2011 09:45:31	70d 2h 6m 16s	1/2	OK: noBatteryNeedsReplacing
	Battery status	OK	13-12-2011 09:44:49	70d 2h 5m 52s	1/2	OK: batteryNormal
	Battery temperature	OK	13-12-2011 09:45:44	38d 18h 22m 2s	1/2	SNMP OK - 35
	Input voltage	OK	13-12-2011 09:45:28	0d 8h 8m 34s	1/2	SNMP OK - 244
	Output load	OK	13-12-2011 09:44:12	70d 2h 7m 33s	1/2	SNMP OK - 16
	Output status	OK	13-12-2011 09:45:07	0d 5h 10m 55s	1/2	OK: onLine
dams-rsl-sw1	ports	OK	13-12-2011 09:44:38	19d 19h 1m 19s	1/2	OK: host '10.2.0.251', interfaces up: 41, down: 0, dormant: 0, excluded: 0, unused: 0
dams-rsl-sw2	ports	OK	13-12-2011 09:45:29	19d 19h 0m 33s	1/2	OK: host '10.2.0.250', interfaces up: 5, down: 0, dormant: 0, excluded: 0, unused: 0

Рис. 7.1 – Работоспособность сервисов в штатном режиме

В случае появления внештатных ситуаций, система мониторинга фиксирует аномальные явления, такие как недоступность хостов или отдельных сервисов. Мониторинг способен запоминать код ошибки, поэтому диагностика неисправностей является несложной задачей. Пример реакции системного мониторинга на недоступность сервера представлен на рис. 7.2.

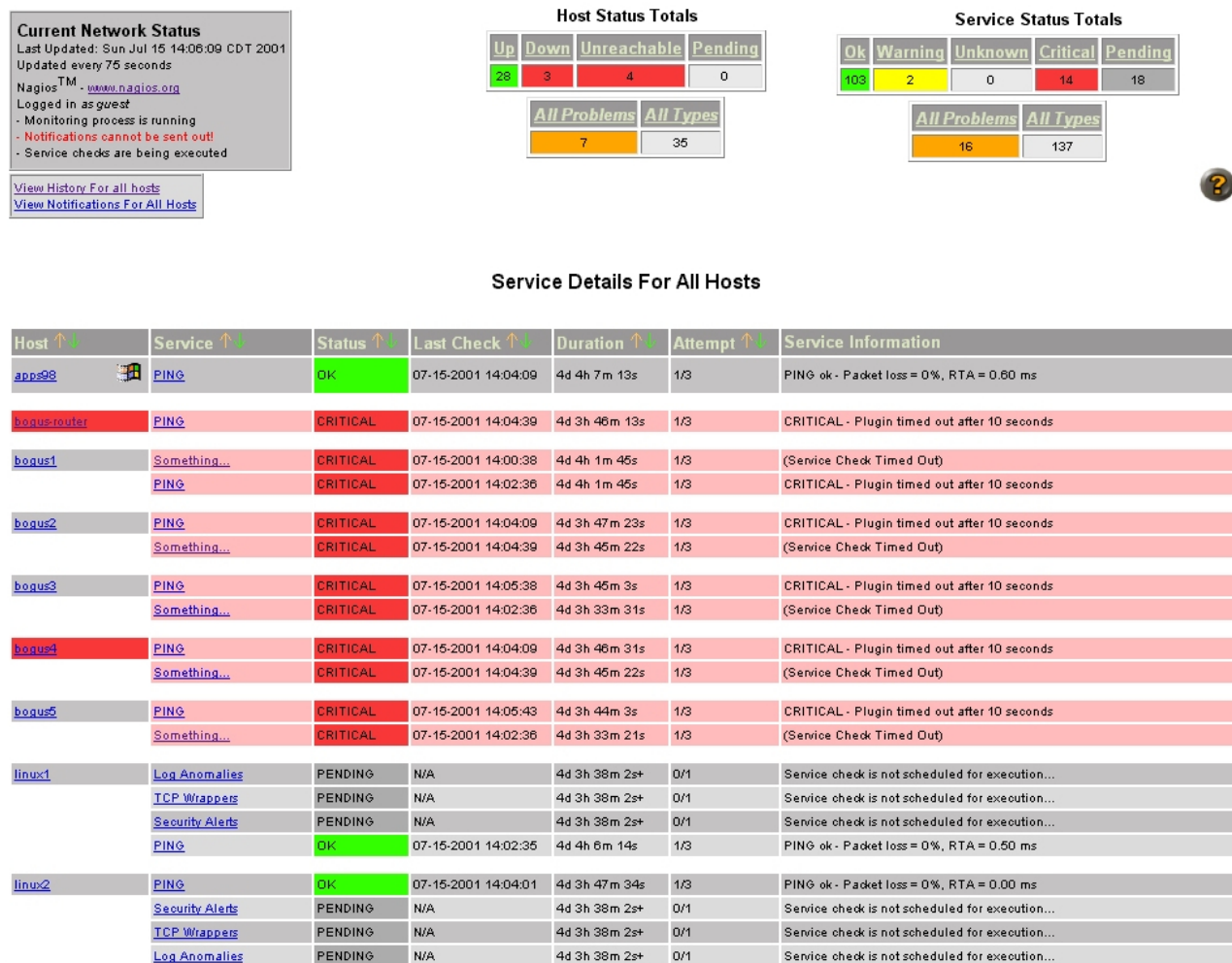


Рис. 7.2 – Недоступность некоторых серверов инфраструктуры

Пример реакции системы мониторинга Nagios на неисправность одного из дисков в RAID-массиве представлен на рис. 7.3.

Hardware	CRITICAL	01-26-2011 08:29:38	1d 15h 21m 10s	3/3	CRITICAL : Disk Drive Bay 1 Drive 5: In Critical Array CRITICAL : Disk Drive Bay 1 Drive 4: In Critical Array CRITICAL : Disk Drive Bay 1 Drive 3: In Critical Array CRITICAL : Disk Drive Bay 1 Drive 2: Drive Fault CRITICAL : Disk Drive Bay 1 Drive 2: In Critical Array CRITICAL : Disk Drive Bay 1 Drive 1: In Critical Array CRITICAL : Disk Drive Bay 1 Drive 0: In Critical Array CRITICAL : Drive 2 in enclosure 32 on controller 0 Fw: D305 - FAILED WARNING : RAID 5 Logical Volume 0 on controller 0, Drives(0e32,1e32,2e32,3e32,4e32,5e32) - DEGRADED - Server: Dell Inc. PowerEdge 2950 s/h: System BIOS: 2.4.3 2008-08-15
----------	----------	---------------------	----------------	-----	---

Рис. 7.3 – Сбой работы диска в RAID-массиве

7.1 Нагрузка сети во время резервного копирования

Каждую ночь на серверах запускается задача резервного копирования, в это время наблюдается повышенная нагрузка на сервере, связанная со снятием дампов баз данных, архивацией и сжатием архивов для передачи по сети на сервер резервного копирования.

Как правило повышение нагрузки незначительное, что можно наблюдать на рис. 7.4.

```
top - 13:10:28 up 5 days, 6:26, 1 user, load average: 3,04, 3,47, 4,40
Tasks: 185 total, 1 running, 184 sleeping, 0 stopped, 0 zombie
%Cpu(s): 26,1 us, 8,1 sy, 0,0 ni, 65,2 id, 0,5 wa, 0,0 hi, 0,0 si, 0,0 st
KiB Mem: 24745308 total, 14228264 used, 10517044 free, 1392500 buffers
KiB Swap: 16383996 total, 59036 used, 16324960 free, 9615508 cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
21171	mysql	20	0	849m	527m	5532	S	27,6	2,2	497:16.96	mysqld
4437	root	20	0	816m	307m	12m	S	3,0	1,3	60:51.18	core
6127	www-data	20	0	331m	257m	2236	S	1,0	1,1	0:03.62	nginx
21194	root	20	0	548m	64m	11m	S	1,0	0,3	26:35.18	core
31987	root	20	0	99,2m	5360	1876	S	1,0	0,0	6:29.09	ihttpd
7221	root	20	0	322m	42m	568	S	0,7	0,2	0:02.78	apache2
18065	root	20	0	322m	42m	568	S	0,7	0,2	0:00.06	apache2
18181	root	20	0	322m	42m	568	S	0,7	0,2	0:00.02	apache2
34	root	20	0	0	0	0	S	0,3	0,0	3:44.26	kworker/7:0
46	root	20	0	0	0	0	S	0,3	0,0	4:12.10	kworker/2:1
50	root	20	0	0	0	0	S	0,3	0,0	3:33.21	kworker/6:1
223	root	20	0	0	0	0	S	0,3	0,0	38:07.82	jbd2/sda3-8
6126	www-data	20	0	331m	257m	2204	S	0,3	1,1	0:01.41	nginx
9860	root	20	0	322m	42m	568	S	0,3	0,2	0:02.06	apache2
14453	root	20	0	322m	42m	568	S	0,3	0,2	0:01.05	apache2
15297	root	20	0	322m	42m	568	S	0,3	0,2	0:00.77	apache2
16826	root	20	0	322m	42m	568	S	0,3	0,2	0:00.35	apache2
16827	root	20	0	322m	42m	568	S	0,3	0,2	0:00.39	apache2
17319	root	20	0	322m	42m	568	S	0,3	0,2	0:00.25	apache2
17846	root	20	0	322m	42m	568	S	0,3	0,2	0:00.09	apache2
18158	root	20	0	24476	1744	1196	R	0,3	0,0	0:00.02	top
28120	root	20	0	322m	51m	9524	S	0,3	0,2	2:23.84	apache2
1	root	20	0	10648	680	648	S	0,0	0,0	0:34.27	init

Рис. 7.4 – Состояние системы во время процесса резервного копирования

На графике Munin можно наблюдать значительное повышение утилизации сетевого канала во время резервного копирования. График представлен на рис. 7.5. Значения представленные на графике являются условными, сетевого канала в 10 Мб/с явно недостаточно для такой инфраструктуры. На деле используется сеть с пропускной способностью 100 Мб/с.

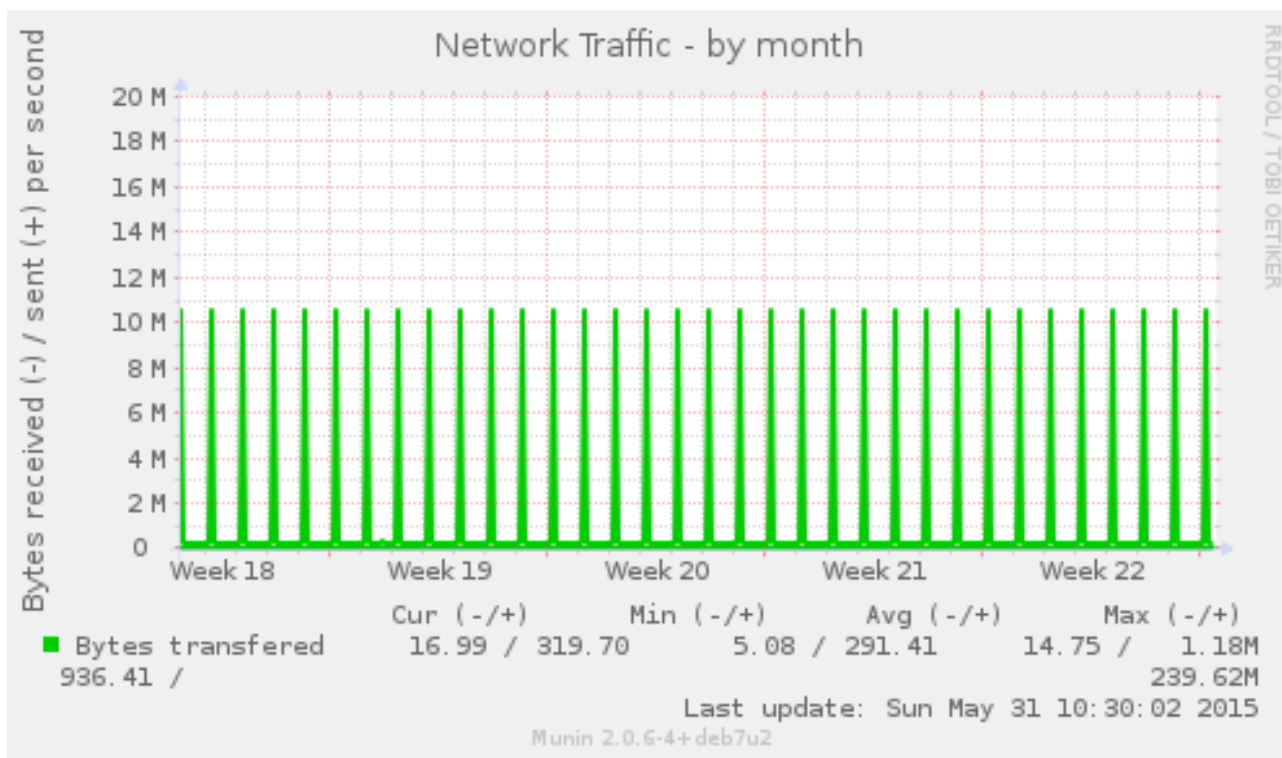


Рис. 7.5 – Состояние системы во время процесса резервного копирования

На графике представлено состояние канала сети за последний месяц, пиками на графике являются значения максимальной пропускной способности сети.

7.2 Состояние сервера в период атаки на отказ

Можно выделить три основных типа DDoS атак:

- переполнение сетевого канала;
- большое количество одновременных подключений (syn-flood);
- исчерпание процессорных мощностей сервера (HTTP-flood).

От последних двух типов атак можно защищаться средствами операционной системы. В случае переполнения сетевого канала, решением проблемы является расширение сетевого канала.

Состояние нагрузки системы во время DDoS-атаки можно наблюдать на рис. 7.6.

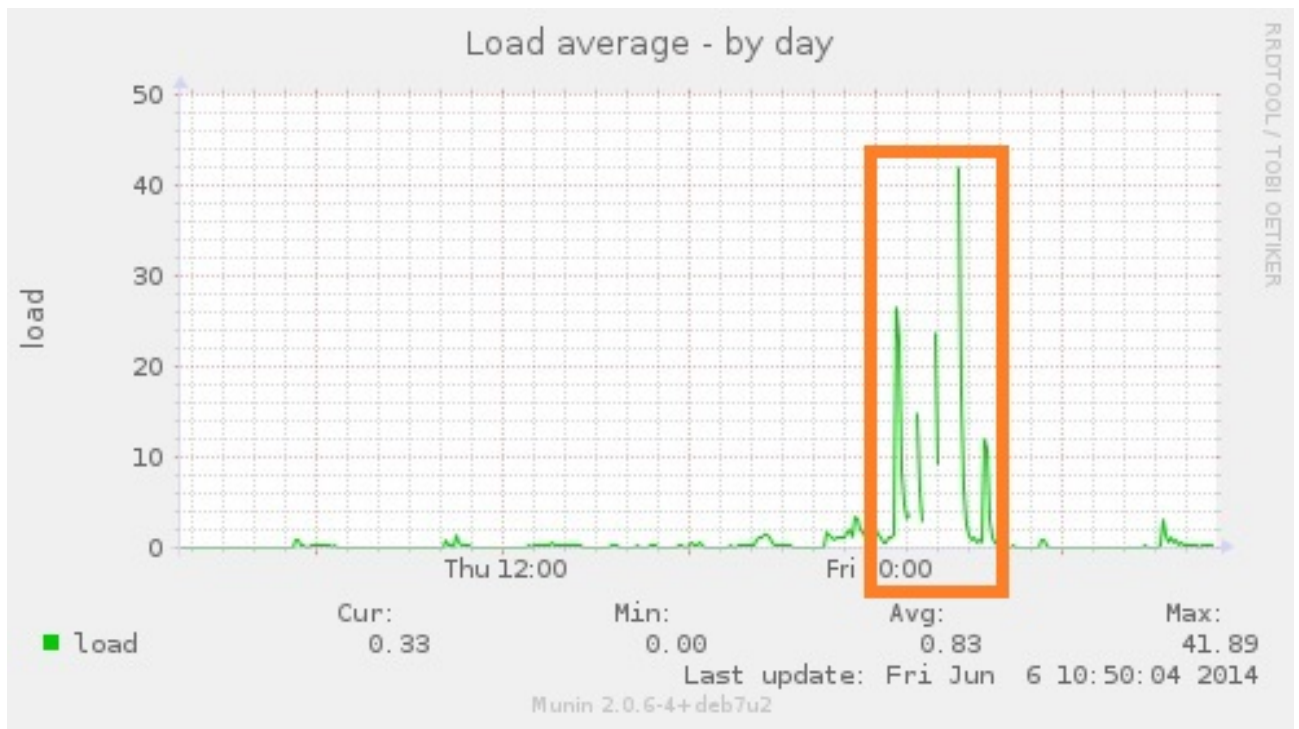


Рис. 7.6 – Значительное повышение нагрузки в период 0:00

В данный промежуток времени нагрузка составила 40 LA, что превышает среднюю нагрузку на используемом сервере в 20 раз. При этом, в процессах системы наблюдается явная нагрузка на сервер баз данных (рис. 7.7).

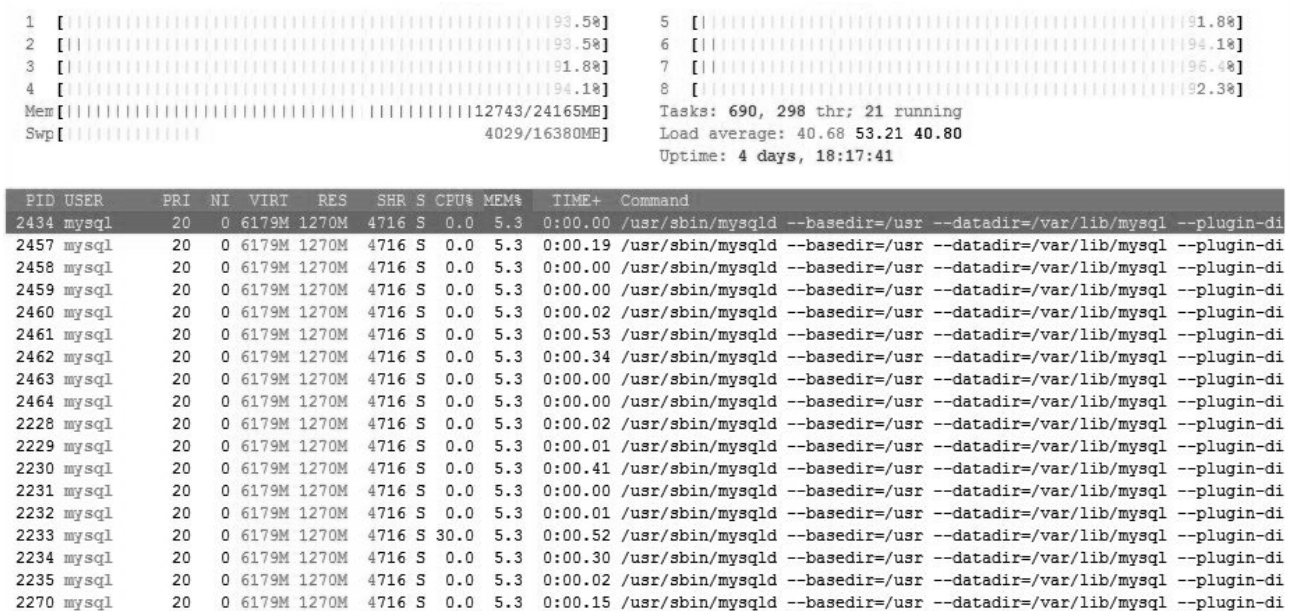


Рис. 7.7 – Все процессорные ядра заняты обработкой запросов

На графиках соединений с брандмауэром также наблюдается аномальное явление (рис. 7.8).

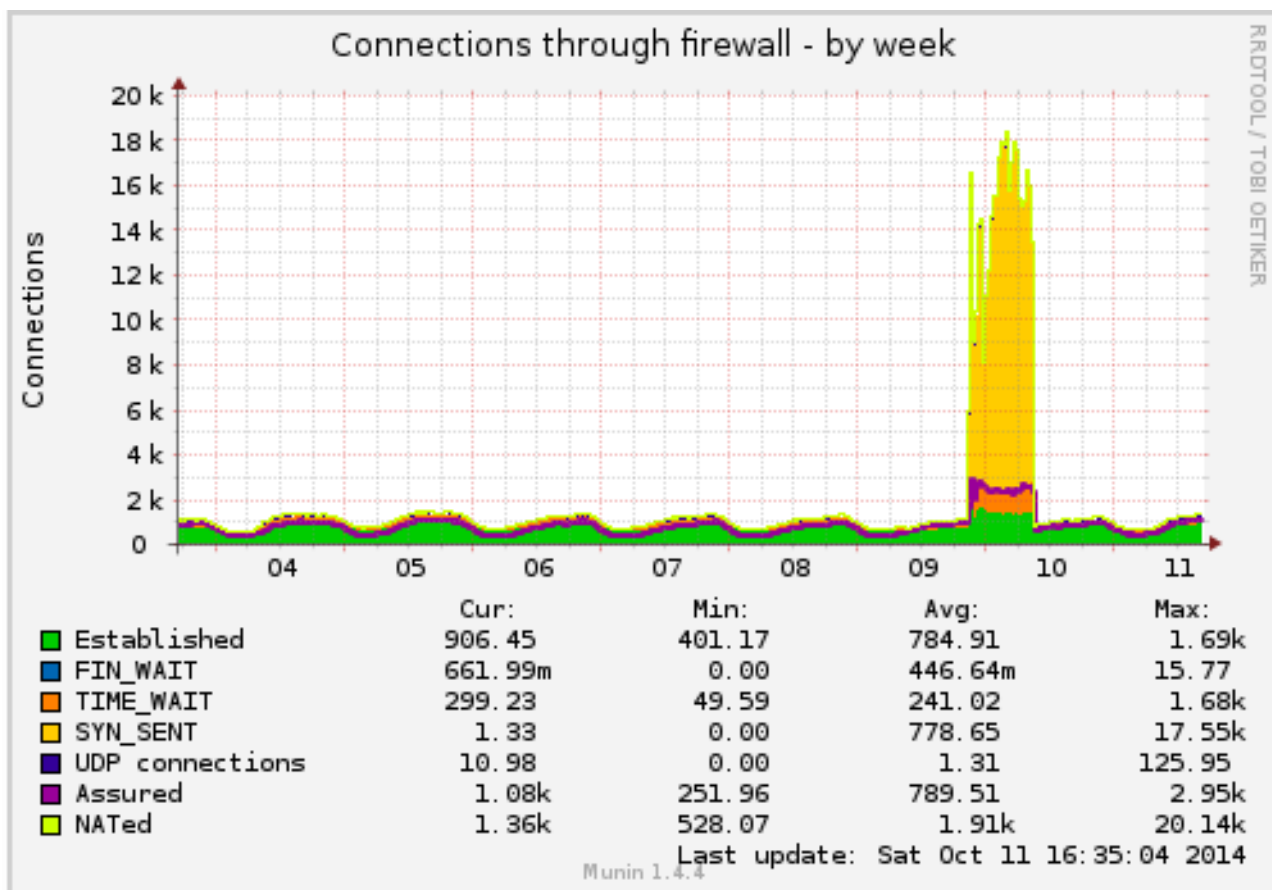


Рис. 7.8 – Превышено число соединений к брандмауэру в 40 раз

Если взглянуть на вывод команды netstat, то можно заметить наиболее активные адреса:

```
# netstat -ntu | awk '{print $5}' | cut -d: -f1 | sort | \
uniq -c | sort -rn | grep -v 127.0.0.1 | head
22373 209.72.209.54
17766 88.81.32.224
16060 176.209.90.198
13226 217.209.214.248
221 62.163.78.143
221 188.255.255.177
220 198.27.82.153
118 216.242.148.101
113 95.216.88.50
111 216.73.216.251
```

7.3 Отказ одного из DNS-серверов

Nagios зафиксировал недоступность одного из DNS-серверов, неспособного осуществлять обработку запросов (рис. 7.9).

DNS IP Match	OK	05-31-2015 10:24:24	1010d 17h 4m 5s	1/5	DNS OK: 0.006 seconds response time. www.nagios.com returns 50.116.21.73
DNS Resolution	OK	05-31-2015 10:24:34	1804d 2h 9m 6s	1/5	DNS OK: 0.005 seconds response time. www.nagios.com returns 50.116.21.73
HTTP	OK	05-31-2015 10:21:09	962d 17h 25m 30s	1/5	HTTP OK: HTTP/1.1 301 Moved Permanently - 557 bytes in 0.001 second response time
Ping	OK	05-31-2015 10:22:33	1771d 9h 57m 6s	1/5	OK - www.nagios.com: rta 0.261ms, lost 0%
DNS IP Match	CRITICAL	05-31-2015 10:21:37	943d 17h 27m 37s	5/5	DNS CRITICAL - expected '199.59.148.10,199.59.150.39,199.59.150.7' but got '199.59.148.10,199.59.148.82,199.59.149.198,199.59.150.7'
DNS Resolution	CRITICAL	05-31-2015 10:24:39	1758d 15h 42m 10s	1/5	DNS CRITICAL 199.59.148.10,199.59.149.198,199.59.149.230,199.59.150.39
HTTP	UNKNOWN	05-31-2015 10:21:14	382d 20h 27m 48s	5/5	check_http: Invalid onredirect option - -u
Ping	OK	05-31-2015 10:24:19	1758d 15h 44m 33s	1/5	OK - www.twitter.com: rta 42.352ms, lost 0%

Рис. 7.9 – Недоступность DNS-сервера

При этом все сайты доступны и работают в штатном режиме, так как отсутствие одного из трех серверов не влияет на работоспособность всей инфраструктуры. Один из подчиненных серверов все еще доступен и способен обрабатывать пользовательские DNS-запросы.

После возобновления работоспособности сервера, оба подчиненных сервера функционируют в штатном режиме:

```
$ host -t ns site.ru
site.ru name server ns1.site.ru.
site.ru name server ns2.site.ru.
```

Таким образом, в ходе проектирования и тестирования инфраструктуры удалось реализовать отказоустойчивую систему, способную оперативно обнаруживать неисправности в инфраструктуре. Также немаловажным фактором является реакция команды системных администраторов и технической поддержки дата-центра, в условиях критических ситуация важно проявить особую внимательность в решении проблем.

8 БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

В разделе выполнен анализ условий труда администратора и их требованиям освещенности, электробезопасности, микроклимата, шума, требованиям к оборудованию, к организации рабочего места пользователя ПК.

8.1 Краткая характеристика помещения

Согласно санитарным нормам, ширина стола должна быть не менее 0.6 м, глубина не менее 0.8 м, также необходимо обеспечить расстояние между боковыми поверхностями мониторов не менее 1.2 м.

На рис. 8.1 изображена планировка и размещение оборудования на рабочем месте.

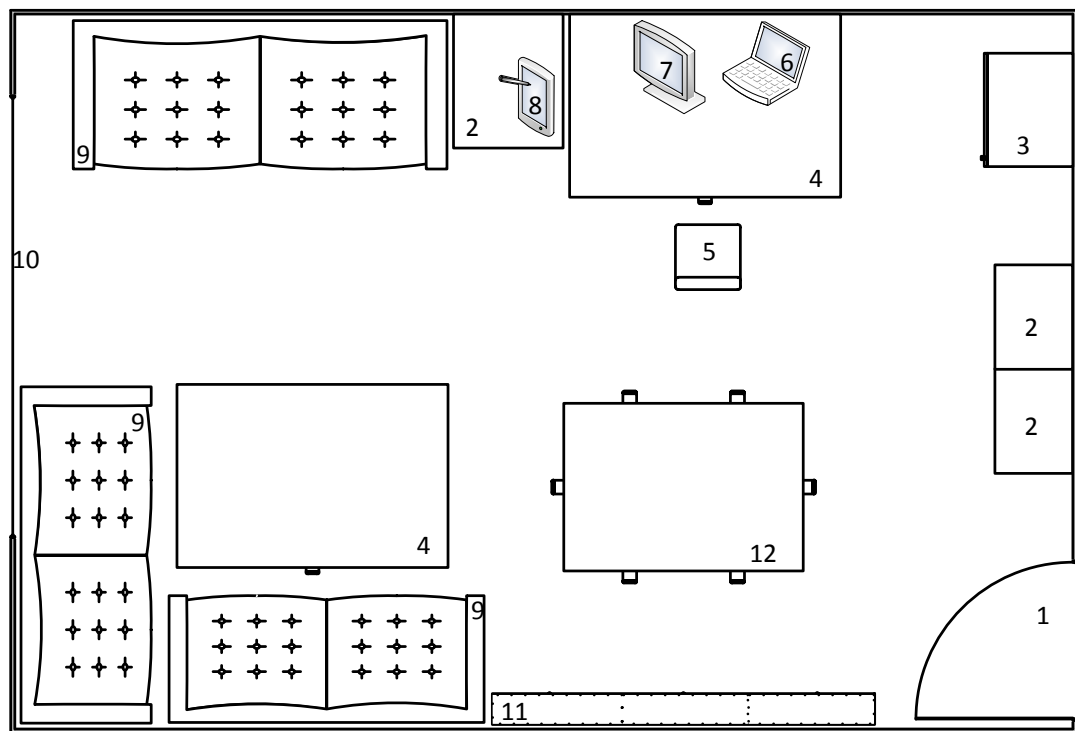


Рис. 8.1 – Планировка и размещение оборудования на рабочем месте

1 — дверь; 2 — тумбочка; 3 — холодильник; 4 — стол; 5 — кресло; 6 — ноутбук; 7 — монитор; 8 — планшет; 9 — кровать; 10 — окно; 11 — шкаф; 12 — обеденный стол

8.2 Микроклимат

Метеорологические условия (микроклимат) характеризуются следующими параметрами:

- температурой воздуха;
- относительной влажностью;
- скоростью движения воздуха на рабочем месте;
- барометрическим давлением.

В комнате, где находится персональный компьютер, должен поддерживаться определенный температурный режим для нормальной эксплуатации ЭВМ и условий труда администратора.

Параметры воздушной среды в кабинете должны соответствовать требованиям ГОСТ 12.1.005-88 «Общие санитарно-гигиенические требования к воздуху санитарной зоны». При этом необходимо учитывать, что работа администратора относится к разряду легких работ (разряд 1а — затраты энергии до 150 ккал/час), а кабинет — к производственным помещениям.

В помещении имеются источники избыточного тепла:

- тепловыделение от ноутбука;
- тепло, выделяемое персоналом.

Тепловыделение от светильников отсутствует, так как используется люминесцентное освещение. Для поддержания оптимального уровня температуры используется как естественная вентиляция (через двери и окна путем проветривания), так и искусственная (при помощи вентилятора), так как в текущих условиях проживания невозможно установить кондиционер. Для обогрева помещения в зимнее время используется водяное отопление.

8.3 Освещение

В зависимости от необходимости, производственное освещение в кабинете может быть как естественным, создаваемым непосредственно солнцем и диффузным (рассеянным) светом, так и искусственным, осуществляемым электрическими лампами. Естественное освещение характеризуется тем, что создаваемое освещение изменяется в очень широких пределах, в зависимости от времени года, дня и метеорологических факторов. При выборе норм

естественного освещения учитывается разряд выполняемых работ, система освещения, коэффициент солнечности, коэффициент светового климата [14].

Мероприятия, за счет которых выполняются требования норм СП 52.13330.2011 «Естественное и искусственное освещение»:

- проверка, не реже одного раза в год, соответствия освещенности на рабочей поверхности нормам искусственного освещения;
- очистка светильников не реже одного раза в три месяца;
- протирка окон (стекол) не реже двух раз в год.

8.4 Шум и вибрация

Источниками шума в помещении являются:

- непосредственно персональный компьютер (вентиляторы охлаждения процессора и видеокарты);
- разговорная речь;
- шум вне рабочей зоны.

Постоянный шум оказывает отрицательное воздействие на человека, как биологически, так и психологически, что отражается на качестве работы и общей производительности труда сотрудников. Снижается производительность труда и повышается количество допущенных ошибок, некоторые из которых могут быть критическими. Допустимый уровень звука — 50 дБА, при работающем оборудовании в кабинете ожидаемый уровень звука — 40-48 дБА.

8.5 Пожаробезопасность

Пожар в помещении может возникнуть при взаимодействии горючих веществ, окислителя (условия пожара) и источников воспламенения (причина пожара). Горючие вещества в кабинете: деревянные столы, двери, полы (паркет), покрытия стен, изоляция соединительных кабелей, жидкости для протирки узлов компьютера и другие.

Возможные источники и причины возникновения пожара:

- эксплуатация неиспользованного оборудования;
- неправильное применение электронагревательных приборов;

- неисправность проводки;
- короткое замыкание;
- нарушение правил пожарной безопасности.

Для отвода тепла от персонального компьютера необходимы работающие вентиляторы, помещение проветривается, поэтому кислород, как окислитель процессов горения, имеется в достаточном количестве. Исходя из этого, помещение кабинета, согласно нормам СП 2.13130.2012 «Системы противопожарной защиты. Обеспечение огнестойкости объектов защиты», по степени пожаробезопасности следует отнести к категории Д (помещения, в которых в обращении находятся негорючие вещества и материалы в холодном состоянии).

В качестве средств тушения пожара применяются углекислотные огнетушители, используемые для тушения электроустановок, находящихся под напряжением.

8.6 Электробезопасность

В кабинет электроэнергия поступает для питания персональных компьютеров и электрического освещения. Питание осуществляется от трехфазной сети переменного тока напряжением 380/220 В (+10..-15%) частотой 50 Гц (+1 Гц).

Поскольку помещение сухое (относительная влажность не более 75%), температура не превышает 30 °С, то, согласно ПУЭ 7 («Правилам устройства электроустановок»), оно не относится к категории помещений повышенной опасности. Однако возможна потенциальная опасность поражения людей электрическим током. Источниками и причинами опасности являются:

- открытые токопроводящие части оборудования, кабельной проводки;
- неисправность электрооборудования, электрических розеток;
- короткое замыкание в результате повреждения изоляции.

Для предотвращения поражения электрическим током потребителей электроэнергии в кабинете необходимо предусмотреть следующие технические мероприятия:

- все токопроводящие части оборудования и кабельной проводки должны быть защищены ограждающими кожухами;
- все металлические конструкции, которые могут оказаться под напряжением в результате короткого замыкания, должны быть заземлены, защищены и выполнено защитное отключение.

В качестве заземляющих проводников должны быть использованы элементы металлических конструкций, металлическое обрамление кабельных каналов. Здание должно быть оборудовано комплексом мер, предотвращающих попадание энергии молнии в электрическую сеть, а также поражение людей, для чего на здание устанавливаются громоотводы.

Кроме технических, необходимо проведение организационных мероприятий:

- к работе с электроустановками допускаются только лица, прошедшие инструктаж и проверку знаний правил техники безопасности в соответствии с ГОСТ 12.1.009-76, ПТЭ и ПТБ;
- периодически осуществляется контроль сопротивления электрической изоляции токоведущих частей (в соответствии с требованиями ПУЭ 7, оно не должно быть ниже 0.5 мм по отношению к корпусу ЭВМ).

8.7 Эргономика и техническая эстетика

Эффективность работы администратора (программиста) во многом зависит от организации рабочих мест. Рабочее положение администратора — сидячее. Стул по возможности должен быть регулируемым по высоте, поскольку клавиатура и дисплей компьютеров должны находиться в зоне наилучшего обзора. Для сохранения работоспособности имеет большое значение выбор основной рабочей позы.

Техническая эстетика позволяет снижать нервное утомление и вредные воздействия на функции организма в процессе труда. Огромное значение в эстетическом оформлении производства имеет цвет. Окраска, форма, внешний вид производственного помещения и оборудования улучшают условия освещения, а также психологическое состояние человека. Стены имеют светло-

зеленый цвет, не вызывающий раздражения, потолок — белый цвет, что обеспечивает максимальное отражение света.

Рассматриваемое помещение соответствует требованиям ГОСТ 12.2.032-78 «Рабочее место при выполнении работ сидя. Общие эргономические требования».

8.8 Режим труда и отдыха

Работа администратора относится к категории работ связанных с опасными и вредными условиями труда. В процессе труда на администратора оказывают действие следующие опасные и вредные производственные факторы, физические:

- повышенный уровень статического электричества;
- повышенный уровень шума;
- повышенные уровни запыленности воздуха рабочей зоны;
- повышенная яркость светового изображения;
- повышенный или пониженный уровень освещенности;
- неравномерность распределения яркости в поле зрения;
- повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека.

Рациональный режим труда и отдыха — это правильное чередование работы и перерывов в ней в течение смены, суток, недели, года, устанавливаемое с целью обеспечения высокой производительности труда и сохранения здоровья работающих. Основным перерывом является перерыв на обед. Обеденный перерыв при 8-часовой рабочей смене устанавливается продолжительностью не менее 30 мин через 4 часа после начала работы.

Режим отдыха складывается из нескольких компонентов:

- времени на гигиенические процедуры и личные надобности (2-3) от сменного времени независимо от вида труда;
- времени регламентированных перерывов на отдых (входят в состав рабочего времени), определяемого по показателю условий труда или по интегральному показателю снижения работоспособности;

- времени микропауз, а также времени обеденного перерыва (нерабочего времени), остающегося от приема пищи.

Режим труда и отдыха должен быть построен в соответствии с особенностями трудовой деятельности пользователей персонального компьютера и характером функциональных изменений со стороны различных систем организма работников.

8.9 Выводы

В ходе выполнения работы, была дана краткая характеристика помещения и выполняемых работ. Составлен план помещения и размещения оборудования. Были определены оптимальные параметры микроклимата, шума, освещения. Даны рекомендации по эргономике и режиму труда.

Наиболее неблагоприятными факторами являются микроклимат и искусственное освещение, обеспечиваемое люминесцентными лампами.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы бакалавра была разработана виртуальная инфраструктура для реализации облачных услуг. Данная инфраструктура успешно используется в бизнесе и обладает высоким уровнем отказоустойчивости, надежности и масштабируемости.

Инфраструктура прошла все этапы проектирования и разработки, начиная от закупки серверов и лицензий на программное обеспечение до окончательного тестирования. Тестирование показало, что при отказе некоторых компонентов инфраструктуры, включая управляющие сервера, система все еще продолжает функционировать.

Продолжительное время находятся в эксплуатации различные системы виртуализации, такие как OpenVZ и KVM. Сочетание работы этих систем в связке с программным обеспечением, ориентированным на пользователей позволяет организовать продажу услуг в сфере облачных вычислений.

В ходе разработки тарифов для пользователей были подобраны оптимальные параметры контейнеров, виртуальных машин и лимитов хостинга. Использование свободного программного обеспечения позволило сократить материальные расходы на ПО.

Данная виртуальная инфраструктура может быть расширена за счет использования других систем виртуализаций, таких как Xen, добавления дополнительных обслуживающих серверов, а также улучшение отказоустойчивости за счет применения репликации данных. Обновление аппаратной части инфраструктуры также увеличит производительность, однако данные действия требуют существенных материальных затрат.

При анализе безопасности и применении мер по ее обеспечению были использованы и дописаны такие скрипты как DDoS Deflate, который является эффективным решением защиты от небольших DDoS-атак. Скрипт не является законченным продуктом и распространяется под свободной лицензией.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Каталог программных продуктов семейства Oracle Database [Электронный ресурс] // Oracle Россия URL: http://oracle.ocs.ru/files/catalog_Oracle_Database_12C.pdf (дата обращения: 13.04.2015)
2. Tier: уровни надежности ЦОД и что из этого следует [Электронный ресурс] // AboutDC. Решения для ЦОД URL: <http://www.aboutdc.ru/page/390.php> (дата обращения: 14.04.2015)
3. Джонс Т.М. Виртуальный Linux. Обзор методов виртуализации, архитектур и реализаций [Электронный ресурс] // IBM developerWorks Россия: Ресурс IBM для разработчиков и IT профессионалов URL: <http://www.ibm.com/developerworks/ru/library/l-linuxvirt/index.html> (дата обращения: 14.04.2015)
4. Умеров А.Р. Руководство по созданию и управлению контейнерами на базе OpenVZ [Электронный ресурс] // GitHub URL: <https://github.com/Amet13/openvz-tutorial/blob/master/main.pdf> (дата обращения: 15.04.2015)
5. Джонс Т.М. Эмуляция систем с помощью QEMU [Электронный ресурс] // IBM developerWorks Россия: Ресурс IBM для разработчиков и IT профессионалов URL: <http://www.ibm.com/developerworks/ru/library/l-qemu/> (дата обращения: 16.04.2015)
6. Толети Б.П. Гипервизоры, виртуализация и облако: Анализ гипервизора KVM [Электронный ресурс] // IBM developerWorks Россия: Ресурс IBM для разработчиков и IT профессионалов Режим доступа: <http://www.ibm.com/developerworks/ru/library/cl-hypervisorcompare-kvm/> (дата обращения: 16.04.2015)
7. Чубин И. Руководство пользователя Xen v3.0 [Электронный ресурс] // Xgu.ru URL: <http://xgu.ru/xen/manual/> (дата обращения: 18.04.2015)

8. Одинцов П. Контейнеризация на Linux в деталях – LXC и OpenVZ Часть 2 [Электронный ресурс] // Хабрахабр URL: <http://habrahabr.ru/company/FastVPS/blog/209084/> (дата обращения: 19.04.2015)

9. Виртуализация на уровне ОС: теория и практика LXC [Электронный ресурс] // creativeyp.com URL: <http://creativeyp.com/568-virtualizaciya-na-urovne-os-teoriya-i-praktika-lxc.html> (дата обращения: 20.04.2015)

10. Основы виртуализации. Инфраструктура [Электронный ресурс] // BW-IT.RU | Информационные технологии URL: http://www.bw-it.ru/virtualization_virtual_infrastructure.php (дата обращения: 20.04.2015)

11. Методические указания «Процедура системного анализа при проектировании программных систем» для студентов-дипломников дневной и заочной формы обучения специальности 7.091501 / Сост.: Сергеев Г.Г., Скатков А.В., Мащенко Е.Н. – Севастополь: Изд-во СевНТУ, 2005. – 32с.

12. Немец Э., Снайдер Г., Хейн Т., Уэйли Б. Unix и Linux: руководство системного администратора, 4-е изд.: Пер. с англ. – М.: ООО «И.Д. Вильямс», 2012. – 1312 с.: ил.

13. Балансировка нагрузки: основные алгоритмы и методы [Электронный ресурс] // Блог компании Селектел URL: <http://blog.selectel.ru/balansirovka-nagruzki-osnovnye-algoritmy-i-metody/> (дата обращения: 06.05.2015)

14. Методические указания для выполнения раздела «Охрана труда и окружающей среды» в дипломных проектах специальностей 7.080401 — «Информационные и управляющие системы и технологии», 7.092502 — «Компьютерно-интегрированные технологические процессы и производства», 7.091401 — «Системы управления автоматике», 7.091501 — «Компьютерные системы и сети» / Сост.: Е.И Азаренко. – Севастополь: Изд-во СевНТУ, 2005. – 10с.

ПРИЛОЖЕНИЕ

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД СКРИПТА DDOS DEFLATE

```
1 ##### Paths of the script and other files
2 PROGDIR="/usr/local/ddos"
3 PROG="/usr/local/ddos/ddos.sh"
4 IGNORE_IP_LIST="/usr/local/ddos/ignore.ip.list"
5 APF="/etc/apf/apf"
6 IPT="/sbin/iptables"
7
8 ##### How many connections define a bad IP? Indicate that below.
9 NO_OF_CONNECTIONS=150
10
11 ##### APF_BAN=1 (Make sure your APF version is atleast 0.96)
12 ##### APF_BAN=0 (Uses iptables for banning ips instead of APF)
13 APF_BAN=0
14
15 ##### KILL=0 (Bad IPs are'nt banned, good for interactive execution
    of script)
16 ##### KILL=1 (Recommended setting)
17 KILL=1
18
19 ##### An email is sent to the following address when an IP is
    banned.
20 ##### Blank would suppress sending of mails
21 EMAIL_TO="root"
22
23 ##### Number of seconds the banned ip should remain in blacklist.
24 BAN_PERIOD=600

```



```
1 #!/bin/bash
2 #####
3 # DDoS-Deflate version 0.6 Author: Zaf <zaf@vsnl.com>
4 # It's fork of DDoS-Deflate by Amet13 <admin@amet13.name>
5 # https://github.com/Amet13/ddos-deflate
6 #####
```

```

7 # This program is distributed under the "Artistic License"
8 # Agreement
9 #
10 # The LICENSE file is located in the same directory as
11 # this program. Please read the LICENSE file before you
12 # make copies or distribute this program
13 #####
14 load_conf()
15 {
16     CONF="/usr/local/ddos/ddos.conf"
17     if [ -f "$CONF" ] && [ "$CONF" != "" ]; then
18         source $CONF
19     else
20         head
21         echo "\$CONF not found."
22         exit 1
23     fi
24 }
25
26 head()
27 {
28     echo "DDoS-Deflate version 0.6"
29     echo "Copyright (C) 2005, Zaf <zaf@vsnl.com>"
30     echo "It's fork of DDoS-Deflate by Amet13 <admin@amet13.name>"
31     echo
32 }
33
34 showhelp()
35 {
36     head
37     echo 'Usage: ddos.sh [OPTIONS] [N]'
38     echo 'N : number of tcp/udp connections (default 150)'
39     echo 'OPTIONS:'
40     echo '-h | --help: Show this help screen'
41     echo '-k | --kill: Block the offending ip making more than N
42         connections'
43 }

```

```

43
44 unbanip()
45 {
46     UNBAN_SCRIPT='mkttemp /tmp/unban.XXXXXXXXXX'
47     TMP_FILE='mkttemp /tmp/unban.XXXXXXXXXX'
48     UNBAN_IP_LIST='mkttemp /tmp/unban.XXXXXXXXXX'
49     echo '#!/bin/sh' > $UNBAN_SCRIPT
50     echo "sleep $BAN_PERIOD" >> $UNBAN_SCRIPT
51     if [ $APF_BAN -eq 1 ]; then
52         while read line; do
53             echo "$APF -u $line" >> $UNBAN_SCRIPT
54             echo $line >> $UNBAN_IP_LIST
55         done < $BANNED_IP_LIST
56     else
57         while read line; do
58             echo "$IPT -D INPUT -s $line -j DROP" >> $UNBAN_SCRIPT
59             echo $line >> $UNBAN_IP_LIST
60         done < $BANNED_IP_LIST
61     fi
62     echo "grep -v --file=$UNBAN_IP_LIST $IGNORE_IP_LIST > $TMP_FILE"
63     >> $UNBAN_SCRIPT
64     echo "mv $TMP_FILE $IGNORE_IP_LIST" >> $UNBAN_SCRIPT
65     echo "rm -f $UNBAN_SCRIPT" >> $UNBAN_SCRIPT
66     echo "rm -f $UNBAN_IP_LIST" >> $UNBAN_SCRIPT
67     echo "rm -f $TMP_FILE" >> $UNBAN_SCRIPT
68     . $UNBAN_SCRIPT &
69 }
70
71 load_conf
72 while [ $1 ]; do
73     case $1 in
74         '-h' | '--help' | '?' )
75             showhelp
76             exit
77             ;;
78         '--kill' | '-k' )
79             KILL=1

```



```

79         ;;
80     *[0-9]* )
81         NO_OF_CONNECTIONS=$1
82         ;;
83     * )
84         showhelp
85         exit
86         ;;
87     esac
88     shift
89 done
90
91 TMP_PREFIX='/tmp/ddos'
92 TMP_FILE="mktemp $TMP_PREFIX.XXXXXXXXXX"
93 BANNED_IP_MAIL='$TMP_FILE'
94 BANNED_IP_LIST='$TMP_FILE'
95 echo "Banned the following ip addresses on 'date'" >
    $BANNED_IP_MAIL
96 echo "From 'hostname -f' ('hostname --ip-address')" >>
    $BANNED_IP_MAIL
97 echo >> $BANNED_IP_MAIL
98 BAD_IP_LIST='$TMP_FILE'
99 netstat -ntu | awk '{print $5}' | cut -d: -f1 | sort | uniq -c |
    sort -nr > $BAD_IP_LIST
100 cat $BAD_IP_LIST
101 if [ $KILL -eq 1 ]; then
102     IP_BAN_NOW=0
103     while read line; do
104         CURR_LINE_CONN=$(echo $line | cut -d" " -f1)
105         CURR_LINE_IP=$(echo $line | cut -d" " -f2)
106         if [ $CURR_LINE_CONN -lt $NO_OF_CONNECTIONS ]; then
107             break
108         fi
109         IGNORE_BAN=$(grep -c $CURR_LINE_IP $IGNORE_IP_LIST)
110         if [ $IGNORE_BAN -ge 1 ]; then
111             continue
112         fi

```

```

113     IP_BAN_NOW=1
114     echo "$CURR_LINE_IP with $CURR_LINE_CONN connections" >>
        $BANNED_IP_MAIL
115     echo $CURR_LINE_IP >> $BANNED_IP_LIST
116     echo $CURR_LINE_IP >> $IGNORE_IP_LIST
117     if [ $APF_BAN -eq 1 ]; then
118         $APF -d $CURR_LINE_IP
119     else
120         $IPT -I INPUT -s $CURR_LINE_IP -j DROP
121     fi
122 done < $BAD_IP_LIST
123 if [ $IP_BAN_NOW -eq 1 ]; then
124     dt='date'
125     if [ $EMAIL_TO != "" ]; then
126         cat $BANNED_IP_MAIL | mail -s "IP addresses banned on $dt"
            $EMAIL_TO
127     fi
128     unbanip
129 fi
130 fi
131 rm -f $TMP_PREFIX.*

1 127.0.0.1

1 #!/bin/sh
2 if [ -d '/usr/local/ddos' ]; then
3     echo; echo; echo "Please uninstall the previous version first"
4     exit 0
5 else
6     mkdir /usr/local/ddos
7 fi
8 clear
9 echo; echo 'Installing DOS-Deflate 0.6'; echo
10 echo; echo -n 'Downloading source files...' ; echo
11 wget -q -O /usr/local/ddos/ddos.conf https://raw.githubusercontent.com/Amet13/ddos-deflate/master/ddos.conf
12 echo -n '30% '

```

```

13 wget -q -O /usr/local/ddos/LICENSE https://raw.githubusercontent.com/Amet13/ddos-deflate/master/LICENSE
14 echo -n '50% '
15 wget -q -O /usr/local/ddos/ignore.ip.list https://raw.githubusercontent.com/Amet13/ddos-deflate/master/ignore.ip.list
16 echo -n '100%'
17 wget -q -O /usr/local/ddos/ddos.sh https://raw.githubusercontent.com/Amet13/ddos-deflate/master/ddos.sh
18 chmod 0755 /usr/local/ddos/ddos.sh
19 echo '... done'
20 echo; echo 'Installation has completed'
21 echo 'Config file is at /usr/local/ddos/ddos.conf'
22 echo 'Please send in your comments and/or suggestions to zaf@vsnl.com'
23 echo 'About new changes: admin@amet13.name'
24 echo 'Do not forget create a cron job!'
25 cat /usr/local/ddos/LICENSE | less

1 #!/bin/sh
2 echo; echo "Uninstalling DOS-Deflate"
3 echo; echo; echo -n "Deleting script files..."
4 if [ -e '/usr/local/sbin/ddos' ]; then
5     rm -f /usr/local/sbin/ddos
6     echo -n "..."
7 fi
8 if [ -d '/usr/local/ddos' ]; then
9     rm -rf /usr/local/ddos
10    echo -n "..."
11 fi
12 echo "done"
13 echo; echo "Uninstall Complete"; echo

1 ddos-deflate
2 =====
3 Shell script blocking DDoS attacks. Fork of [(D)DoS Deflate](http://deflate.medialayer.com/).
4 It works on Debian 7 and CentOS 7 (please tell me if you've tested script on other distros).

```

```

5  Installation
6  -----
7  '''bash
8  su -
9  cd /tmp
10 wget https://raw.githubusercontent.com/Amet13/ddos-deflate/master/
    install.sh
11 chmod +x install.sh
12 ./install.sh
13 '''
14 Add your contact e-mail:
15 '''bash
16 vim /usr/local/ddos/ddos.conf
17 EMAIL_TO="mail@example.com"
18 '''
19 Add your ignore ip's to ignorelist:
20 '''bash
21 vim /usr/local/ddos/ignore.ip.list
22 127.0.0.1
23 1.1.1.1
24 2.2.2.2
25 '''
26 Add cronjob:
27 '''bash
28 crontab -e
29 # run script every minute
30 * * * * * /usr/local/ddos/ddos.sh >/dev/null 2>&1
31 '''
32 Check:
33 '''bash
34 /usr/local/ddos/ddos.sh
35     724 127.0.0.1
36     214 2.2.2.2
37     59 3.3.3.3
38 ...
39 '''

```

ПРИЛОЖЕНИЕ Б

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

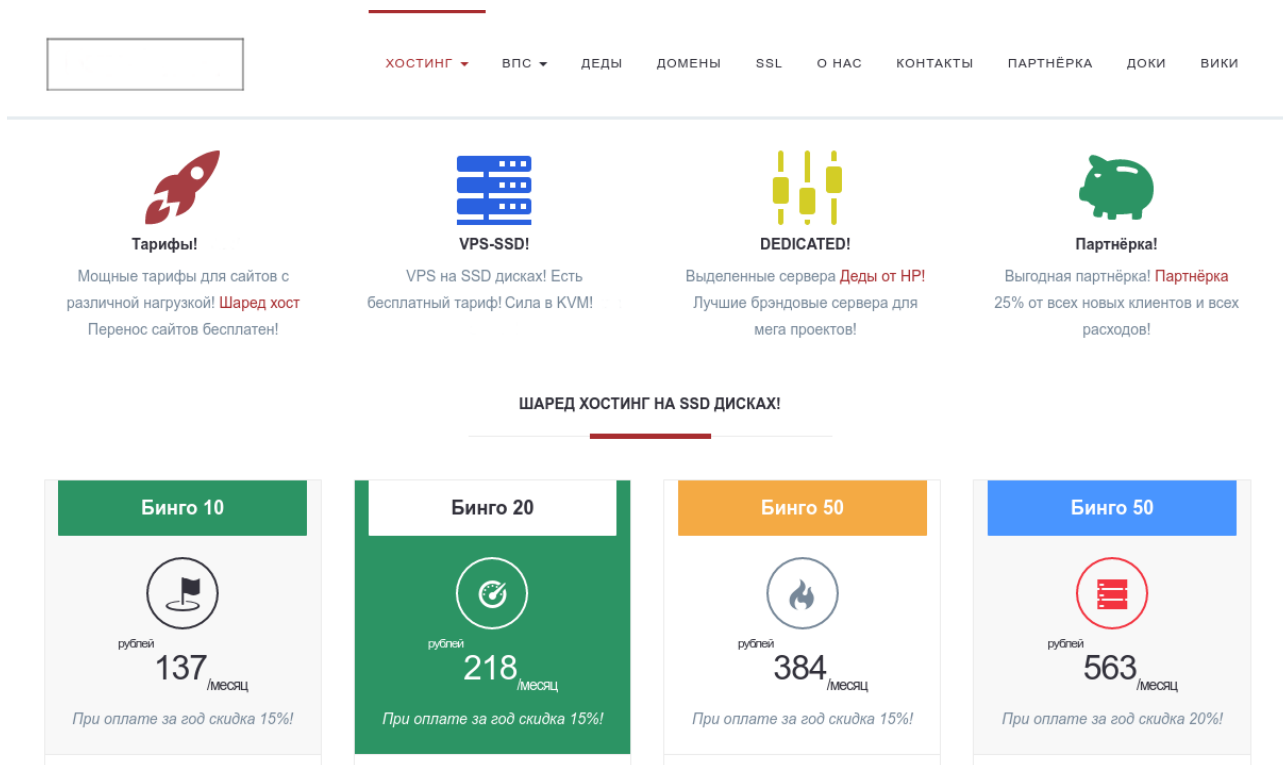


Рис. Б.1 — Список тарифов на сайте

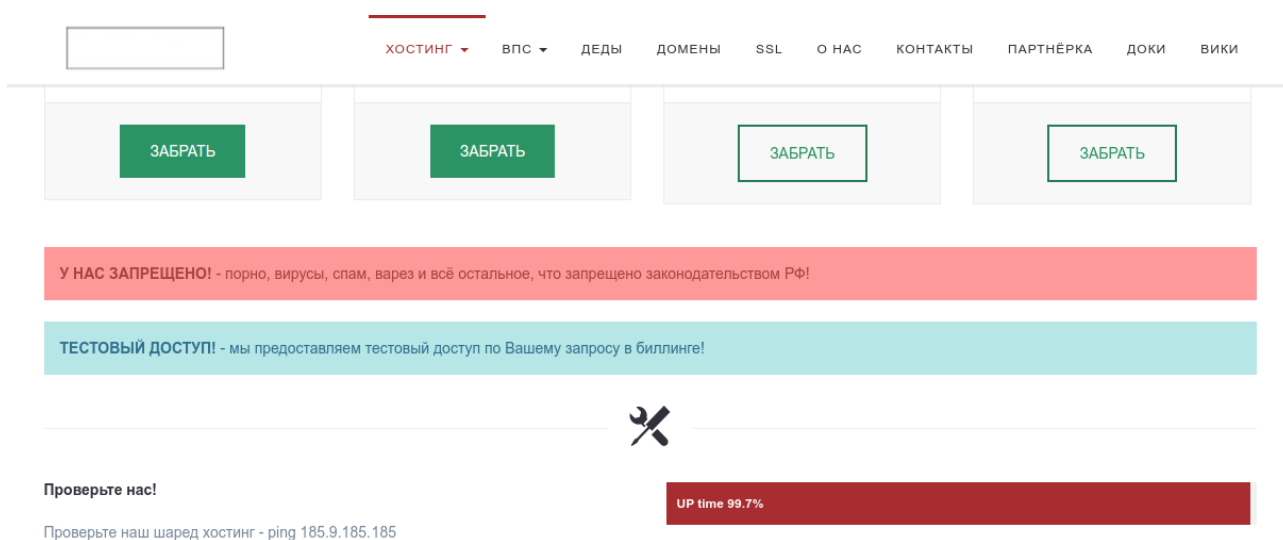


Рис. Б.2 — Для выбора нужного тарифа необходимо нажать кнопку «ЗАБРАТЬ» под ним

Регистрация

У меня уже есть учетная запись

Язык интерфейса

Русский

Имя пользователя

Иван Иванов

Пароль доступа

.....

Подтверждение пароля

.....

E-mail адрес

test@example.com

Страна

Российская Федерация

Статус заказчика

Частное лицо

Контактное лицо (Ф.И.О.)

Иванов И.И,

Откуда вы узнали о нас

Поисковые системы

Ok

Рис. Б.3 — Регистрация в биллинге для возможности заказа услуги

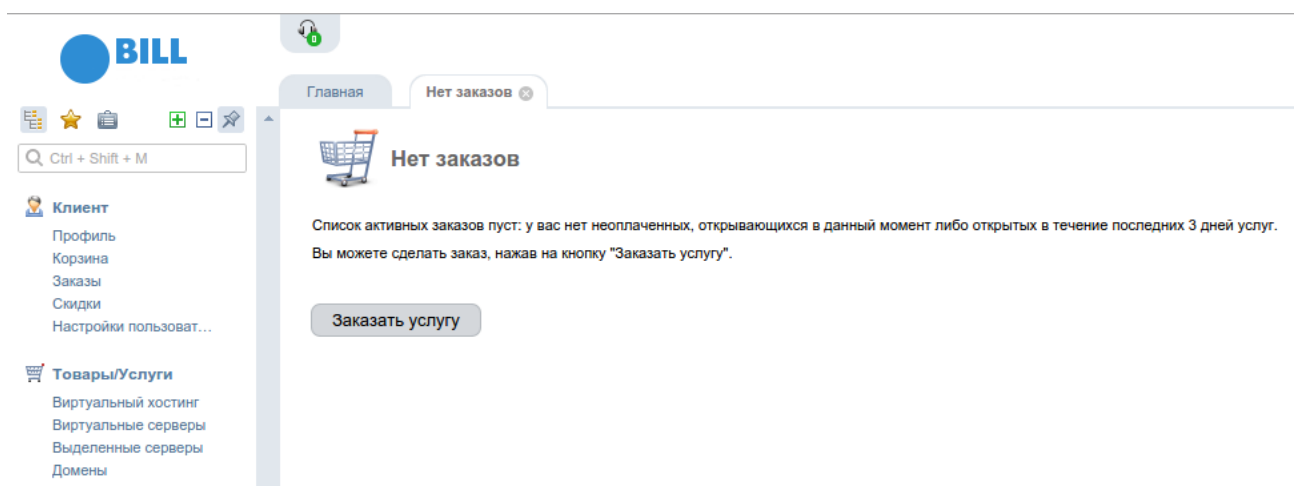


Рис. Б.4 — Пример заказа услуги в биллинге

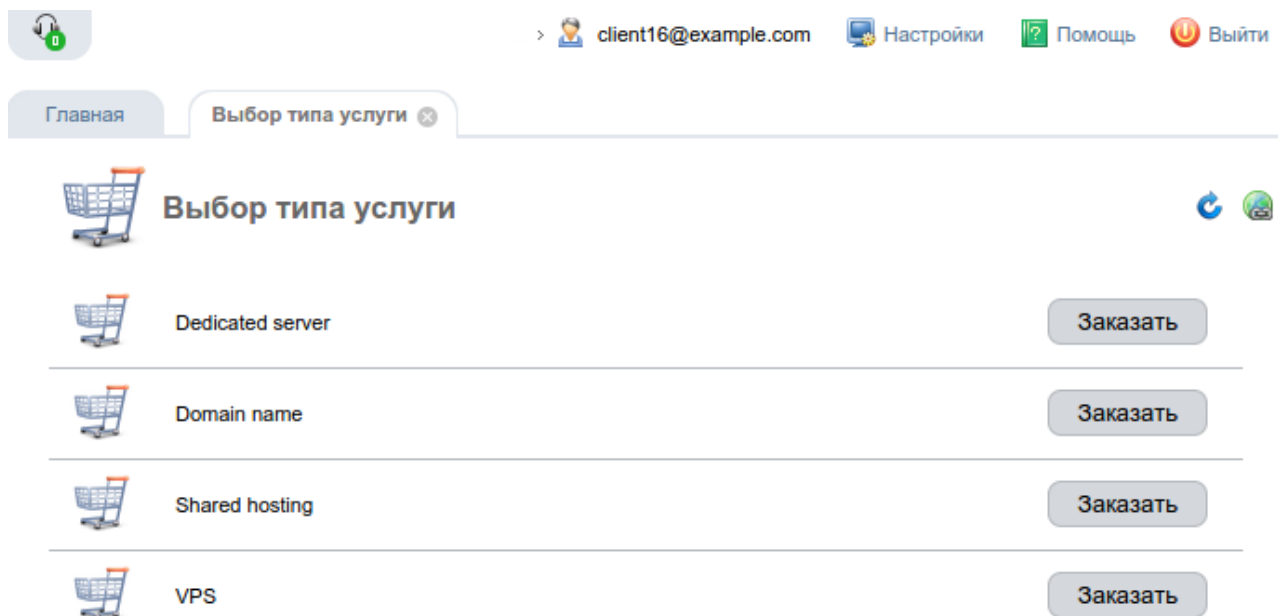


Рис. Б.5 — Выбор типа услуги

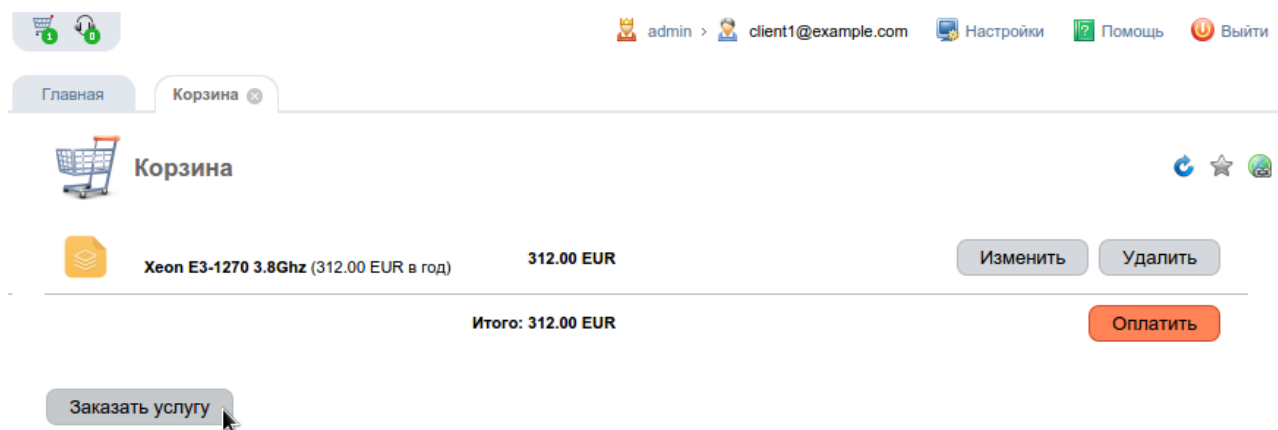


Рис. Б.6 — Список услуг в корзине

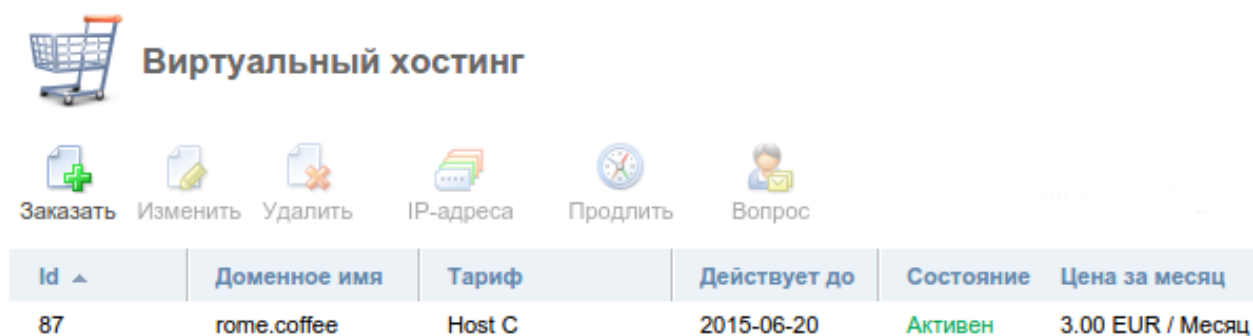


Рис. Б.7 — Список заказанных услуг в разделе хостинга

Главная

Расходы

Расходы

🔄

★

📄

🔍 Ctrl + Shift + F

Id	Наименование	Дата списания	Сумма	Не оплачено	Оплачено платежами
10	Телематические услуги - VPS Two #70 (pretty.one) , Месяц	2015-03-20	11.00 EUR	0.00 EUR	pxf/7
11	Телематические услуги - Host A #80, Полгода	2015-03-20	6.00 EUR	0.00 EUR	pxf/7

Рис. Б.8 — Список расходов на услуги

Рис. Б.9 — Настройки пользователя хостинга

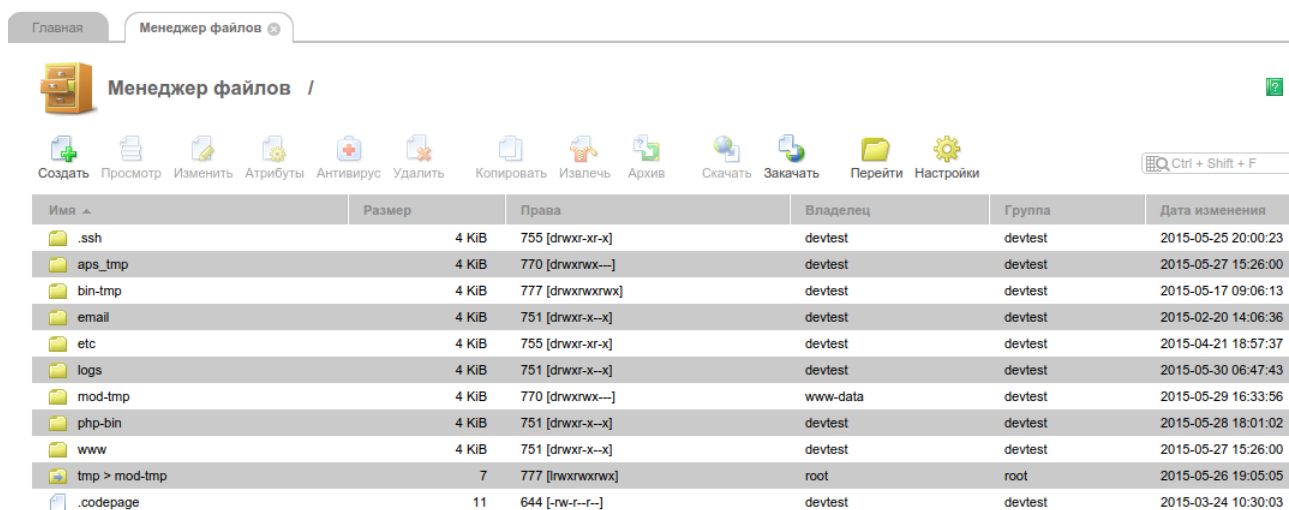


Рис. Б.10 — Файловый менеджер в панели управления хостингом

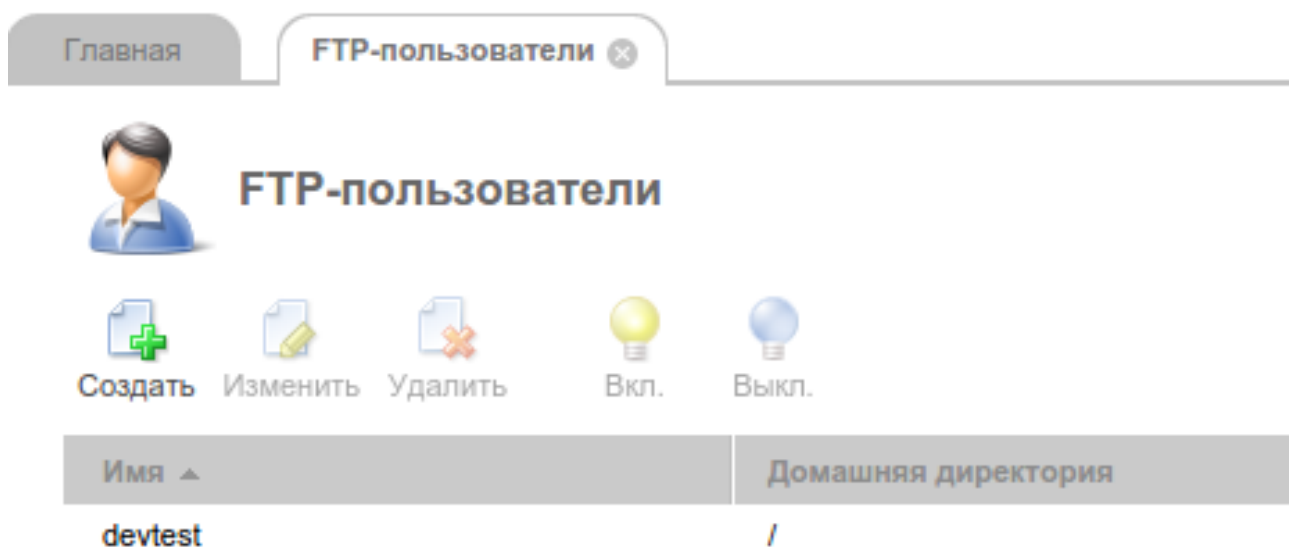


Рис. Б.11 — Список FTP-пользователей

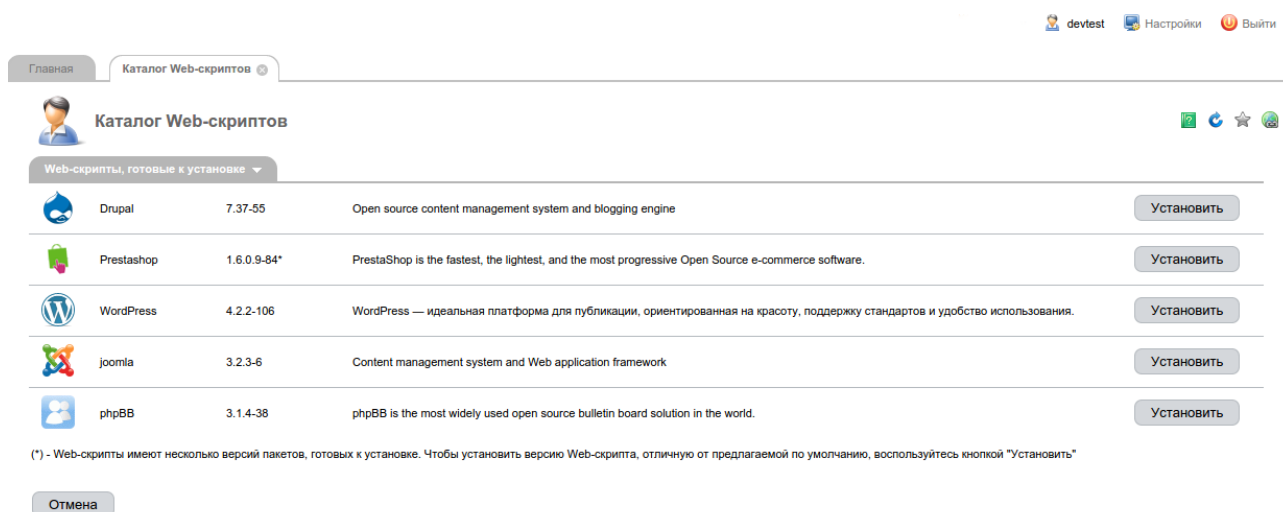


Рис. Б.12 — Возможность установки популярных CMS из каталога Web-скриптов

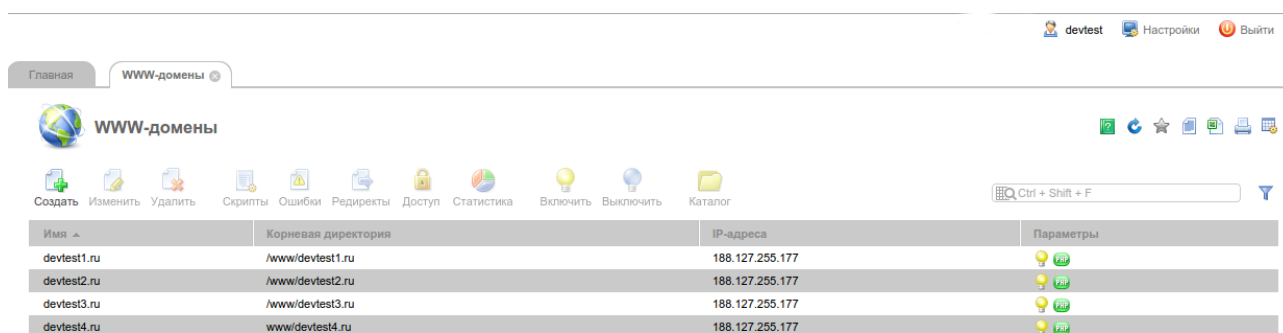


Рис. Б.13 — Управление www-доменами



Рис. Б.14 — Управление резервными копиями

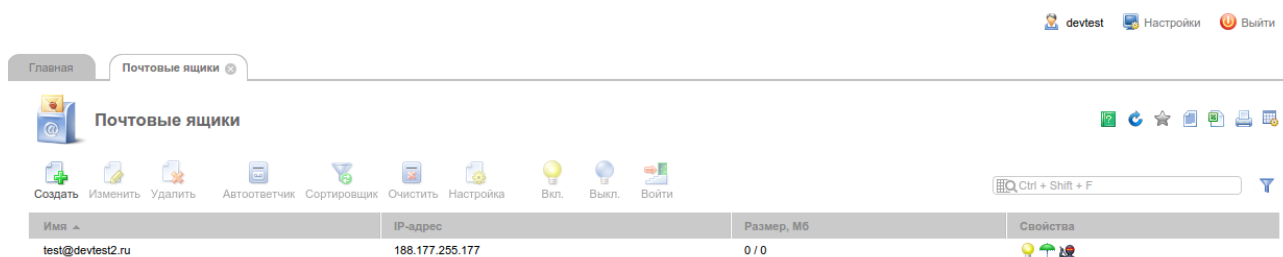


Рис. Б.15 — Управление почтовыми ящиками

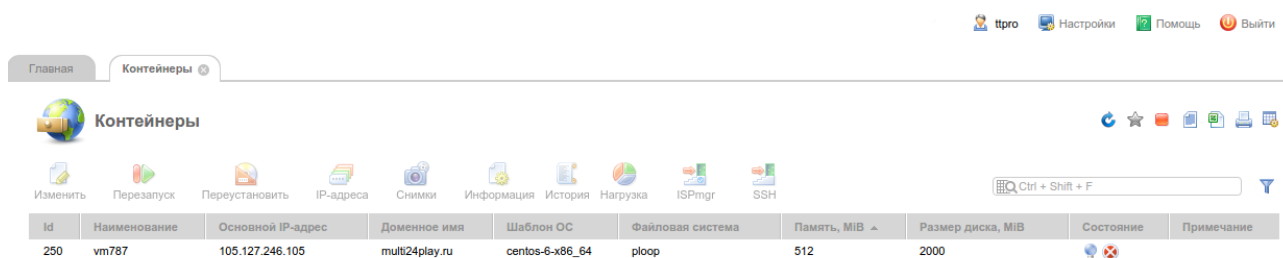


Рис. Б.16 — Список контейнеров в панели управления OpenVZ

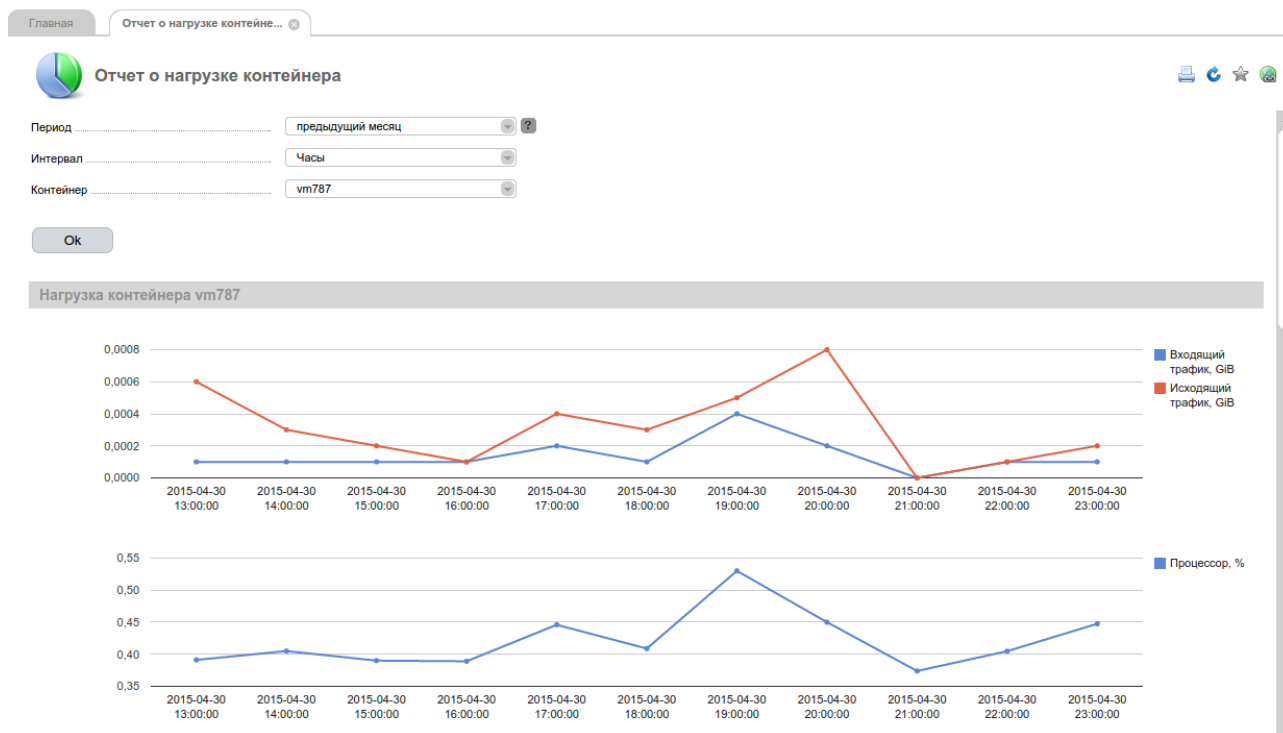


Рис. Б.17 — Отчет о нагрузке контейнера

<div> <div>Главная</div> <div>Журнал посещений</div> </div>		
<div> <div>Журнал посещений</div> <div>Нажмите на заголовок, чтоб обновить данные</div> </div>		
Время	Пользователь	Удалённый IP-адрес
2015-05-18 16:17:59	ttpro	95.156.156.54
2015-05-13 16:10:12	ttpro	95.215.158.220
2015-05-12 17:34:31	ttpro	95.215.158.220
2015-05-12 17:34:31	ttpro	95.215.158.220
2015-05-12 15:57:48	ttpro	95.215.158.220
2015-05-05 16:24:14	ttpro	95.215.157.152
2015-04-16 16:20:57	ttpro	178.212.102.38
2015-04-10 00:47:27	ttpro	171.212.97.171
2015-04-07 22:43:53	ttpro	95.215.156.54
2015-04-07 19:10:23	ttpro	95.215.156.54
2015-04-07 19:10:23	ttpro	95.215.156.54
2015-04-07 18:21:28	ttpro	54.215.156.54
2015-04-07 18:21:28	ttpro	95.156.156.54
2015-04-07 18:09:59	ttpro	54.215.156.54
2015-04-07 18:09:59	ttpro	95.156.156.54
2015-04-07 18:04:52	ttpro	54.215.156.54


Рис. Б.18 — Журнал посещений панели OpenVZ

<div> <div>Виртуальные машины</div> <div> <div>Создать</div> <div>Изменить</div> <div>Удалить</div> <div>Старт</div> <div>Стоп</div> <div>Перезапуск</div> <div>Переустановить</div> <div>Пароль</div> <div>Диски</div> <div>Интерфейсы</div> <div>IP-адреса</div> <div>Информация</div> <div>VNC</div> </div> <div>Ctrl + Shift + F</div> </div>									
Id	Наименование	Доменное имя	Основной IP-адрес	Шаблон ОС	Память, MiB	Количество ядер	Размер дисков, MiB	Состояние	
137	testISP5	testisp5.ru	250.127.238.250	CentOS-6-amd64	1024	1	10000		


Рис. Б.19 — Список виртуальных машин в панели управления KVM

<div> <div>Список ISO-образов</div> <div>Удалить</div> <div>Ctrl + Shift + F</div> </div>				
Id	Имя образа	Время удаления	Размер	Состояние
1	debian-8.0.0-amd64-netinst.iso		246.00 MB	
2	CentOS-7-x86_64-Minimal-1503-01.iso		636.00 MB	

Рис. Б.20 — Список ISO-образов



ГлавнаяСписок зап...Запрос



Запрос

Тема запроса *

Недоступность сайта test.ru?

Услуга

Host A #80?

Отдел

☒ Technical?

☐ Sales?

Текст сообщения *

Здравствуйте.
Сайт test.ru недоступен, пожалуйста найдите причину его неработоспособности.
Спасибо.

Прикрепить файл

Screenshot.pngВыберите файл?

Прикрепить файл

Выберите файл?

Отправить

Отмена

Рис. Б.21 — Запрос к технической поддержке

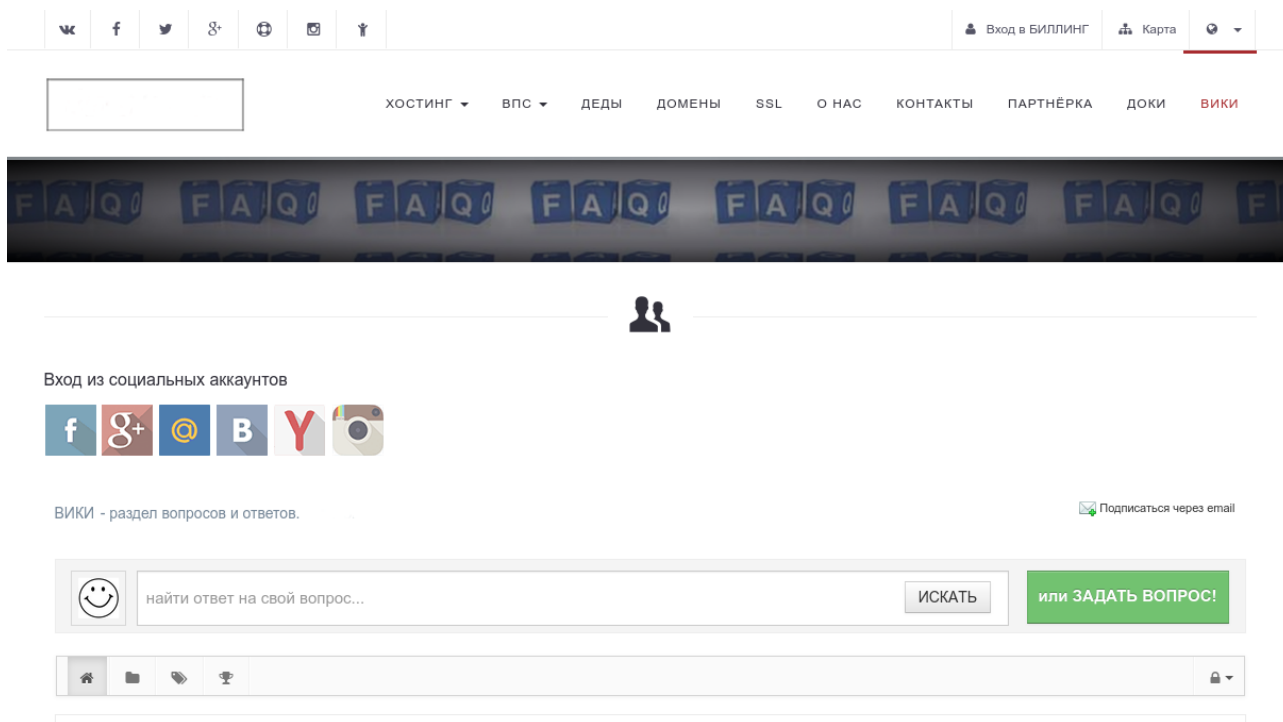


Рис. Б.22 — Регистрация в wiki

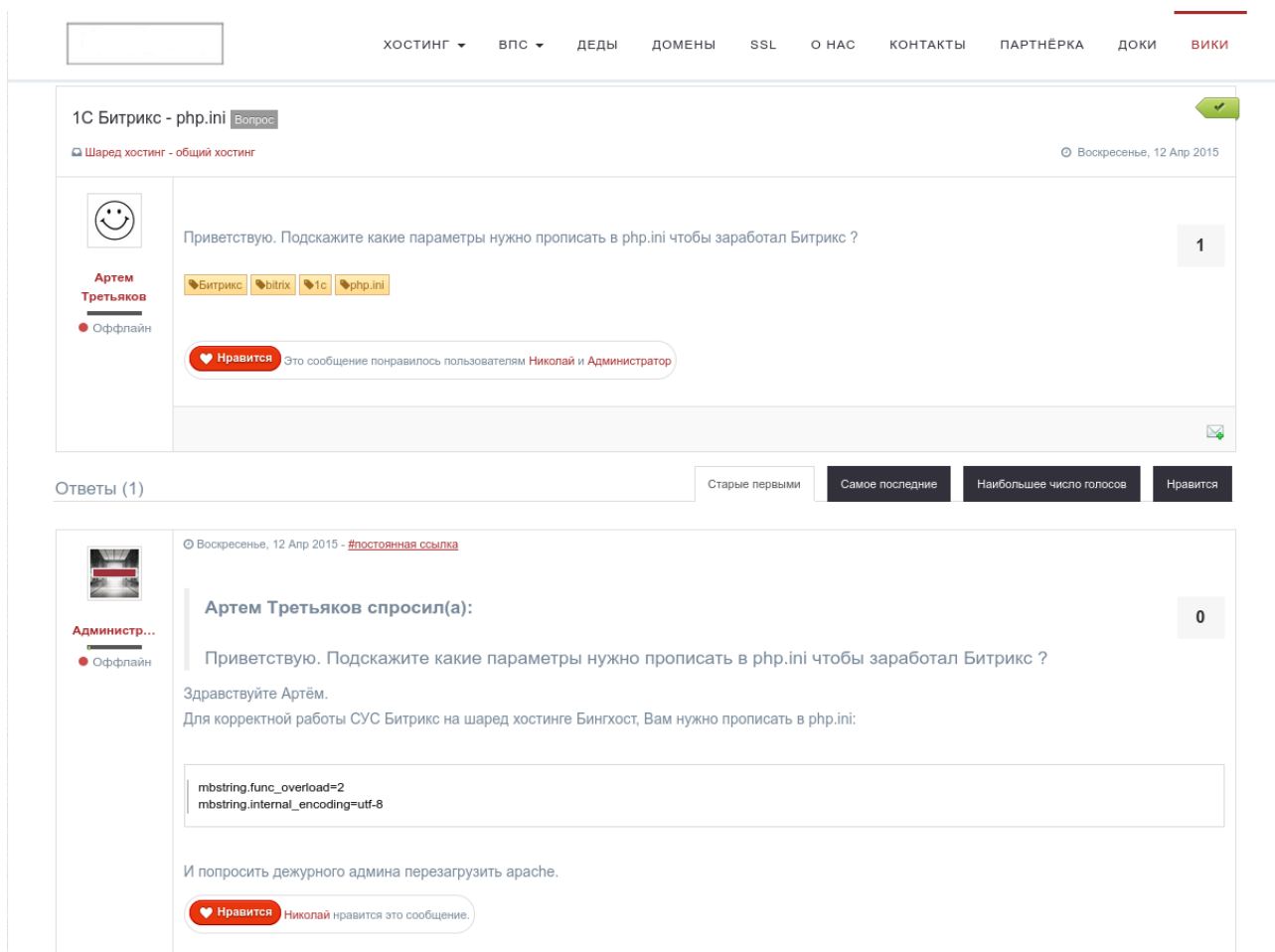


Рис. Б.23 — Пример вопроса в wiki

«УТВЕРЖДАЮ»

Генеральный директор Ильин А.А.

«09» апреля 2015г.

АКТ

внедрения дипломной работы бакалавра, студента кафедры Информационных технологий и компьютерных систем Севастопольского государственного университета Умерова Амета Ремзиевича, на тему «Разработка виртуальной инфраструктуры для реализации облачных услуг», выполненной по специальности 09.03.01 — информатика и вычислительная техника.

Я, ниже подписавшийся, генеральный директор ООО «Информационные технологии» Ильин Антон Александрович, составил настоящий акт о том, что результаты дипломной работы бакалавра Умерова А.Р. использованы в прикладных разработках по созданию виртуальной инфраструктуры, для реализации облачных услуг в ООО «Информационные Технологии». В частности:

- использована свободная технология виртуализации OpenVZ;
- организован мониторинг и система резервного копирования контейнеров клиентов на базе свободного ПО;
- внедрена схема использования свободной панели VestaCP в клиентских контейнерах.

Данный акт не является основанием для взаимных финансовых расчетов.

(генеральный директор)

Ильин Антон Александрович