Cloud of Science. 2014. Т. 1. № 4
http://cloudofscience.ru
ISSN 2409-031X

Проект компьютерной сети на основе технологии Content Delivery Network

Д. Г. Радчук[1, 2], Е. В. Никульчев[1]

[1]Московский технологический институт
119334, Москва, Ленинский проспект, 38а
[2]Cisco Systems Poland
30-707, Poland, Krakow, Powstancow Wielkopolskich str., 13e
e-mail: nikulchev@mail.ru, lliopt@gmail.com

Аннотация. Приведены результаты проектирования компьютерной сети, обепечивающей повышение скорости доставки контента в Интернете за счет использования технологии Content Delivery Network (CDN). Эта технология разработана для географически распределенной сетевой инфраструктуры с целью повышения эффективности доставки контента конечным пользователям в сети Интернет. Использование контент-провайдерами CDN способствует увеличению скорости загрузки интернет-пользователями аудио, видео и других видов цифрового контента в точках присутствия сети CDN.

В статье приводятся настройки оборудования и результаты тестирования на экспериментальном стенде.

Ключевые слова: Content Delivery Network, проектирование сетей, балансировка загрузки.

1. Введение

Традиционная сеть Интернет — это международная компьютерная сеть, которая направлена на обеспечение доступности ресурсов находящейся в ней, используя стек протоколов TCP/IP. Оптимизация и снижение трафика в его изначальные функции не входит (рис. 1) [1].

CDN (Content Delivery Network) — географически распределенная сетевая инфраструктура, которая накладывается на сеть Интернет (рис. 2), и направлена на надежную доставку контента пользователям, а также на снижение трафика в сетях операторов связи (CDN-операторов) [2]. При использовании сети CDN данные центрального сервера ресурса копируются на региональные кэш-сервера. Каждый кэш-сервер поддерживает в обновленном состоянии полную или частичную копию часто запрашиваемых данных. Маршрутизатор, находящийся рядом с кэш-сервером, взаимодействует с локальными сетями интернет-провайдером и распространяет контент конечным пользователям по кратчайшему сетевому маршруту с оптимального по загруженности сервера.

Одной из характеристикой доставки данных является время между отправкой запроса и получением ответа (RTT, от англ. *Round Trip Time*), позволяет определять

двусторонние задержки по маршруту и частоту потери пакетов, то есть косвенно определять загруженность на каналах передачи данных и промежуточных устройствах (см. рис. 1 и 2).

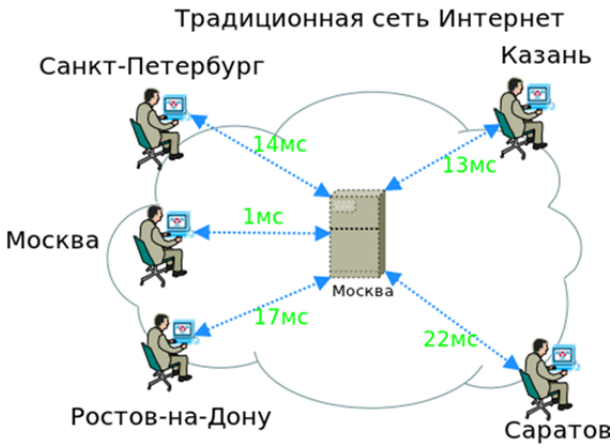

Рисунок 1. Традиционная сеть Интернет и RTT до контента

Кэширование является одним из самых распространенных методов реализации CDN решения, в связи с тем, что предлагает оптимальное использование дискового пространства и каналов передачи данных. При этом самые большие временные затраты загрузки файла берет на себя пользователь, который первый обратился на оригинальный сервер контент-провайдера. Если данных нет на локальном кэш-сервере, то они запрашиваются с центрального сервера и пополняют кэш. Все последующие пользователи будут брать файл из ближайшего к ним кэш-сервера. Таким образом, на региональных кэш-серверах хранится только популярный и наиболее часто запрашиваемый контент.

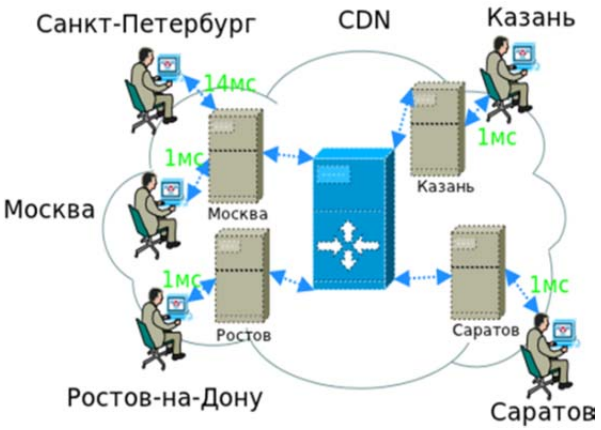

Рисунок 2. Сеть CDN и RTT до контента

Крупные CDN провайдеры могут состоять из огромнейшего числа узлов и размещать свои сервера непосредственно в сети каждого локального интернет-провайдера. Это очень выгодно провайдерам, т. к. они не платят за трафик, а CDN-провайдер получает выгоду за счет быстрой отдачи контента с кэш-серверов.

Многие CDN-операторы опираются на пропускную способность каналов связи и минимально необходимом количестве точек обмена трафика в регионе. При этом главным назначением таких сетей является увеличение скорости передачи как статического контента, так и непрерывного потока данных.

По данным аналитики от компании 5coins, если у пользователя не загрузилась страница в течение 4 секунд, то 25% пользователей уходит в веб-страницы.

На скорость загрузки страницы влияют характеристики канала передачи данных от пользователя до сервера, а также географическое удаление сервера от пользователя. Также, чем меньше географически расстояние до сервера, тем меньше вероятности того, что возникнут аварии у операторов связи. Плановые работы или аварии на сети операторов норма для эксплуатации сети.

Также не редки случаи, когда оператор связи не может быстро локализировать место проблемы или работы по восстановлению оптического волокна затягиваются. Технология CDN способна снизить задержки при передаче данных, зависимость от аварий на каналах связи, а также потери на перегруженных каналах у операторов. Применение технологий управления нагрузкой при передаче трафика позволяет распределять нагрузку между разными узлами сети.

Локальное к конечным пользователям размещение кэш-серверов может увеличить исходящую пропускную способность всей системы. Современные инфраструктуры могут производить автоматический контроль целостности данных на всех серверах в пути следования пакета. При этом гарантируется практически 100%-ая доступность контента для пользователя в случае аварии канала передачи данных между узлами сети, или любого из серверов в сети.

Рынок CDN в России еще молодой. На западе он существует уже более десяти лет. Первый CDN-оператор — компания NGENIX в России появился в 2008 году, вслед за ним подключилась к гонке компания CDNVideo. Эти компании не имеют своей магистральной сети и каналы связи, до ближайшей точки присутствия к пользователю, арендуют у операторов связи.

Операторы связи: Мегафон, МТС, Ростелеком — строят свои сети доставки контента и в дальнейшем вытеснят CDN-операторов, не имеющих своих магистральных каналов.

Целью работы является разработка сети с технологией CDN для уменьшения времени загрузки веб-страниц пользователям, находящимся на больших расстояниях от центров обработки данных. Использование CDN позволило бы повысить отказоустойчивость архитектуры в целом.

В качестве поставленной задачи, проектируемая сеть должна обеспечить:

– коэффициент надежности $K_{г} = 0,99$ при $t_{в} \leq 30$ минут;

- для 90% пользователей сокращение RTT для пакетов объемом от 64 до 1500 байт, не менее чем на 20% по сравнению с традиционной сетью;
- для 90% пользователей сокращение времени доставки статического контента, не менее чем на 20%, при 2% потерь пакетов и RTT 150 мс, по сравнению с традиционной сетью;
- время принятия решения по балансировке пакетов объемом от 64 до 1500 байт, на балансировщике не более чем 1 мс за 1 с.;
- обработку на балансировщике не менее 80 000 пакетов в секунду, объемом от 64 до 1500 байт.

2. Протоколы и стандарты используемые в CDN

Стандарт IEEE 802.3ab Ethernet. (базируется на IEEE 802.3) позволяет работать сетевым устройствам на скорости 1Gbit/s с поддержкой технологии IEEE 802.1Q VLAN. Стандарт позволяет работать с витой парой категорий 5e. В передаче данных участвуют все 4 пары. Скорость передачи данных — 250 Мбит/с по одной паре. Используется метод кодирования PAM5, частота основной гармоники 62,5 МГц. Максимальное расстояние кабеля 100 метров.

Стандарт IEEE 802.3ae Ethernet включает в себя семь стандартов физической среды для LAN, MAN и WAN, который обеспечивает скорость 10Gb/s. Физическая среда — оптический кабель. При проектировании мы будем использовать стандарт 10GBASE-ER и одномодовое оптическое волокно, способное работать на расстоянии до 40 км.

Стандарт IEEE 802.1Q VLAN. Virtual Local Area Network — логическая локальная компьютерная сеть, представляет собой группу хостов с общим набором требований, которые взаимодействуют так, как если бы они были подключены к широковещательному домену, независимо от их физического местонахождения. VLAN имеет те же свойства, что и физическая локальная сеть, но позволяет конечным станциям группироваться вместе, даже если они не находятся в одной физической сети. Такая реорганизация может быть сделана на основе программного обеспечения вместо физического перемещения устройств.

При проектировании мы будем настраивать теги на порты, чтобы логически их объединить в один сегмент.

*Технология NAT (*Network Address Translation) — это механизм в сетях TCP/IP, позволяющий преобразовывать IP-адреса транзитных пакетов. Самый популярный режим его работы NAT/PAT. Терминология NAT описана в RFC 2663.

Есть 3 вида трансляции: статическая, динамическая, перегруженная.

Статический NAT — отображение незарегистрированного IP-адреса на зарегистрированный IP-адрес на основании один к одному. Особенно полезно, когда устройство должно быть доступным снаружи сети.

Динамический NAT — отображает незарегистрированный IP-адрес на зарегистрированный адрес от группы зарегистрированных IP-адресов. Динамический NAT также устанавливает непосредственное отображение между незарегистриро-

ванным и зарегистрированным адресом, но отображение может меняться в зависимости от зарегистрированного адреса, доступного в пуле адресов, во время коммуникации.

Перегруженный NAT (NAPT, NAT Overload, PAT, маскарадинг) — форма динамического NAT, который отображает несколько незарегистрированных адресов в единственный зарегистрированный IP-адрес, используя различные порты. Известен так же, как и PAT. При перегрузке каждый компьютер в частной сети транслируется в тот же самый адрес, но с различным номером порта.

Протокол ARP — протокол сетевого уровня, предназначенный для определения MAC-адреса по известному IP-адресу. Используется для того, чтобы узнать, как собрать кадр и отправить в пункт назначения. Описан в RFC 826.

Протокол IP. С 1983 года является основным протоколом для доставки пакетов, объединяет их в сегменты сети в единую сеть, обеспечивая доставку данных между любыми узлами сети. Он классифицируется как протокол третьего уровня по сетевой модели OSI. IP не гарантирует надежной доставки пакета до адресата. В частности, пакеты могут прийти не в том порядке, в котором были отправлены, продублироваться (приходят две копии одного пакета), оказаться поврежденными (обычно поврежденные пакеты уничтожаются) или не прийти вовсе. Гарантию безошибочной доставки пакетов дают некоторые протоколы более высокого уровня — транспортного уровня сетевой модели OSI, например, TCP, которые используют IP в качестве транспорта.

Протокол OSPF (Open Shortest Path First) — протокол динамической маршрутизации IP трафика, основанный на технологии отслеживания состояния канала (link-state technology) и использующий для нахождения кратчайшего пути Алгоритм Дейкстры (Dijkstra’s algorithm). Протокол OSPF был разработан IETF в 1988 году. Последняя версия протокола представлена в RFC 2328. Протокол OSPF представляет собой протокол внутреннего шлюза (Interior Gateway Protocol — IGP). Протокол OSPF распространяет информацию о доступных маршрутах между маршрутизаторами одной автономной системы.

OSPF предлагает решение следующих задач [3]:

- увеличение скорости сходимости;
- поддержка сетевых масок переменной длины (VLSM);
- достижимость сети (быстро обнаруживаются отказавшие маршрутизаторы, и топология сети изменяется соответствующим образом);
- оптимальное использование пропускной способности (т. к. строится минимальный остовный граф по алгоритму Дейкстры).

Маршрутизаторы обмениваются hello-пакетами через все интерфейсы, на которых активирован OSPF. Маршрутизаторы, разделяющие общий канал передачи данных, становятся соседями, когда они приходят к договоренности об определенных параметрах, указанных в их hello-пакетах.

На следующем этапе работы протокола маршрутизаторы будут пытаться перейти в состояние смежности с маршрутизаторами, находящимися с ним в преде-

лах прямой связи (на расстоянии одного хопа). Переход в состояние смежности определяется типом маршрутизаторов, обменивающихся hello-пакетами, и типом сети, по которой передаются hello-пакеты. OSPF определяет несколько типов сетей и несколько типов маршрутизаторов. Пара маршрутизаторов, находящихся в состоянии смежности, синхронизирует между собой базу данных состояния каналов.

Каждый маршрутизатор посылает объявление о состоянии канала маршрутизаторам, с которыми он находится в состоянии смежности.

Каждый маршрутизатор, получивший объявление от смежного маршрутизатора, записывает передаваемую в нем информацию в базу данных состояния каналов маршрутизатора и рассылает копию объявления всем другим смежным с ним маршрутизаторам.

Рассылая объявления через зону, все маршрутизаторы строят идентичную базу данных состояния каналов маршрутизатора.

Когда база данных построена, каждый маршрутизатор использует алгоритм «кратчайший путь первым» для вычисления графа без петель, который будет описывать кратчайший путь к каждому известному пункту назначения с собой в качестве корня. Этот граф — дерево кратчайших путей.

Каждый маршрутизатор строит таблицу маршрутизации из своего дерева кратчайших путей.

Типы пакетов OSPF:

- Hello — формирование и поддержание смежности между маршрутизаторами, передаются каждые 10 сек в multiaccess и point-to-point сетях, 30 сек в NBMA;
- DBD — Database Description, краткое содержание базы link-state, принимающий проверят свою базу, по умолчанию 4 hello, multiaccess и point-to-point — 40 сек, 120 сек в NBMA;
- LSR — Link-State Request, принимающий может запросить более подробную информацию;
- LSU — ответ на LSR, анонсирование новой информации (LSU содержит 7 типов LSA), LSA (Link State Advertisement) — содержит информацию о соседях и стоимости пути до них;
- LSACK — подтверждение приема LSU.

В данном проекте мы будем использовать OSPF для связи балансировщика и маршрутизатора.

Протокол BGP предназначен для обмена информацией о достижимости подсетей между автономными системами (AS), т. е. группами маршрутизаторов под единым техническим управлением, использующими протокол внутридоменной маршрутизации для определения маршрутов внутри себя и протокол междоменной маршрутизации для определения маршрутов доставки пакетов в другие AS [4]. Передаваемая информация включает в себя список AS, к которым имеется доступ через данную систему. Выбор наилучших маршрутов осуществляет исходя из правил, принятых в сети.

BGP поддерживает бесклассовую адресацию и использует суммирование маршрутов для уменьшения таблиц маршрутизации. С 1994 года действует четвертая версия протокола, все предыдущие версии являются устаревшими.

BGP, наряду с DNS, является одним из главных механизмов, обеспечивающих функционирование Интернета.

BGP является протоколом прикладного уровня и функционирует поверх протокола транспортного уровня TCP (порт 179) [5]. После установки соединения передается информация обо всех маршрутах, предназначенных для экспорта. В дальнейшем передается только информация об изменениях в таблицах маршрутизации. При закрытии соединения удаляются все маршруты, информация о которых передана противоположной стороной.

Механизм выбора лучшего маршрута:

1. Если путь, указанный в поле next-hop[1], недоступен, этот update[2] удаляется.
2. Лучшим считается путь с наибольшим значением weight[3].
3. Если weight путей одинаковы, то выбирается путь с наибольшим localpref.
4. Если weight и local pref[4]одинаковы, то выбирается путь, который ориджинировался в таблице маршрутов BGP на данном роутере. (подразумевается, что это будет самый наикратчайший путь).
5. Если нет путей, которые «родились» на данном маршрутизаторе, то выбирается путь с as-path[5] минимальной длины.
6. Если все пути имеют одинаковую длину (одинаковое количество hop'ов, следовательно), выбирается путь с «lowest origin type[6]». Считается, что IGP < EGP < Incomplete[7].
7. Если Origin attr. code[8] одинаковы для всех путей, выбирается путь с наименьшим MED attribute[9].
8. Если и MED одинаковы для всех возможных путей, предпочтительным считается external path[10] перед internal path[11].
9. Если опять все пути одинаковы, выбирается тот, у которо метрика IGP до next-hop[12] меньше.
10. Если включена команда maximum-paths и есть несколько eBGP маршрутов, пришедших от соседов из одной AS, то инсталлируются все последние маршруты, но не больше *n*. Самый старый маршрут помечается как лучший (хотя при тесте

[1] Next-hop (англ.) — следующий прыжок.
[2] Update (англ.) — обновление.
[3] Weight (англ.) — вес.
[4] Local pfef (англ.) — локальный префикс.
[5] As-path (англ.) — путь до автономной системы.
[6] Lowest origin type (англ.) — тип наименьшего зарождения (маршрута).
[7] Incomplete (англ.) — незавершенный.
[8] Origin attribute code (англ.) — код атрибута зарождения (маршрута).
[9] Attribute (англ.) — атрибут.
[10] External path (англ.) — внешний маршрут.
[11] Internal path (англ.) — внутренний маршрут.
[12] Next-hop (англ.) — следующий прыжок.

выбирался маршрут с наименьшим RID) и только он анонсируется дальше, причем при анонсе по iBGP делается next-hop-self. Для iBGP маршрутов должна использоваться команда maximum-paths ibgp n.

Чтобы балансировать маршруты, полученные из разных AS, надо ввести магическую потаенную команду bgp bestpath[13] as-path multipath-relax[14].

11. Выбирается путь, пришедший от маршрутизатора с наименьшим BGP RID.

12. Если RID одинаковый, то выбирается маршрут с более коротким Cluster list[15].

13. В самом последнем случае выбирается маршрут, пришедший от соседа с наименьшим peer[16]-адресом.

В нашей сети используется для обмена маршрутами между кэш-серверов, маршрутизатором во Владивостоке и Москве.

3. TCP опции, алгоритмы и расчеты плавающего окна

Процесс установки соединения TCP состоит из 3 шагов [6].

1. Клиент, который намеревается установить соединение, посылает серверу сегмент с номером последовательности и флагом SYN. Сервер получает сегмент, запоминает номер последовательности и пытается создать сокет (буферы и управляющие структуры памяти) для обслуживания нового клиента. В случае успеха сервер посылает клиенту сегмент с номером последовательности и флагами SYN и ACK, и переходит в состояние SYN-RECEIVED (принятая синхронизация). В случае неудачи сервер посылает клиенту сегмент с флагом RST.

2. Если клиент получает сегмент с флагом SYN, то он запоминает номер последовательности и посылает сегмент с флагом ACK. Если он одновременно получает и флаг ACK (что обычно и происходит), то он переходит в состояние ESTABLISHED. Если клиент получает сегмент с флагом RST, то он прекращает попытки соединиться. Если клиент не получает ответа в течение 10 секунд, то он повторяет процесс соединения заново.

3. Если сервер в состоянии SYN-RECEIVED получает сегмент с флагом ACK, то он переходит в состояние ESTABLISHED.

Протокол TCP функционирует нормально при выполнении ряда условий.

1. Вероятность ошибки доставки (BER) невелика и потеря пакета вероятнее всего происходит из-за переполнения буфера. Если потеря пакета из-за его искажения существенна, понижение CWND не поможет, и пакеты будут теряться с той же вероятностью (здесь было бы уместно поискать оптимальное значение MTU).

2. Время доставки (RTT) достаточно стабильно и для его оценки можно использовать простые линейные аппроксимации. Здесь подразумевается, что в рамках

[13] Bestpath (англ.) — лучший маршрут.
[14] Multipath-relax (англ.) — ослабленные множественные пути.
[15] Cluster list (англ.) — список кластера.
[16] Peer (англ.) — участник.

сессии все пакеты следуют одним и тем же путем и смена порядка прихода пакетов, хотя и допускается, но маловероятна. Разрешающая способность внутренних часов отправителя должна быть достаточно высока, в противном случае возникают серьезные потери из-за таймаутов.

3. Сеть имеет фиксированную полосу пропускания и, во всяком случае, не допускает скачкообразных ее вариаций. В противном случае потребовался бы механизм для прогнозирования полосы пропускания, а действующие алгоритмы задания CWND оказались бы не эффективными.

4. Буферы сетевых устройств используют схему первый вошел-первым вышел (FIFO). Предполагается, что размер этих буферов соответствует произведению RTT×B (B — полоса пропускания, RTT — сумма времен транспортировки сегмента от отправителя к получателю и времени движения отклика от получателя к отправителю). Если последнее условие нарушено, пропускная способность неизбежно понизится и будет определяться размером буфера, а не полосой пропускания канала.

5. Длительность TCP-сессии больше нескольких RTT, чтобы оправдать используемую протокольную избыточность. Короткие TCP-сессии, широко используемые WEB-технологией, снижают эффективность обмена. (Именно это обстоятельство вынудило в версиях HTTPv1.1 и выше не разрывать TCP-соединение после вызова очередной страницы).

Чтобы минимизировать влияние избыточности, связанной с заголовком (20 байт IP +20 байт TCP + 14 байт MAC-заголовок), используемое поле данных должно иметь большой объем. Для узкополосных каналов, где MTU мало, нарушение данного требования делает канал низкоэффективным. По этой причине выявление допустимого MTU в начале сессии должно приветствоваться.

Взаимодействие с другими TCP-сессиями не должно быть разрушительным, приводящим к резкому снижению эффективности виртуального канала.

В основных версиях протокола TCP для подавления перегрузки используется механизм окон, который управляется потерей пакетов. В современных сетях передача данных, которая создает стационарный трафик, сосуществует с мультимедиа трафиком, который по своей природе нестабилен, что создает дополнительные проблемы для управления с применением оконных алгоритмов. Кроме того, мультимедийный трафик транспортируется обычно протоколом UDP, не предполагающим подтверждений доставки. Именно дейтограммы UDP могут вызвать переполнение буфера и об этом станет известно отправителю TCP позднее после регистрации потери сегмента.

При выборе балансировщика, как устройства, которое будет балансировать трафик между контрольно-измерительным оборудованием и контент-сервером, мы ориентировались на готовые решения в области открытого программного обеспечения. Эти решения:

– *HAProxy* — свободное, очень быстрое и надежное решение, предлагающее высокую производительность при балансировании, и proxying для TCP вплодь до HTTP трафика. Это особенно хорошо для веб-сайтов, работаю-

ших при очень высоких нагрузках, нуждающихся в постоянном контроле Layer7 запросов и ответов. Поддерживает десятки тысяч подключений. При этом его интеграция в существующую систему происходит очень легко.

- *Nginx* — простой, быстрый и надежный сервер, не перегруженный функциями. Применение nginx целесообразно прежде всего для статических вебсайтов и как прокси-сервера перед динамическими сайтами. Умеет акселерированное проксирование без кэширования, простое распределение нагрузки и отказоустойчивость.
- *ipvsadm* — модулья ядра Linux умеющий перехватывать сообщения на четвертом уровне OSI распределять между серверами, при этом не терминиует на себе TCP сессии.

Из всех решений только ipvsadm подходит под наши технические требования.

4. Расчет надежности сети

Надежность восстанавливаемых систем, а наша система относится именно к таким, характеризуется рядом показателей: среднее время безотказной работы, коэффициент готовности и среднее время восстановления системы после сбоев [7].

Надежность — свойство ВС сохранять во времени в установленных пределах значения всех параметров, характеризующих способность выполнять требуемые функции в заданных режимах и условиях применения, технического обслуживания, ремонтов, хранения.

Краткие характеристики показателей надежности:

- среднее время безотказной работы — T_0, это время, при котором сохраняется работоспособность системы;
- коэффициент готовности — $K_Г$, его значение определяет, какова вероятность того, что в произвольный момент времени t система находится в состоянии работоспособности (кроме планируемых периодов, в течение которых применение объекта по назначению не предусматривается);
- время восстановления — $t_в$, это время, затрачиваемое на восстановление работоспособности системы нарушенной вследствие возникшего сбоя, путем ремонта, состоящего в выявлении причины нарушения работоспособности и восстановлении работоспособности путем замены или ремонта неисправного элемента.

Будем считать, что CDN считается неработоспособной в случае нарушения связанности с: контент-сервером, балансировщиком, коммутатором, оптики между коммутаторами, кэш-сервером.

Далее, составим табл. 1, где, для перечисленных типов оборудования, на основе усредненных данных фирм-изготовителей, будут приведены: среднее время безотказной работы T_0, интенсивность потока отказов λ_{Oi}, величина обратная T_0 — время необходимое на восстановление.

Таблица 1. Характеристики надежности компонентов СДК

Оборудование	Среднее время безотказной работы, T_0, ч	Интенсивность потока отказов, λ_{Oi} 1/ч	Время восстановления, $t_в$, ч	Коэффициент готовности, $K_{гi}$
Кэш-сервер	80000	$1.25 \cdot 10^{-5}$	0.2500	$1–3.125 \cdot 10^{-6}$
Маршрутизатор	35000	$2.86 \cdot 10^{-5}$	0.1667	$1–4.768 \cdot 10^{-6}$
Коммутатор	120000	$0.83 \cdot 10^{-5}$	0.3333	$1–2.766 \cdot 10^{-6}$
Соед. оптика	30000	$3.33 \cdot 10^{-5}$	0.4167	$1–13.876 \cdot 10^{-6}$
Балансировщик	50000	$2.00 \cdot 10^{-5}$	1.0000	$1–20 \cdot 10^{-6}$
Контрольно-измерительный сервер	40000	$1.9 \cdot 10^{-5}$	0.8884	$1–18 \cdot 10^{-6}$
Соед.витая пара	20000	$3.21 \cdot 10^{-5}$	0.4033	$1–12.876 \cdot 10^{-6}$

Примечания.

1. Группа «соединения оптики между коммутаторами» — общая характеристика надежности оптиков, разъемов, патч-панели, с учетом их количества;

2. Среднее время безотказной работы и интенсивность потока отказов указаны для работы оборудования в нормальных условиях.

3. Время восстановления берется исходя из опыта работы и указано для случая, когда запасные части и комплектующие (ЗиП) для ведения мелкого ремонта, а также жизненно важные узлы для горячей замены (дополнительный коммутатор, комплектующие для ремонта контент-сервера, кэш-сервера, балансировщика) есть в наличии.

Построив граф надежности (рис. 3), видим, что на нем имеется участок с последовательным соединением.

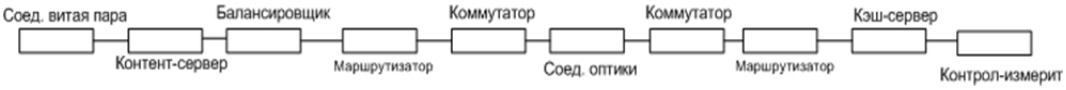

Рисунок 3. Граф надежности

Соотношения для последовательного соединения восстанавливаемых подсистем ПО.

1. Интенсивность отказов:

$$\lambda = \sum_{i=1}^{n} \lambda_i, \quad [1/ч],$$

где λ_i — интенсивность отказов i-й подсистемы; n — количество подсистем.

Для рассматриваемого случая:

$$\lambda = (\lambda_{\text{соед.вит.пара}} + \lambda_{\text{балансировщик}} + 2\lambda_{\text{маршрутизатор}} + 2\lambda_{\text{коммутатор}} + \\ + \lambda_{\text{соед.оптики}} + \lambda_{\text{кэш-сервер}} + \lambda_{\text{кон.изм.}}), \quad [1/ч];$$

$$\lambda = (1.9 + 2 + 2 \cdot 2.86 + 2 \cdot 0.83 + 3.33 + 1.25 + 3.21) \cdot 10^{-5} = 19.07 \cdot 10^{-5}, \ [1/ч].$$

2. Среднее время безотказной работы:

$$T_0 = \frac{1}{\lambda} = \frac{1}{19.07} \cdot 10^{-5} = 5243, \quad [ч].$$

3. Коэффициент готовности:

$$K_\Gamma = \sum_{i=1}^{n} (1 - K_{\Gamma i}),$$

где $K_{\Gamma i}$ — интенсивность отказов *i*-й подсистемы; *n* — количество подсистем.
Вычислим:

$$K_\Gamma = 1 - (18 + 20 + 2 \cdot 4.768 + 2 \cdot 2.766 + 13.876 + 3.125 + 12.876) \cdot 10^{-6} = 1 - 82.94 \cdot 10^{-6}.$$

4. Интенсивность восстановления:

$$\mu = \frac{\lambda}{1 - K_\Gamma} = \frac{19.07 \cdot 10^{-5}}{1 - (1 - 82.94 \cdot \le 10^{-6})} = 2.229, \quad [1/ч];$$

5. Среднее время восстановления системы:

$$t_В = \frac{1}{\mu} = \frac{1}{2.229} = 0.37\,[ч] = 25.8 \ [мин].$$

Производительность и функциональность АПК балансировщика приведена в табл. 2.

Таблица 2. Производительность полностью балансировщика на базе ipvs

	1998	2014
Процессор	Intel Pentium 200MHz	Intel Xeon E5645 2.4GHz
Метод балансировки	NAT	NAT
Пропускная способность	70 Mbit/s	5 Gbit/s
Количество обработанных пакетов в секунду	~ 15K pps	~1M pps
Обработка пакета на балансировщике	60 ms	< 1 ms

Функциональность и масштабируемость АПК балансировщика [8]:

- поддерживает режим балансировки IPIP;
- поддерживает режим балансировки Direct Routing;
- поддерживает режим балансировки NAT;
- не терминирует на себе TCP, а, что важно, экономит ресурсы и делает передачу данных быстрее.

При росте трафика балансировщик может не справиться с нагрузкой и в таком случае решением является горизонтальное масштабирование — добавление еще одного балансировщика. Они могут анонсировать через OSPF один из тот же префикс. Механизм распределения нагрузки заложен в протокол OSPF и называется equal cost path. Принцип простой, если маршрутизатору приходят 2 префикса с

одинаковой метрикой, то он устанавливает их оба в таблицу маршрутизации, передавая трафик по обоим каналам по очереди. Количество добавляемых маршрутов с одинаковой стоимостью зависит лишь только от аппаратных возможностей маршрутизатора, на данный момент наиболее часто можно увидеть реализацию из максимум 16 маршрутов с одинаковой стоимостью.

4. Проектирование сети

Структурная схема СДК изображена на рис. 4.

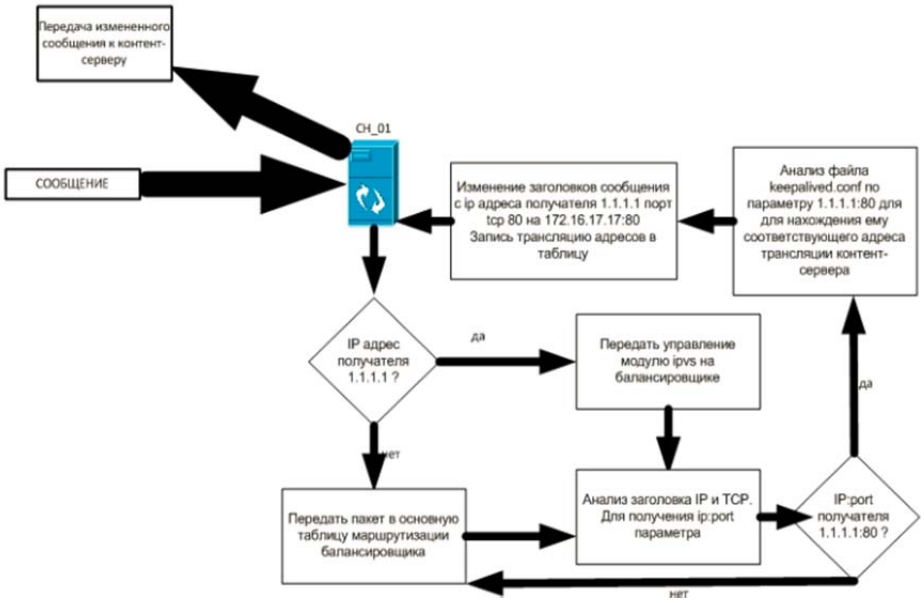

Рисунок 4. Структурная схема СДК

Настройка трех контент-серверов отличается лишь сетевой конфигурацией.

В качестве аппаратного сервера мы выберем Supermicro SYS-6017R-NTF , т. к. он подходит для процессоров XEONE5-2660, которые показали хорошие показатели при нагрузочном тестировании E5-2660 (2.2 GHz, 8 cores, 20 MB cache, 95W TDP) — 256 Gflops.

Операционная система Ubuntu с предварительно настроенным сетевым стеком. В качестве программного обеспечения сервера будет выступать nginx.

Настройка сервера для статики:

```
sudo apt-get install nginx
```

После установки необходимо изменить настройки файла nginx.conf — главный конфигурационный файл, собственно с него и начинается загрузка:

```
user root root;
worker_processes  4;
timer_resolution 100ms;
worker_priority -5;
#error_log  /var/log/nginx/error.log;
error_log off;
pid        /var/run/nginx.pid;
events {
# accept_mutex on;
# accept_mutex_delay 500ms;
# worker_aio_requests 32
use epoll;
worker_connections  4096;           #~ 1G RAM
}
http {
                                    # Подключение mimetypes
include /etc/nginx/mime.types;
                                      # Подключение прокси
include /etc/nginx/proxy_params;
                                # Не показывать информацию о сервере
server_tokens off;
                                                # Логи доступа
#access_log      /var/log/nginx/access.log;
access_log off;
                                    # Протокол отдачи статики
sendfile        on;
tcp_nodelay on;
                                  # Подключение других настроек
include /etc/nginx/conf.d/*.conf;
                                 # Подключение виртуальных хостов
include /etc/nginx/sites-enabled/*;
location ~* ^.+\.(jpg|jpeg|gif|png)$ {
        root /path/to/site.ru;
        expires 10d;
        error_page 404 =404 /pics/empty.gif;
}
expires 10d;                    - nginx будет отдавать заголовки кеширования Expires и Cache-
Control (в примере кеш ставится на 10 дней).
error_page 404 =404 /pics/empty.gif;
```

Если nginx не находит запрошенный файл, то он отдаст 404-ю http ошибку сервера и указанный empty.gif, который физически должен быть размещен по указанному пути (в данной работе это — однопиксельный .gif).

Также можно сделать , чтобы nginx обращался к back-end за ненайденным статическим файлом:

```
          location ~* ^.+\.(jpg|jpeg|gif|png)$ {
        root /path/to/site.ru;
        expires 10d;
```

```
                error_page 404 = @img_not_found;
}
location @img_not_found{
        proxy_pass http://site.ru:8080;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Real-IP $remote_addr;
    }
настройка сети для RS_01:
ip -4 addr add 172.16.17.17/24 dev eth1
ip -4 route add 172.16.17.254 dev eth1 — терминируется на маршрутизаторе RT_02
настройка сети для RS_02:
ip -4 addr add 172.16.17.18/24 dev eth1
ip -4 route add 172.16.17.254 dev eth1 — терминируется на маршрутизаторе RT_02
настройка сети для RS_03:
ip -4 addr add 172.16.17.19/24 dev eth1
ip -4 route add 172.16.17.254 dev eth1 — терминируется на маршрутизаторе RT_02
Настройка TCP:
В файле /etc/sysctl.conf вносим изменения:
sysctl net.ipv4.tcp_available_congestion_control
sysctl -w net.ipv4.tcp_congestion_control=yeah
```

Настройка контрольно-измерительного сервера. Операционная система на базе Linux Ubuntu и установленные пакеты joe, curl для нагрузочного тестирования. Контент сервера с ip 1.1.1.1, на котором работает сервис HTTP и отдающий по запросу статический контент. Зайти на него можно по ссылке:

http://1.1.1.1:80/index.html

Контрольно-измерительное оборудование будет заходить на сервер и смотреть с какой задержкой сервер отдал ему статический контент.

Ниже приведено регулярное выражение, которое запускается на контрольно-измерительном сервере:

```
for i in `jot 100 1`; do curl --connect-timeout 10 --compressed --user-agent "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/535.2 (KHTML, like Gecko) Chrome/15.0.874.106 Safari/535.2 YE" -o /dev/null -L -w
'%{url_effective}\t%{http_code}\t%{num_redirects}\t%{time_namelookup}\t%{time_connect}\t%{time_pretransfer}\t%{time_redirect}\t%{time_starttransfer}\t%{time_total}\t%{size_download}\n' -s 'http://1.1.1.1:80/inxex.hmtl' ; sleep 1; done
Настройка сети:
ifconfig 172.16.12.12 eth0
route add 0.0.0.0/0 gw 172.16.12.254
```

Настройка маршрутизаторов. В сети есть два маршрутизатора на FreeBSD. На них настроен firewall, который, при прохождении пакетов от контрольно-измерительного сервера к контент-серверу, создает задержку в 150 мс. Это необходимо для воспроизведения задержки на канале Москва-Владивосток.

На маршрутизаторе RT_01:

```
ipfw pipe 3 config bw 100000kbit/s delay 75 plr 0.01
 ipfw pipe show 4
00004:  10.000 Mbit/s   60 ms burst 0
q131076  50 sl. 0 flows (1 buckets) sched 65540 weight 0 lmax 0 pri 0 droptail
 sched 65540 type FIFO flags 0x0 0 buckets 0 active

 sudo ipfw list 1-20
00001 pipe 4 ip from any to 1.1.1.0/24 out
00002 skipto 50 ip from any to me
00004 skipto 50 ip from me to any
00006 skipto 50 ip from me6 to any
00008 skipto 50 ip from any to me6
00009 allow ip from any to 1.1.1.0/24
00009 allow ip from 1.1.1.0/24 to any
00010 skipto 500 ip from any to any
На маршрутизаторе RT_02:
ipfw pipe 3 config bw 100000kbit/s delay 75 plr 0.01
ipfw pipe list 3
00003:   1.000 Mbit/s   60 ms burst 0
q131075  50 sl. 0 flows (1 buckets) sched 65539 weight 0 lmax 0 pri 0 droptail
 sched 65539 type FIFO flags 0x0 0 buckets 1 active
  0 ip           0.0.0.0/0            0.0.0.0/0        2      184  0    0   0

 ipfw list
00001 pipe 3 ip from any to any xmit bce0
00100 allow ip from any to any via lo0
00200 deny ip from any to 127.0.0.0/8
00300 deny ip from 127.0.0.0/8 to any
65000 allow ip from any to any
65535 deny ip from any to any
```

Также на маршрутизаторе RT_01 работает quagga с настроенным протоколом BGP для поддержании в актуальном состоянии таблицы маршрутизации до кэш-сервера и маршрутизатора в Москве.

```
router bgp 777
 bgp router-id 10.0.0.1
 bgp log-neighbor-changes
 bgp deterministic-med
 bgp dampening
 redistribute connected route-map CDN
neighbor 10.0.0.2 remote-as 777
 neighbor 10.0.0.2 peer-group CDN
 neighbor 10.0.0.2 description CDN BGP hook
 neighbor 10.0.0.2 prefix-list PFXS-CDN-BGP-37.140.136.1 in
neighbor 172.16.17.17 remote-as 65204
 neighbor  172.16.17.17 peer-group CDN_CACHE
 neighbor 172.16.17.17 description CDN_CACHE
```

```
 neighbor 10.0.0.2 prefix-list PFXS-CDN-BGP-37.140.136.1 in
 route-map CDN  permit 10
 match ip address prefix-list CDN_SUBNETS
 set community 13238:999 13238:1104
ip prefix-list CDN_SUBNETS seq 50 1.1.1.0/24
 route-map CDN  permit 20
match ip address prefix-list CDN_CACHE
ip prefix-list CDN_CACHE seq 50 172.16.17.0/24
```

Настройка кэш-сервера. Устанавливаем операционную систему Linux Ubuntu. Устанавливаем пакеты squid, quagga.

Настройка squid.

Редактируем файл /usr/local/squid/etc/squid.conf.

Установим объем памяти, доступный squid, и каталог для кэша.

```
cache_mem 65536
cache_dir /usr/local/squid/cache 1024 16 256
```

Здесь 1024 — количество мегабайт для кэша.

Укажем хосты, с которых разрешен доступ к прокси

```
acl allowed_hosts src 192.16.0.0/255.240.255.0
acl localhost src 127.0.0.1/255.255.255.255
```

разрешенные HTTP порты:

```
acl HTTP_port 80
```

запретим метод CONNECT для всех портов, кроме указанных в acl SSL_ports:

```
http_access deny CONNECT !HTTP_ports
```

и запретим доступ всем, кроме тех, кому можно:

```
http_access allow localhost
http_access allow allowed_hosts
http_access allow HTTP_ports http_access deny all
```

пропишем пользователей, которым разрешено пользоваться squid (den, admin, developer):

```
ident_lookup on
acl allowed_users user den admin developer
http_access allow allowed_users
http_access deny all
```

Тэги maxium_object_size и maxium_object устанавливают ограничения на размер передаваемых объектов.

Настройка TCP:

В файле /etc/sysctl.conf вносим изменения:

```
sysctl net.ipv4.tcp_available_congestion_control
sysctl -w net.ipv4.tcp_congestion_control=yeah
```

Настройка балансировщика.

Устанавливаем операционную систему Linux Ubuntu. Устанавливаем пакеты ipvsadm, keepalived, quagga:

```
sudo apt-get install ipvsadm keepalived quagga
```

Настройка keepalived.conf — отвечает за балансировку до контент серверов.

```
virtual_server 172.16.17.17, 172.16.17.18, 172.16.17.19  80 {
```

```
protocol TCP
quorum_up   "/etc/keepalived/quorum-handler2.sh up   1.1.1.1, 80 e1-100,1"
quorum_down "/etc/keepalived/quorum-handler2.sh down 1.1.1.1, 80  e1-100,1"
        alpha
        omega
        lvs_method NAT
        quorum 1
        hysteresis 0
        lvs_sched wlc
        delay_loop 8
        real_server 172.16.17.17 80 {
                HTTP_GET {
                        url {
                                path /ping
                                status_code 200
                        }
                        connect_port 80
                        bindto 192.168.0.1
                        connect_timeout 1
                        nb_get_retry 1
                        delay_before_retry 1
                }
        }
        real_server 172.16.17.18 80 {
                HTTP_GET {
                        url {
                                path /ping
                                status_code 200
                        }
                        connect_port 80
                        bindto 192.168.0.1
                        connect_timeout 1
                        nb_get_retry 1
                        delay_before_retry 1
                }
        }
        real_server 172.16.17.19 80 {
                HTTP_GET {
                        url {
                                path /ping
                                status_code 200
                        }
                        connect_port 80
                        bindto 192.168.0.1
                        connect_timeout 1
                        nb_get_retry 1
                        delay_before_retry 1
                }
        }
}
```

Настройка коммутаторов. В данном проекте используется коммутационное оборудование компании Huawei, это обусловлено только лишь его меньшей ценой (текст приведен в приложении 1).

Стенд проектируемой СДК. На контрольно-измерительном оборудовании запускается скрипт для нагрузки контент-серверов. Скрипт заходит на страницу http://1.1.1.1:80/index.html и скачивает ее 100 раз.

Нагрузочное тестирование проектируемой сети без технологии CDN при RTT 150 мс.

Нагрузочное тестирование проектируемой сети без технологии CDN при RTT 150 мс приведено в табл. 3 и на рис. 5, 6. Здесь: time_pretransfer — время согласования до отправки данных; time_starttransfer — время начала передачи; time_total — время полной загрузки страницы в секундах; size_download - объем веб-страницы в Килобайтах.

Таблица 3. Результаты тестовых испытаний

Вид текста	Среднее время получения файла, мс	time_ pretransfer	time_ redirect	time_ starttransfer	time_ total	size_ download
Нагрузочное тестирование без технологии CDN при RTT 150 мс	648	0.12457	0.00151	0.09597	0.64881	34218.07
Нагрузочное тестирование без технологии CDN при RTT 150 мс и 2% потерь пакетов	989	0.00151	0.00152	0.09597	0.98929	34218.07
Нагрузочное тестирование с применением технологии CDN и RTT 150 мс	384	0.00151	0.00152	0.09597	0.98929	34218.07
Нагрузочное тестирование с применением технологии CDN и RTT 150 мс и 2 % потерь пакетов	408	0.0015	0.0015	0.096	0.4086	34218

Заметим, что из-за особенностей протоколов [9] в трафике наблюдается динамический хаос [10]. Управление такими процессами требует использования специальных алгоритмов, направленных на использование этих свойств [11].

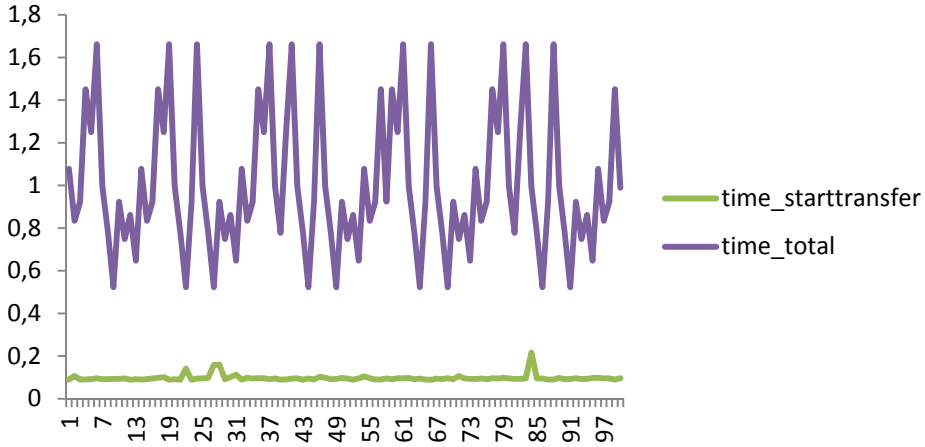

Рисунок 5. Результаты тестирования на экспериментальном стнде без технологии CDN при RTT 150 мс

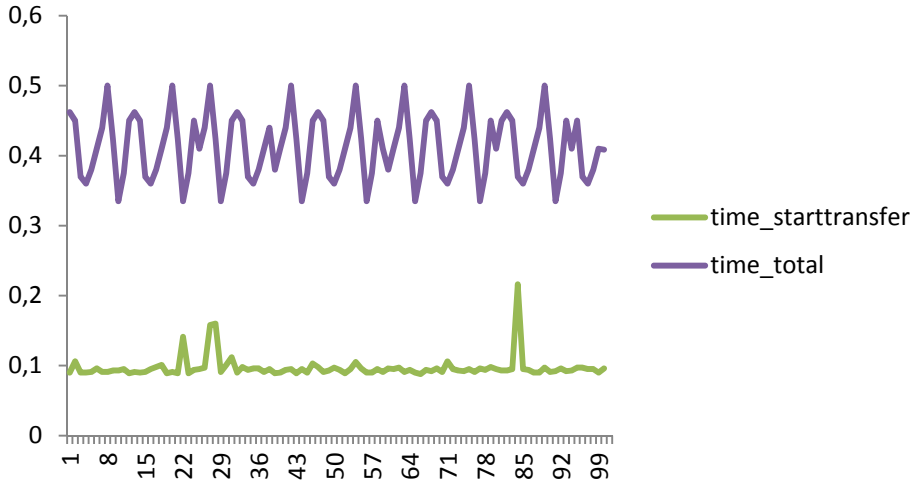

Рисунок 6. Результаты тестирования на экспериментальном стенде без технологии CDN при RTT 150 мс

6. Заключение

В работе построена сеть CLN на открытой архитектуре, и доступном оборудовании, что снимает зависимость от каких то определенных вендоров. Проведенный расчет надежности обеспечивает быструю замену компонентов. Балансировка работает на программном обеспечении с открытым исходным кодом и легко масшабируется при внедрении практически в любую архитектуру.

Внедрение усовершествованных механизмов регулирования TCP окна позволяет оптимальным образом использовать канал, при RTT 150 мс эффективность выше на 59%, а при RTT 150 мс и 2% потерь, эффективность канала выше на 41%, по сравнению с традиционной сетью Интернет и использованием стандартного механизма TCP.

Литература

[1] *Fortino G., Palau C. E.* Next generation content delivery infrastructures: emerging paradigms and technologies. — IGI Global, 2012.

[2] *Maillé P., Tuffin B.* How Do Content Delivery Networks Affect the Economy of the Internet and the Network Neutrality Debate? // Economics of Grids, Clouds, Systems, and Services. — Springer, 2014. C. 222–230.

[3] *Parkhurst W. R.* Cisco OSPF command and configuration handbook. — Cisco Press, 2002.

[4] *Parkhurst W. R.* Cisco BGP–4 command and configuration handbook. — Cisco Press, 2001.

[5] *Shamim F.* (ed.). Troubleshooting IP routing protocols. — Cisco Press, 2002.

[6] *Doyle J., Carroll J. D. H.* Routing TCP/IP. Vol. I and II. — Cisco Systems. Inc, 2001

[7] *Xiao X. P.* Technical, commercial and regulatory challenges of QoS: An internet service model perspective. — Morgan Kaufmann, 2008.

[8] *Membrey P., Hows D., Plugge E.* Practical Load Balancing. — New Delhi: Apress, 2012.

[9] *Карпухин А. В.* Особенности реализации протокола TCP в современных компьютерных сетях // Системы обработки информации. 2009. Вып. 6(80). С. 49–53.

[10] *Pluzhnik E., Nikulchev E.* Study of Chaos in the Traffic of Computer Networks // International Journal of Advanced Computer Science and Applications. 2014. Vol. 5. No. 9. P.60–62.

[11] *Никульчев Е. В., Паяин С. В., Плужник Е. В.* Динамическое управление трафиком программно-конфигурируемых сетей в облачной инфраструктуре // Вестник РГРТУ. 2013. № 3. С.54–58.

Авторы:

Радчук Дмитрий Геннадьевич, магистрант Московского технологического института, инженер Cisco Systems Poland

Никульчев Евгений Витальевич, доктор технических наук, профессор, проректор по научной работе Московского технологического института

Приложение 1. Настройка коммутаторов

```
interface Vlanif1
 shutdown
#
interface Vlanif555
 ip address 172.16.20.20 255.255.255.0
#
interface MEth0/0/1
 shutdown
#
interface GigabitEthernet0/0/1
 description C_01
 port link-type default vlan 444
 port trunk allow-pass vlan 1
 port link-type access
#
interface GigabitEthernet0/0/2
 description CH_01
 port link-type access
 port default vlan 800
#
interface GigabitEthernet0/0/3
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/4
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/5
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/6
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/7
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/8
description \
```

```
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/9
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/10
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/11
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/12
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/13
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/14
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/15
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/16
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/17
 description \
 shutdown
 port link-type access
```

```
 port default vlan 999
#
interface GigabitEthernet0/0/18
 ddescription \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/19
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/20
  description RT_01
 port link-type trunk
 undo port trunk allow-pass vlan 1
 port trunk allow-pass vlan 444 555 666 777 888
#
interface GigabitEthernet0/0/21
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/22
  port media type fiber
  undo negotiation auto
 combo-port copper
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/23
 port media type fiber
  undo negotiation auto
 combo-port copper
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/24
 port media type fiber
  undo negotiation auto
 combo-port auto
 port link-type access
 port default vlan 999
#
interface XGigabitEthernet0/1/1
  shutdown
 port link-type access
 port default vlan 999
#
```

```
interface XGigabitEthernet0/1/2
shutdown
 port link-type access
 port default vlan 999
#
interface XGigabitEthernet0/1/3
 description SW_02 xg0/1/3
 port link-type trunk
 undo port trunk allow-pass vlan 1
 port trunk allow-pass vlan 111
#
interface XGigabitEthernet0/1/4
 description SW_02 xg0/1/4
 port link-type trunk
 undo port trunk allow-pass vlan 1
 port trunk allow-pass vlan 287 297 317 601 659 901 2006 2612 3202
#
interface NULL0
#
ip route-static 0.0.0.0 0.0.0.0 172.16.20.20.254
stelnet server enable
ssh authentication-type default password
#
command-privilege level 0 view shell display elabel
command-privilege level 0 view shell display version
command-privilege level 0 view shell display patch-information
command-privilege level 0 view shell undo terminal monitor
command-privilege level 0 view shell undo terminal trapping
user-interface maximum-vty 10
user-interface con 0
 authentication-mode password
 user privilege level 0
 set authentication password cipher %$%$25353423>SbQ%'FrwIE4t}3|5,#zY.^;AUz\F|>Ze:+})A5B;2-
%$%$
 idle-timeout 0 0
 screen-length 25
user-interface vty 0 4
 authentication-mode aaa
 user privilege level 4
 screen-length 50
 protocol inbound all
user-interface vty 5 9
 authentication-mode aaa
 screen-length 50
 protocol inbound all
user-interface vty 16 20
```

Конфигурация коммутатора SW_02:

```
interface Vlanif1
 shutdown
#
interface Vlanif555
 ip address 172.16.21.21  255.255.255.0
#
interface MEth0/0/1
 shutdown
```

```
#
interface GigabitEthernet0/0/1
 description RS_01
 port link-type access
 port default vlan 666
#
interface GigabitEthernet0/0/2
 description RS_02
 port link-type access
 port default vlan 666
#
interface GigabitEthernet0/0/3
description RS_03
 port link-type access
 port default vlan 666
#
interface GigabitEthernet0/0/4
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/5
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/6
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/7
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/8
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/9
description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/10
 description LB_01
 port link-type access
 port default vlan 678
```

```
#
interface GigabitEthernet0/0/11
 description LB_01
 port link-type access
 port default vlan 678
#
interface GigabitEthernet0/0/12
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/13
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/14
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/15
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/16
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/17
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/18
 ddescription \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/19
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/20
  description RT_02
```

```
 port link-type trunk
 undo port trunk allow-pass vlan 1
 port trunk allow-pass vlan 444 555 666 678 777 888
#
interface GigabitEthernet0/0/21
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/22
  port media type fiber
  undo negotiation auto
 combo-port copper
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/23
 port media type fiber
  undo negotiation auto
 combo-port copper
 description \
 shutdown
 port link-type access
 port default vlan 999
#
interface GigabitEthernet0/0/24
 port media type fiber
  undo negotiation auto
 combo-port auto
 port link-type access
 port default vlan 999
#
interface XGigabitEthernet0/1/1
  shutdown
 port link-type access
 port default vlan 999
#
interface XGigabitEthernet0/1/2
shutdown
 port link-type access
 port default vlan 999
#
interface XGigabitEthernet0/1/3
 description SW_01 xg0/1/3
 port link-type trunk
 undo port trunk allow-pass vlan 1
 port trunk allow-pass vlan 111
#
interface XGigabitEthernet0/1/4
 description SW_01 xg0/1/4
 port link-type trunk
 undo port trunk allow-pass vlan 1
 port trunk allow-pass vlan 444 555 666 777 888
```

```
#
interface NULL0
#
ip route-static 0.0.0.0 0.0.0.0 172.16.21.21.254
stelnet server enable
ssh authentication-type default password
#
command-privilege level 0 view shell display elabel
command-privilege level 0 view shell display version
command-privilege level 0 view shell display patch-information
command-privilege level 0 view shell undo terminal monitor
command-privilege level 0 view shell undo terminal trapping
user-interface maximum-vty 10
user-interface con 0
 authentication-mode password
 user privilege level 0
 set authentication password cipher %$%$25353423>SbQ%'FrwIE4t}3|5,#zY.^;AUz\F|>Ze:+})A5B;2-
%$%$
 idle-timeout 0 0
 screen-length 25
user-interface vty 0 4
 authentication-mode aaa
 user privilege level 4
 screen-length 50
 protocol inbound all
user-interface vty 5 9
 authentication-mode aaa
 screen-length 50
 protocol inbound all
```

Content Delivery Network Computer Network

Dmitry Radchuk[1, 2], *Evgeny Nikulchev*[1]

[1]*Moscow Technologycal Institute*
119334, Leninskii prospect 38 a, Moscow, Russia
[2]*Cisco Systems Poland*
30-707, Powstancow Wielkopolskich str., 13e, Krakow, Poland
e-mail: nikulchev@mail.ru, lliopt@gmail.com

Abstract. The results of the design Content Delivery Network (CDN) computer network, providing increased speed of delivery of content on the Internet. This technology was developed for geographically distributed network infrastructure. Using content provider CDN increases the download speed of the Internet users audio, video, and other types of digital content in the presence of a CDN. The article describes the hardware setup and the results of testing on a test stand.
Keywords: Content Delivery Network, Network Design, Load Balancing.

Reference

[1] *Fortino G., Palau C. E.* (2012) Next generation content delivery infrastructures: emerging paradigms and technologies. IGI Global.

[2] *Maillé P., Tuffin B.* (2014) How Do Content Delivery Networks Affect the Economy of the Internet and the Network Neutrality Debate? *Economics of Grids, Clouds, Systems, and Services*. Springer. P. 222–230.

[3] *Parkhurst W. R.* (2002) Cisco OSPF command and configuration handbook.

[4] *Parkhurst W. R.* (2001) Cisco BGP–4 command and configuration handbook..

[5] *Shamim F.* (ed.). (2002) Troubleshooting IP routing protocols. Cisco Press.

[6] *Doyle J., Carroll J. D. H.* (2001) Routing TCP/IP. Vol. I and II. Cisco Systems. Inc.

[7] *Xiao X. P.* (2008) Technical, commercial and regulatory challenges of QoS: An internet service model perspective. Morgan Kaufmann.

[8] *Membrey P., Hows D., Plugge E.* (2012) Practical Load Balancing. New Delhi.

[9] *Karpuhin A. V.* (2009) Osobennosti realizacii protokola TCP v sovremennyh komp'juternyh setjah. Sistemy obrabotki informacii, 6(80), 49–53. (In Rus)

[10] *Pluzhnik E., Nikulchev E.* (2014) Study of Chaos in the Traffic of Computer Networks. *International Journal of Advanced Computer Science and Applications*, 5(9), 60–62.

[11] *Nikulchev E. V., Payain S. V., Pluzhnik E. V.* (2013) Dinamicheskoe upravlenie trafikom programmno-konfiguriruemyh setej v oblachnoj infrastructure. Vestnik RGRTU, 3, 54–58. (In Rus)