УДК 33 **DOI: 10.34670/AR.2022.64.71.021**

Эффективная модель хранения данных предприятия

Мишина Светлана Викторовна

Старший преподаватель,
Елецкий государственный университет им. И.А. Бунина,
399770, Российская Федерация, Елец, ул. Коммунаров, 28;
e-mail: svmishina2017@mail.ru

Аннотация

Внедрение программного обеспечения для автоматизации хозяйственной деятельности предприятии имеет очень большую стоимость. Процесс внедрение таких решений может занимать длительное время. Принятие решения о внедрении информационных технологий должно основываться на предварительном расчете выгод от их эксплуатации и определения экономической эффективности. Особую важность приобретают вопросы по выбору методики по оценке эффективности и рисков от такого внедрения. Многие предприятия рассматривают вопросы переноса в облака своих корпоративных сервисов и приложений. Использование облачных технологий сильно изменяют архитектуру компьютеров, разработку инструментальных средств и программного обеспечения, а также способы хранения и распределения информации. Специфика применения облачной модели в корне отличается от традиционных. В связи с этим и подход к оценке должен быть иным. Применение облачных вычислений позволит бизнесу экономить средства как на закупку программного обеспечения, аппаратной части, так и электропотреблении. Таким образом, сэкономленные средства можно потратить на развитие бизнеса. В работе проведен анализ проблем внедрения и функционирования облачных хранилищ данных на предприятиях и решена актуальная научная задача разработки моделей и методов повышения пропускной способности распределенных телекоммуникационных систем высокодоступных облачных хранилищ данных на основе новых протоколов доступа. В процессе изучения выделены технологические, функциональные и финансовые проблемы внедрения облачных хранилищ данных в организациях. Реализована архитектура системы на основе предложенных и смоделированных методов организации доступа к облачному хранилищу.

Для цитирования в научных исследованиях

Мишина С.В. Эффективная модель хранения данных предприятия // Экономика: вчера, сегодня, завтра. 2022. Том 12. № 9А. С. 370-381. DOI: 10.34670/AR.2022.64.71.021

Ключевые слова

Экономика, организация, облачные хранилища, защита, данные.

Введение

На современном этапе развития информационных систем весьма важную роль для организации стали играть облачные технологии. Огромное количество предприятий рассматривают возможность перехода к облачным технологиям, которые имеют большой потенциал для повышения эффективности без ущерба для производительности.

Однако для того, чтобы реализовать все преимущества и получить наибольшую отдачу от вложенных инвестиций, организации должны принимать во внимание различные проблемы и особенности внедрения облачных технологий, уникальные для каждой конкретной ситуации.

Облачная система хранения данных, или хранения данных как услуга – это абстрактное понятие, которое соответствует системе хранения данных, которую можно администрировать по требованию через специальный интерфейс [1-3]. Этот интерфейс абстрагирует местонахождение системы, так что локальная она либо удаленная, либо гибридная – не имеет значения. Облачные инфраструктуры хранения данных образуют новые архитектуры, которые поддерживают различные уровни обслуживания поверх потенциально большой группы пользователей и географически распределенных накопителей. Важное значение имеет способность клиента контролировать и управлять тем, как хранятся его данные, и связанными с этим расходами. Многочисленные поставщики облачных услуг предлагают средства управления, которые обеспечивают пользователям повышенный контроль над расходами.

Эффективность хранения данных – важная характеристика облачной инфраструктуры хранения, особенно учитывая ее акцент на общую экономию. Чтобы сделать систему хранения более эффективной, нужно хранить больше данных. Общим решением является сокращение объема исходных данных, чтобы они занимали меньше физического пространства. Два способа достижения этой цели: сжатие – упаковка данных путем их кодирования с использованием различных представлений – и дедупликация – исключение всех дубликатов данных [4; 5]. Хотя оба метода полезны, сжатие предполагает обработку (перекодирование данных в инфраструктуру и из нее), а дедупликация – вычисление сигнатур для поиска дубликатов.

Одна из наиболее особенностей облачного хранения данных – способность обеспечить экономию. Это экономия на приобретении накопителей, их энергоснабжении, ремонте, а также на управлении хранением. Если рассматривать облачное хранение с этой точки зрения (включая Service Level Agreement и повышенную эффективность хранения), оно может оказаться выгодным при определенных моделях использования. API (application programming interface) доступа к хранилищу является важнейшим компонентом услуг. Многие приложения требуют доступа к хранилищу услуг с использованием API, который оптимизирован для этой конкретной системы хранения данных, либо на собственном оборудовании или облачным. Так облачная система хранения Amazon S3 API предоставляет разработчикам SDK (software development kit) для .NET и Java, а также библиотеки для дополнительных платформ и языков. Эти интерфейсы обычно используют протоколы веб-сервисов передачи репрезентативного состояния (REST) и/или простой протокол доступа к объектам (SOAP) [6-8].

При рассмотрении архитектуры нужно учитывать ее рабочие параметры. Под ними понимают различные характеристики архитектуры, учитывая стоимость, производительность, возможность удаленного доступа и т.п. Архитектура облачного хранения данных – это прежде всего доставка ресурсов хранения данных по требованию в высоко-масштабируемой и мультиагентной среде. Обобщенно архитектура облачного хранения данных представляет собой внешний интерфейс, который предоставляет API для доступа к накопителям. В

традиционных системах хранения данных это протокол SCSI (Small Computer System Interface), но в облачке появляются новые протоколы. Среди них можно найти внешние протоколы Web-сервисов, файловые протоколы и, даже, более традиционные внешние интерфейсы (Internet SCSI, iSCSI и др.). За внешним интерфейсом располагается уровень промежуточного программного обеспечения – логика хранения данных. Этот уровень реализует ряд функций, таких как репликация данных и сокращение объема данных, по традиционным алгоритмам размещения данных с учетом географического расположения. Наконец, внутренний интерфейс организует физическое хранение данных. Это может быть внутренний протокол, реализующий специфические функции, или традиционный сервер с физическими дисками.

Основная часть

Быстрое увеличение пропускной способности компьютерных сетей позволило разрабатывать многочисленные приложения, которые предусматривают интенсивную обработку данных [Cong et al., 2017]. Эти новые приложения могут выполнять задачи, начиная от массовой передачи данных (SDSS (Sloan Digital Sky Surve) и electronic Very Long Baseline Interferometry), до интерактивных систем высокой пропускной способности (GeoWall) [Benbelgacem, 2020]. Однако, различные программы ставят различные требования к услугам передачи данных [Tong et al., 2013]. Например, приложение GeoWall может отдать предпочтение в плавном изменении скорости передачи данных, тогда как для приложения SDSS желательно, чтобы данные передавались с максимально возможной скоростью в частных сетях [Jiao et al., 2017]. Тем не менее, нынешняя система интернет предназначена для обеспечения поддержки целого множества различного типа приложений. Эта философия дизайна Интернета оказывает большое влияние на развитие транспортных протоколов. Большинство трафика в Интернете формируется за счет потоков TCP (Transmission Control Protocol), но существуют приложения, для которых протокол TCP не обеспечивает достаточного уровня эффективности. В контексте высокопроизводительных вычислений TCP хорошо известен своей низкой эффективностью и справедливым разделением ресурсов в сетях с высокой задержкой пропускной способности [Mishina, Kornienko, 2021; Zhu et al., 2013].

Модификации сетевого стека ядра протокола (например, новые варианты TCP) обычно требуют нескольких лет для стандартизации, внедрения и широкого развертывания. В самом деле, со времен появления протокола TCP, около трех десятилетий назад, только его четыре версии были широко развернуты, а именно Tahoe, Reno, NewReno, и SACK [Tian et al., 2010]. Хотя на сегодняшний день все больше сетей получают скорость передачи данных 1 Гбит/с и выше, но по-прежнему актуальной проблемой для веб-приложений остается использование широкой полосы пропускной способности, из-за ограничения существующих транспортных протоколов сети. Ограничения внедренных сетевых транспортных протоколов является одной из главных причин, по которой так трудно масштабировать приложения с интенсивным использованием сетевых соединений от местных кластеров до глобальных сетей [Zhong et al., 2009].

Протокол управления передачей (TCP) успешно используется в течение десятилетий, как основной протокол транспортного уровня стека сетевых протоколов. Однако в последнее время было показано, что TCP имеет некоторые потери производительности при его использовании для высокоскоростных сетей Wide Area, особенно для географически удаленных сетей. Алгоритм управления перегрузкой Additive increase/multiplicative decrease, который

используется протоколом TCP, является довольно «бедным» в раскрытии доступной полосы пропускной способности и в случае большой потери пакетов в высокопроизводительных сетях с достаточно большими временами задержки [Chen et al., 2014].

Исследователи компьютерных сетей работают над новыми транспортными протоколами и алгоритмами контроля насыщения для поддержки высокоскоростных сетей следующего поколения. Многие работы, в том числе вариантов TCP (FAST, BiC, Scalable, и HighSpeed) и XCP показали более высокую производительность при их моделировании [Song et al., 2018]. Однако практическое использование в реальных приложениях этих протоколов все еще очень ограничено из-за трудностей их реализации, установки и ограничения технического уровня [Kornienko et al., 2021]. Пользователи сети, которым нужно передавать большие массивы данных обычно обращаются к решениям уровня приложений, среди которых очень популярные протоколы на основе UDP (User Datagram Protocol), например SABUL, UDT (Data Transfer Protocol), Tsunami, RBUDP (Reliable Blast UDP), FOBS и GTP [Abbaspour, 2019]. Протоколы на основе UDP обеспечивают гораздо лучшую переносимость и просты при их установке. Однако, несмотря на простоту внедрения протоколов на уровне пользователя достаточно трудно настроить их в ядре, чтобы сделать их максимально эффективными [Kornienko, Mishina, Melnikov, 2021]. Поскольку реализации уровня пользователь не могут изменить код ядра, могут быть дополнительные переключения контекста и копированием участков памяти между уровнем пользователя и уровнем ядра. На высоких скоростях передачи данных эти операции очень чувствительны к загрузке процессора и производительности протокола [Zhou et al., 2019].

Для практического применения разработанных методов необходимо спроектировать и разработать распределенную систему хранения данных. Специфика современных хранилищ данных в переходе их к распределенным системам. В таких условиях необходимым условием является обеспечение резервирования критических узлов сети. Также важным условием является то, чтобы узлы сети владели информацией о наличии других узлов и данных на них. В условиях распределенной работы, необходимым функционалом работы является обеспечение своевременное выявление неработоспособного узла обмена/хранения данных и извлечения их с работы. Также после восстановления работы узла, он должен как можно скорее присоединиться к обмену / сохранению данных. Проектирование такой системы требует предварительного анализа требований и планирования ее внутренних процессов.

Программное обеспечение системы должно обеспечивать выполнение всех функций и иметь средства организации всех требуемых процессов обработки, передачи и хранения данных во всех регламентированных режимах функционирования. Программное обеспечение системы должно быть: универсальным; функционально достаточным; надежным; адаптивным; подходит для модернизации и масштабирования; иметь интуитивно понятный пользовательский интерфейс; защищенным от внешних воздействий; осуществлять документирование всех действий пользователей программного обеспечения. Программное обеспечение должно разрабатываться с применением принципов структурного и модульного программирования. Каждая из задач, которая входит в систему должна быть максимально независимой от других.

Контроль качества программных средств, которые разрабатываются, должен быть обеспечен тестированием и проведением опытной эксплуатации. Архитектура системы должна базироваться на принципах высоконагруженных и отказоустойчивых систем. Она должна отвечать следующим основным требованиям: система должна поддерживать горизонтальное масштабирование серверов для увеличения производительности системы в целом (как узлов для хранения данных, так и сателлитов для улучшения эффективности доступа к данным); система

должна иметь возможность дублирования всех критически важных узлов сети и хранения данных для обеспечения отказоустойчивости системы.

В состав системы должны входить следующие функциональные компоненты:

– Хранилище данных как компонент информационной системы, который обеспечивает хранение данных для решения следующих задач: хранение данных; доступа к данным; учет действий пользователей.
– Сателлит-компонент информационной системы, который обеспечивает быстрое информационное и техническое взаимодействие системы с пользователями системы.
– Адаптивный DNS (Domain Name System) сервер – компонент системы, который обеспечивает логистику запросов от пользователя к системе.

Общими требованиями к надежности системы:

– Программно-технический комплекс должен функционировать круглосуточно, в непрерывном режиме, кроме форс-мажорных обстоятельств.
– Должно проводиться регулярное (не реже одного раза в сутки) резервное копирование баз данных. Необходимо наличие как минимум двух резервных копий всех данных. Резервные копии должны храниться в физически удаленных местах.
– Отказы и сбои в работе рабочих станций и сетевых устройств не должны приводить к разрушению данных и сказываться на работоспособности системы в целом.
– Выход из строя одной из подсистем не должен приводить к прекращению функционирования других подсистем, то есть при этом должна обеспечиваться возможность выполнения функций всех других подсистем.
– Плановая остановка или сбой информационного ресурса не должны приводить к сбою в работе программного обеспечения.
– Неправильные действия пользователей не должны приводить к возникновению аварийной ситуации.
– Должны быть минимизированы ошибки технического персонала, в том числе путем четкого разграничения прав доступа к системе, а также ведение журнала событий системы.

Под надежностью информационной системы понимают комплексное свойство системы сохранять во времени их основные свойства системы определены в установленных нормативно-технических документах. При таком понимании программное обеспечение должно: быть устойчивым к ошибочным действиям пользователя (ошибки в действиях персонала не должны приводить к сбоям (отказам) в работе программного обеспечения информационной системы); обеспечивать гарантированный контроль входящей и исходящей информации; обеспечивать быстрое восстановление после отказа (сбоев).

Программное обеспечение разрабатывается на основе распространенных операционных систем, инструментальных средств программирования и СУБД (система управления базами данных). Система должна использовать стандартные решения, основанные на применении типовых протоколов и интерфейсов взаимодействия, которые предусматривают возможность сопряжения и совместной работы оборудования и программного обеспечения различных производителей, а также для сопряжения с информационными системами других организаций. Все технические решения, используемые в проекте системы, должны соответствовать требованиям национальных стандартов или (при отсутствии) международных стандартов. Технические средства, применяемые в составе ИС, должны иметь сертификаты или другие

документы предприятия-поставщика, подтверждающие их соответствие техническим условиям.

В силу большой социальной значимости проекта, и сжатых сроков ввода в эксплуатацию предпочтение следует отдавать унифицированным решением. Такие решения должны иметь следующие свойства: доступ к системе должен предоставляться с помощью глобальной сети Internet; модульность (компонентное решение); иметь возможность интеграции с внешними автоматизированными системами. Система должна обеспечивать выполнение следующих функций: регистрация клиентов системы: о создание облачной записи компании в системе; о создание учетной записи администратора для компании; административная часть: о регистрация пользователей системы для компании; о редактирование профиля пользователя; обеспечение доступа к данным по протоколам: HTTP (HyperText Transfer Protocol); HTTPS (HyperText Transfer Protocol Secure); FTP (File Transfer Protocol); FTPS (File Transfer Protocol + SSL); SFTP (Secure File Transfer Protocol); обработка ошибок: ошибки хранилища; ошибки сателлитов; ошибки сценария; обрывы связи; большая нагрузка; регистрация действий пользователей.

Проанализировав перечень необходимых функций и взаимосвязи между ними, целесообразно выполнить их представление в виде вариантов использования системы (рис. 1, 2). Система должна обеспечивать эффективную организацию обмена информацией между внутренними компонентами. Информационный обмен между компонентами системы должен осуществляться с использованием локальных вычислительных сетей и глобальных сетей передачи данных [Maximov, Ivanov, Sharifullin, 2017; Kornienko, 2020]. Состав, структура, объем и частота передачи сообщений должны определяться соответствующими протоколами информационного обмена, определенными на стадии технического проектирования. В протоколах информационного обмена должны быть предусмотрены меры по исключению возможности несанкционированного доступа к данным.

Также должны быть предусмотрены средства контроля передаваемых входных / выходных данных и средства по контролю информации в базах данных. Требования к информационному обмену между компонентами системы должны быть определены на этапе разработки, исходя из возможностей платформы реализации [Kornienko et al., 2020; Ruan, Zhan, 2014].

Общая архитектура системы состоит из адаптивного DNS сервера, нескольких хранилищ данных (DC) и многих сателлитов (рис. 3). Каждое хранилище данных должно определяться наличием (рис. 4): двух брандмауэров; двух серверов приложений; двух хранилищ с дисками. Такая архитектура необходима для обеспечения отказоустойчивости системы в случае потери физического узла сети [Al Sukhni, Mouftah, 2008; Alexandrovich, 2006]. На первом этапе запросы приходят на уровень брандмауэров, которые для внешних узлов отображаются с одним и тем же IP (Internet Protocol)-адресом с индексами 0 и 1. В случае выхода устройства с индексом 0, то все запросы начинают попадать на устройство с индексом 1.

После получения запроса брандмауэром от пользователя, запрос попадает поочередно на доступный сервер приложений, статус доступности которого постоянно прослеживается на первом уровне. Два хранилища после аппаратного подключения, на аппаратном уровне подключаются как единое хранилище. Также на программном уровне настроена синхронизация данных между хранилищами, для обеспечения резервирования данных. Так, чрезвычайно быстро увеличивается быстродействие процессоров и объем оперативной памяти сервера программы без математически сложных алгоритмов не могут использовать полностью и упираются в количество пользователей, целесообразным является использование контейнеров (виртуализация на уровне операционной системы). Тогда, структурно сервер приложения будет

состоять из нескольких контейнеров и балансировщика нагрузки (рис. 5).

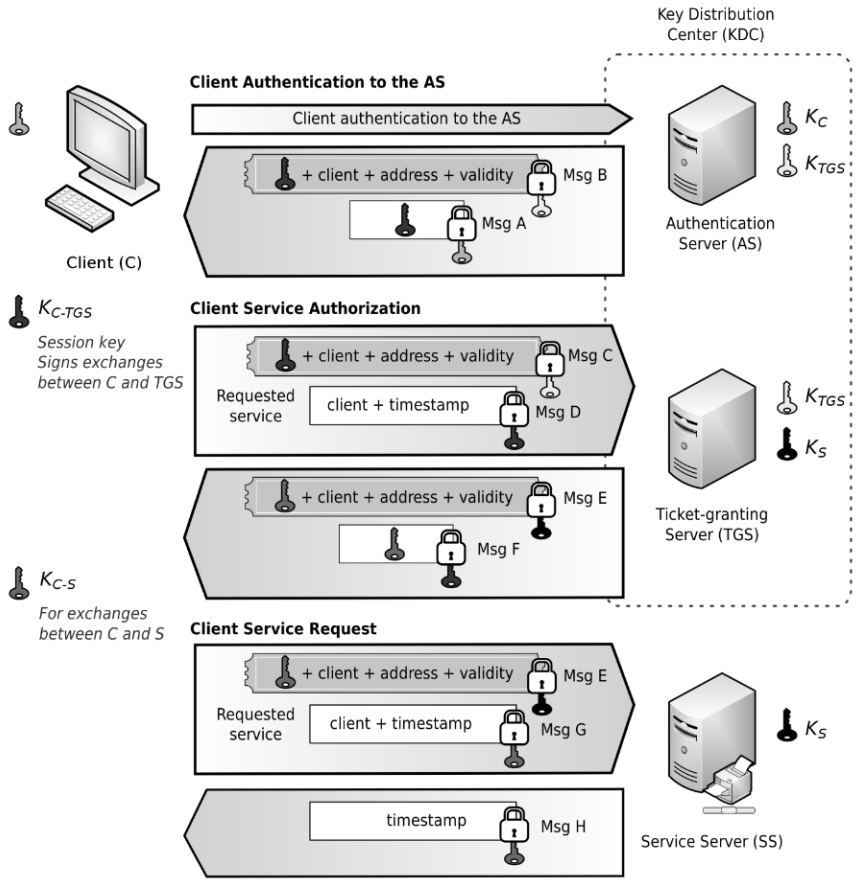

Рисунок 1 - Варианты использования системы с точки зрения Пользователя и Клиента системы

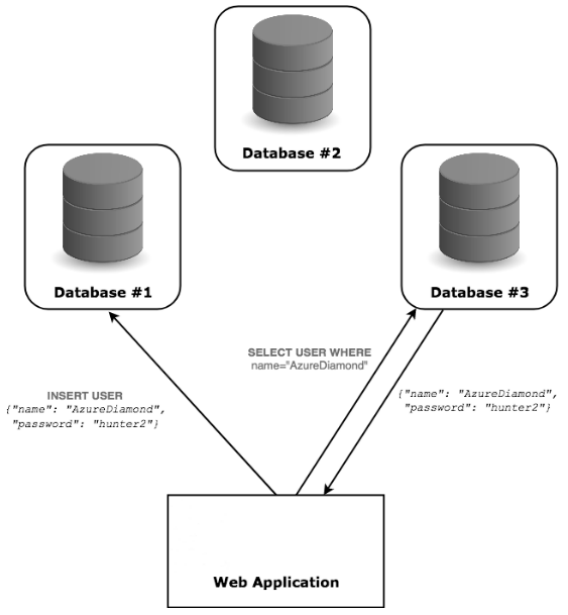

Рисунок 2 - Варианты использования системы с точки зрения Собственника системы

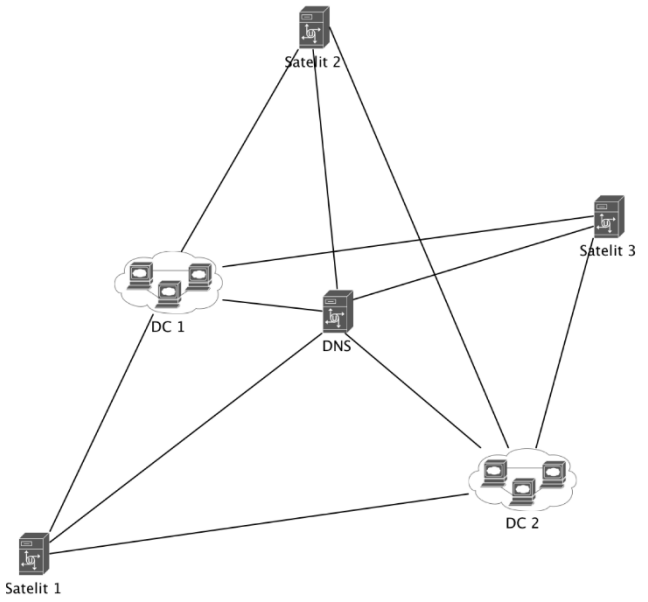

Рисунок 3 - Общая архитектура системы

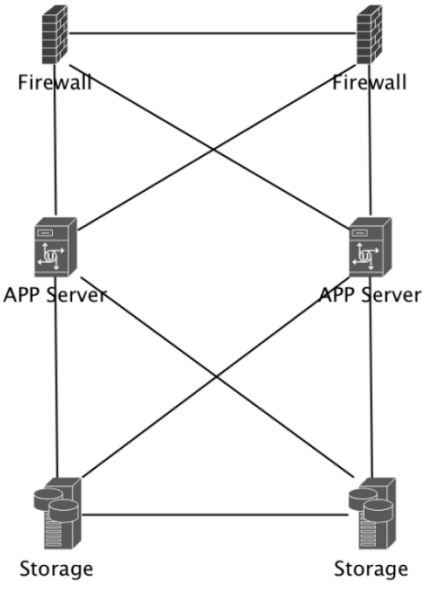

Рисунок 4 - Общая архитектура единичного хранилища

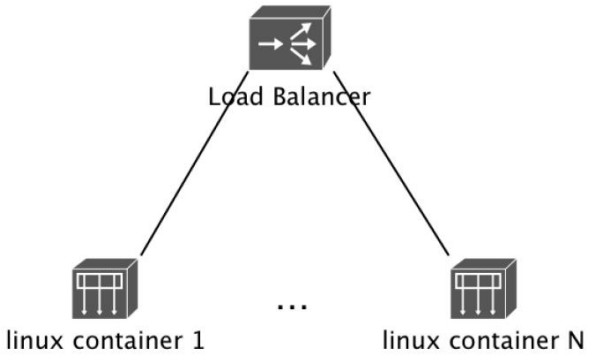

Рисунок 5 - Общая структура сервера приложения

Для мониторинга доступности сервисов в контейнерах существует несколько программных средств, самым популярным из которых является keepalived. Их недостатком при использовании в нашей архитектуре является отсутствие группировки сервисов. Поскольку одним из ключевых требований является возможность доступа к данным по FTP протоколу различных пользователей разных компаний (каждая компания имеет свой hostname), нам нужно для каждой компании иметь свой IP-адрес доступа (специфика использования FTP). FTP-server имеет возможность поддерживать одновременное использование многих IP-адресов. В таком случае keepalived на каждый IP-адрес будет пробовать создавать соединения и проверять доступность, а если их будет достаточное количество, то нагрузки на сервер для проверки доступности контейнеров будут увеличиваться в геометрической прогрессии. Для решения данной проблемы, спроектирована система, которая полностью заменяет функции keepalived, и имеет возможности:

– группировать IP-адреса и сервисы как один сервис (в случае недоступности, данный сервис перестает быть доступен по всем IP-адресам, которые указаны. И наоборот, в случае повторной доступности сервиса, сервисы по всем IP-адресам становятся снова доступны);
– хранение статистики за всеми IP-адресами и сервисам с учетом: о количестве запросов; о передаваемых данных; о полученных данных; о потере работоспособности сервиса; о восстановлении работоспособности сервиса.

Архитектурно Сателлит соответствует архитектуре сервера приложения из хранилища данных. Ему необходимо большое и надежное хранилище данных, поскольку он выступает исключительно «прокси-сервером» между клиентом и его данным. Также у нас нет необходимости его полностью дублировать, поскольку он не несет в себе критических данных и в случае выхода из строя, автоматически запросы, которые шли на него будут идти на ближних с ним другие сателлиты.

В работе современного облачного хранилища данных, которое использует информационные системы, должны быть обеспечены как можно более рациональная организация информационный потоков, так и существенное повышение их интенсивности, то есть ускорение передачи и обработки информации, поступающей от ее источника к потребителю. Для решения этих задач при проектировании информационной системы, прежде всего, проводится анализ информационных потоков.

Заключение

Учитывая затруднения и недостатки существующего состояния передачи данных большого объема через высокопроизводительные регионально-распределенные сети, которые были выявлены в процессе изучения проблемы, был разработан протокол сеансового уровня для таких сетей. Новый протокол сеансового уровня совместим со стандартными протоколами стека протоколов OSI TCP и UDP. Кроме того, он адаптирован для современных сетей, то есть имеет функционал для эффективного использования пропускной способности сети, не зависимо от ее характеристик. При использовании такого протокола не увеличиваются задержки при переходе передачи между различными типами сетей. Тем самым протокол обеспечивает эффективное использование ресурсов крайних узлов сети. В вопросе защищенности он поддерживает шифрование данных, то есть он способен разрешать подключение протоколов шифрования. Была проведена оптимизация эффективности реализации протокола на базе UDP и было

показано, что по этим идеям можно реализовать эффективные и практические приложения на базе протоколов UDP. Например, использование среды протокола UDT (основанный на UDP), может легко поддерживать различные алгоритмы управления перегрузкой, например, высокоскоростные TCP или взрывное RBUDP.

Использование данного подхода позволило увеличить производительность как элемента передачи данных, так и системы в целом. Использование многоканальной связи позволило одновременной передачи данных по одному каналу связи, а универсализация архитектуры позволило использование различных протоколов обмена данных на различных участках сети, в зависимости от эффективности.

Библиография

1. Abbaspour E. et al. A bi-level multi agent-based protection scheme for distribution networks with distributed generation // International Journal of Electrical Power and Energy Systems. 2019. 112. P. 209-220. https://doi.org/10.1016/j.ijepes.2019.05.001
2. Alexandrovich G.P. The applying sets graph protection model to the detection information threats in distributed networks and data base management systems // 8th International Conference on Actual Problems of Electronic Instrument Engineering Proceedings. Piscataway: Institute of Electrical and Electronics Engineers, 2006. P. 126-128. https://doi.org/10.1109/APEIE.2006.4292398
3. Al Sukhni E.M., Mouftah H.T. Availability-guaranteed distributed provisioning framework for differentiated protection services in optical mesh networks // IEEE Globecom Workshops. Piscataway: Institute of Electrical and Electronics Engineers, 2008. https://doi.org/10.1109/GLOCOMW.2008.ECP.46
4. Benbelgacem S. et al. A Distributed Information Retrieval Approach for Copyright Protection // ACM International Conference Proceeding Series. Piscataway: Institute of Electrical and Electronics Engineers, 2020. https://doi.org/10.1145/3386723.3387882
5. Chen X. et al. A coordinated strategy of protection and control based on wide-area information for distribution network with the DG // International Conference on Power System Technology: Towards Green, Efficient and Smart Power System, Proceedings. Piscataway: Institute of Electrical and Electronics Engineers, 2014. P. 2517-2521. https://doi.org/10.1109/POWERCON.2014.6993803
6. Cong W. et al. Distributed Storage and Management Method for Topology Information of Smart Distribution Network // Dianli Xitong Zidonghua/Automation of Electric Power Systems. 2017. 41 (13). P. 111-118. https://doi.org/10.7500/AEPS20161228008
7. Jiao J. et al. Distributed rateless codes with unequal error protection property for space information networks // Entropy. 2017. 19 (1). 38. https://doi.org/10.3390/e19010038
8. Kornienko D.V., Mishina S.V., Melnikov M.O. The Single Page Application architecture when developing secure Web services // Journal of Physics: Conference Series. 2021. 2091 (1). 012065.
9. Kornienko D.V. et al. Principles of securing RESTful API web services developed with python frameworks // Journal of Physics: Conference Series. 2021. 2094 (3). 032016.
10. Kornienko D.V. et al. The effectiveness of the pedagogical conditions for organizing the educational process using distance educational technologies at the university // Journal of Physics: Conference Series. 2020. 1691 (1). 012090.
11. Kornienko D.V. Organization of a system of digital education practices in the municipal sphere of general education // Journal of Physics: Conference Series. 2020. 1691 (1). 012108.
12. Maximov R.V., Ivanov I.I., Sharifullin S.R. Network topology masking in distributed information systems // CEUR Workshop Proceedings. 2017. 2081. P. 83-87.
13. Mishina S.V., Kornienko D.V. Setting up data exchange between information systems that automate accounting at the enterprise // Journal of Physics: Conference Series. 2021. 2094 (3). 032018.
14. Ruan W., Zhan H. A new protection algorithm for distribution network with distributed generation based on intelligent electronic device information // Lecture Notes in Electrical Engineering. 2014. 237 LNEE. P. 275-282. https://doi.org/10.1007/978-3-319-01273-5_30
15. Song X. et al. Active distribution network protection mode based on coordination of distributed and centralized protection // Proceedings of 2017 China International Electrical and Energy Conference. Piscataway: Institute of Electrical and Electronics Engineers, 2018. P. 180-183. https://doi.org/10.1109/CIEEC.2017.8388442
16. Tian J. et al. A fast-current protection scheme for distribution network with distributed generation // IET Conference Publications. Piscataway: Institute of Electrical and Electronics Engineers, 2010. https://doi.org/10.1049/cp.2010.0319
17. Tong B.B. et al. A distributed protection and control scheme for distribution network with DG // Advanced Materials Research. 2013. P. 732-733, 628-633. https://doi.org/10.4028/www.scientific.net/AMR.732-733.628

18. Zhong S. et al. Privacy protection model for distributed service system in converged network // International Conference on E-Business and Information System Security. Piscataway: Institute of Electrical and Electronics Engineers, 2009. https://doi.org/10.1109/EBISS.2009.5138026
19. Zhou C. et al. Principle of Pilot Protection based on Positive Sequence Fault Component in Distribution Networks with Inverter-interfaced Distributed Generators // IEEE PES GTD Grand International Conference and Exposition Asia. Piscataway: Institute of Electrical and Electronics Engineers, 2019. P. 998-1003. https://doi.org/10.1109/GTDAsia.2019.8716011
20. Zhu X. et al. Study on protection scheme for low-voltage distribution network with distributed generation // Information Technology Journal. 2013. 12 (16). P. 3655-3659. https://doi.org/10.3923/itj.2013.3655.3659

Efficient enterprise storage model

Svetlana V. Mishina

Senior Lecturer,
Bunin Yelets State University,
399770, 28, Kommunarov str., Yelets, Russian Federation;
e-mail: svmishina2017@mail.ru

Abstract

The implementation of software for automating business activities of an enterprise is very expensive. The process of implementing such solutions can take a long time. In this regard, the decision to introduce information technologies should be based on a preliminary calculation of the benefits from their operation and the determination of economic efficiency. The issues of choosing a methodology for assessing the effectiveness and risks of such an implementation are of particular importance. Many enterprises are considering migrating their enterprise services and applications to the cloud. The use of cloud technologies greatly changes the architecture of computers, the development of tools and software, as well as the way information is stored and distributed. The specifics of using the cloud model is fundamentally different from traditional ones. The approach to evaluation should be different. The use of cloud computing will allow businesses to save money on both the purchase of software, hardware, and power consumption. Thus, the saved funds can be spent on business development. The paper analyzes the problems of implementation and operation of cloud data storages at enterprises and solves an urgent scientific problem of developing models and methods for increasing the throughput of distributed telecommunication systems of highly available cloud data storages based on new access protocols. In the process of studying, the technological, functional and financial problems of implementing cloud data storages in organizations are highlighted. The architecture of the system is implemented based on the proposed and modeled methods for organizing access to the cloud storage.

For citation

Mishina S.V. (2022) Effektivnaya model' khraneniya dannykh predpriyatiya [Efficient enterprise storage model]. *Ekonomika: vchera, segodnya, zavtra* [Economics: Yesterday, Today and Tomorrow], 12 (9A), pp. 370-381. DOI: 10.34670/AR.2022.64.71.021

Keywords

Economics, organization, cloud storage, protection, data.

References

1. Abbaspour E. et al. (2019) A bi-level multi agent-based protection scheme for distribution networks with distributed generation. *International Journal of Electrical Power and Energy Systems*, 112, pp. 209-220. https://doi.org/10.1016/j.ijepes.2019.05.001
2. Alexandrovich G P. (2006) The applying sets graph protection model to the detection information threats in distributed networks and data base management systems. In: *8th International Conference on Actual Problems of Electronic Instrument Engineering Proceedings*. Piscataway: Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/APEIE.2006.4292398
3. Al Sukhni E.M., Mouftah H.T. (2008) Availability-guaranteed distributed provisioning framework for differentiated protection services in optical mesh networks. In: *IEEE Globecom Workshops.* Piscataway: Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/GLOCOMW.2008.ECP.46
4. Benbelgacem S. et al. (2020) A Distributed Information Retrieval Approach for Copyright Protection. In: *ACM International Conference Proceeding Series*. Piscataway: Institute of Electrical and Electronics Engineers. https://doi.org/10.1145/3386723.3387882
5. Chen X. et al. (2014) A coordinated strategy of protection and control based on wide-area information for distribution network with the DG. In: *International Conference on Power System Technology: Towards Green, Efficient and Smart Power System, Proceedings*. Piscataway: Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/POWERCON.2014.6993803
6. Cong W. et al. (2017) Distributed Storage and Management Method for Topology Information of Smart Distribution Network. *Dianli Xitong Zidonghua/Automation of Electric Power Systems*, 41(13), pp. 111-118. https://doi.org/10.7500/AEPS20161228008
7. Jiao J. et al. (2017) Distributed rateless codes with unequal error protection property for space information networks. *Entropy*, 19 (1), 38. https://doi.org/10.3390/e19010038
8. Kornienko D.V., Mishina S.V., Melnikov M.O. (2021) The Single Page Application architecture when developing secure Web services. *Journal of Physics: Conference Series*, 2091 (1), 012065.
9. Kornienko D.V. et al. (2021) Principles of securing RESTful API web services developed with python frameworks. *Journal of Physics: Conference Series*, 2094 (3), 032016.
10. Kornienko D.V. et al. (2020) The effectiveness of the pedagogical conditions for organizing the educational process using distance educational technologies at the university. *Journal of Physics: Conference Series*, 1691 (1), 012090.
11. Kornienko D.V. (2020) Organization of a system of digital education practices in the municipal sphere of general education. *Journal of Physics: Conference Series*, 1691 (1), 012108.
12. Maximov R.V., Ivanov I.I., Sharifullin S.R. (2017) Network topology masking in distributed information systems. *CEUR Workshop Proceedings*, 2081, pp. 83-87.
13. Mishina S.V., Kornienko D.V. (2021) Setting up data exchange between information systems that automate accounting at the enterprise. *Journal of Physics: Conference Series*, 2094 (3), 032018.
14. Ruan W., Zhan H. (2014) A new protection algorithm for distribution network with distributed generation based on intelligent electronic device information. *Lecture Notes in Electrical Engineering*, 237 LNEE, pp. 275-282. https://doi.org/10.1007/978-3-319-01273-5_30
15. Song X. et al. (2018) Active distribution network protection mode based on coordination of distributed and centralized protection. In: *Proceedings of 2017 China International Electrical and Energy Conference*. Piscataway: Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/CIEEC.2017.8388442
16. Tian J. et al. (2010) A fast-current protection scheme for distribution network with distributed generation. In: *IET Conference Publications*. Piscataway: Institute of Electrical and Electronics Engineers. https://doi.org/10.1049/cp.2010.0319
17. Tong B.B. et al. (2013) A distributed protection and control scheme for distribution network with DG. In: *Advanced Materials Research*. https://doi.org/10.4028/www.scientific.net/AMR.732-733.628
18. Zhong S. et al. (2009) Privacy protection model for distributed service system in converged network. In: *International Conference on E-Business and Information System Security*. Piscataway: Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/EBISS.2009.5138026
19. Zhou C. et al. (2019) Principle of Pilot Protection based on Positive Sequence Fault Component in Distribution Networks with Inverter-interfaced Distributed Generators. In: *IEEE PES GTD Grand International Conference and Exposition Asia*. Piscataway: Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/GTDAsia.2019.8716011
20. Zhu X. et al. (2013) Study on protection scheme for low-voltage distribution network with distributed generation. *Information Technology Journal*, 12 (16), pp. 3655-3659. https://doi.org/10.3923/itj.2013.3655.3659