УДК 004.4

Исследование организации кроссплатформенной рабочей среды с облачным хранилищем данных

А. А. Когутенко, С. В. Плотникова
Государственное бюджетное нетиповое общеобразовательное учреждение
«Республиканский лицей-интернат «Эрудит» – центр для одаренных детей»
Министерства образования и науки Донецкой Народной Республики
and.kogutenko@yandex.ru

Когутенко А. А., Плотникова С. В. Разработка кроссплатформенной рабочей среды с облачным хранилищем данных. *В данной статье рассмотрены этапы создания кроссплатформенной рабочей среды с облачным хранилищем данных. Изучены особенности организации облачных хранилищ данных. Обоснован выбор операционной системы для сервера, проанализированы доступные способы обработки информации в веб-браузере (JavaScript и Flash). Рассмотрено применение и предназначение, достоинства и недостатки основных веб-серверов (Apache HTTP Server, Internet Information Services (IIS) и nginx), серверных языков (Hypertext Preprocessor (PHP), Python, Java, Ruby on Rails), серверных систем управления базами данных (MySQL, SQLite, PostgreSQL) с целью выбора оптимальных инструментов для разработки и обеспечения работоспособности среды. Описана организация системы, определены ключевые моменты политики безопасности (безопасность рабочей среды, безопасность хранилища данных), основной функционал системы.*

Ключевые слова: *WebOS, облачное хранение данных, веб-браузер, расширяемое веб-приложение, JavaScript, Debian, Apache, nginx, MySQL. PHP.*

Введение

В современном мире существует множество различных технологий, одной из которых является облачное хранение данных, дающее доступ к ним из любой точки Интернета и уверенность в сохранности файлов. Есть сервисы, позволяющие хранить информацию удобно и надёжно (Яндекс.Диск [1], Dropbox [2] и т. п.). Однако подобные сервисы не позволяют работать с данными непосредственно из веб-браузера, или же предоставляют крайне ограниченный функционал. Появляется необходимость в разработке расширяемого веб-приложения, которое позволило бы работать с данными и облаком, а именно виртуальный рабочий стол в браузере.

Актуальность разрабатываемой системы заключается в доступе к рабочей среде и облаку непосредственно в веб-браузере – на любом компьютере, ноутбуке, планшете с установленным браузером и доступом к сети Интернет или локальной сети.

Цель статьи: исследовать организацию расширяемой кроссплатформенной рабочей среды с облачным хранилищем данных.

В соответствии с целью в статье ставятся и решаются следующие задачи:

– изучить способы обработки данных посредством веб-браузера.
– исследовать методы организации серверной части приложения.
– спроектировать архитектуру системы: технологии поддержки приложений, процесс обмена данными с облаком.

При использовании виртуального рабочего стола в браузере пользователь имеет следующие преимущества:

– легко переходить с устройства на устройство, работая при этом с актуальными данными из облака;
– в случае поломки устройства данные останутся в облаке, таким образом устраняются проблемы потери данных из-за возможной неисправности накопителей;
– защищённость данных обеспечивается шифрованием на стороне клиента.

Особенности организации облачных хранилищ данных

Облачное хранилище – онлайн-хранилище, в котором данные хранятся на множестве распределённых в сети серверов. То есть, пользователям выделяется пространство на серверах и они получают к нему доступ.

Обычно облачные хранилища работают следующим образом. Как только пользователь начинает загрузку файла в облако, промежуточный сервер делает запрос серверам хранилища на выделение свободного места. С

получением ответа, в котором указывается внутренний адрес сервера с выделенным пространством (что должно происходить почти мгновенно), соединяется с сервером хранилища и передаёт данные непосредственно ему. На серверах хранилища при получении файла сразу же происходит резервное копирование.

Также немаловажную роль играет операционная система, используемая на серверах. От неё требуется надёжность, стабильность работы, производительность и безопасность. Кроме того, изучению и сравнению подлежат принципы работы файловых систем.

Способы обработки данных посредством веб-браузера

Проанализируем доступные способы обработки информации в веб-браузере, их достоинства и недостатки.

Существует две основные технологии, поддерживаемые современными браузерами – JavaScript и Flash [3]. JavaScript поддерживается наверняка, так как встроен непосредственно в веб-обозреватель и поставляется со всеми современными браузерами. Flash является также довольно популярной технологией, но для его установки необходимо скачивать отдельную программу. В настоящее время JavaScript является лучшим выбором для создания интерактивных веб-страниц. Однако во время реализации поддержки работы с микрофоном и камерой не стоит забывать о Flash, так как мультимедийные функции не в полной мере поддерживаются JavaScript. Идеальным вариантом будет комбинирование JavaScript и Flash.

Организация серверной части приложения

Среда обработки *информации.*

Важной составляющей организации работы сервера является выбранная операционная система её надёжность, безопасность и гибкость.

Windows Server хоть и стабильна, но содержит большое количество уязвимостей и является коммерческим продуктом.

Из дистрибутивов Linux одним из самых стабильных является Debian [4]. Операционные системы на базе Linux и BSD очень гибки и имеют бесплатную лицензию. Однако размер ядра Linux постоянно увеличивается; увеличивается и количество уязвимостей в нём.

Известно, что операционные системы семейства BSD отличаются высокой надёжностью и стабильностью. Помимо того, они безопасны и производительны. Вероятность нахождения ошибки в BSD ниже, чем в Linux и Windows Server [5].

Так или иначе, выбор падает на Debian или FreeBSD.

Веб-сервер.

Лидирующие места занимают Apache HTTP Server, Internet Information Services (IIS) и nginx. Возможно разработать новый веб-сервер конкретно для поставленной задачи, однако этот вариант не является надёжным и безопасным. Поэтому лучше использовать решения, зарекомендовавшие себя и проверенные временем.

Рассмотрим применение и предназначение каждого из выбранных серверов.

Веб-сервер Apache [6] создавался для обработки запросов и отдачи контента. К его достоинствам можно отнести гибкость, надёжность, кроссплатформенность, поддержку модулей. Недостатком является его средняя производительность.

Веб-сервер nginx [6] предназначен для отдачи статических данных и передачи динамических запросов другому программному обеспечению для обработки, может использоваться, как прокси. Достоинствами nginx является эффективное потребление ресурсов, кроссплатформенность, возможно использовать, как прокси. Также данный сервер превосходен для отдачи статического контента. Среди недостатков можно отметить среднюю гибкость.

Веб-сервер IIS [7] используется для размещения сайтов в сети Интернет. Его достоинствами являются безопасность, удобство администрирования, а недостатками – средняя производительность, совместимость только с Windows.

Все три веб-сервера поддерживают безопасность на должном уровне. Apache хорошо подходит для обработки информации, nginx – для отдачи статических данных. Также в nginx стоит отметить большую отказоустойчивость, чем в остальных. Так как требуется одновременно быстрое и мощное решение, то лучше всего использовать Apache и nginx в связке: nginx как front-end (для получения запросов и передачи на обработку другому ПО), а Apache – как back-end, для получения и обработки запросов от nginx.

Способ обработки *информации.*

Большое значение имеет выбор языка программирования, позволяющего писать программы/скрипты для обработки данных и генерации динамического контента.

Основными серверными языками являются Hypertext Preprocessor (PHP), Python, Java, Ruby on Rails.

Предпочтение отдаётся производительным и потребляющим мало памяти языкам. Для сравнения эффективности языков были взяты результаты тестирования.

Таблица 1. Сравнение языков программирования по времени работы программы

Операция\Язык	Время выполнения (секунды)			
	PHP	Python	Java	Ruby
Множество Мандельброта	125,17	273,43	7,1	~420
Норма матрицы	37,94	188,83	4,29	141,49
k-нуклеотид	43,96	84,73	7,93	101,95
Задача n тел	~300	~780	21,54	290,75
Сходство белков	59,37	110,91	2,13	77,16
Число π	2,15	–	3,06	3,14
Комплементарность	2,81	2,82	1,1	4,03
Двоичные деревья	88,07	86,9	11,26	54,24
Регулярные выражения	3,34	14,86	12,31	28,8

Таблица 2. Сравнение языков программирования по использованию памяти

Операция\ Язык	Использованная память (КБ)			
	PHP	Python	Java	Ruby
Множество Мандельброта	8,688	13,748	27,108	69,656
Норма матрицы	8,796	9,016	29,884	10,036
k-нуклеотид	235,632	221,028	185,16	133,912
Задача n тел	8,668	7,728	27,092	8,916
Сходство белков	8,812	8,024	28,824	9,712
Число π	9,856	–	31,76	163,316
Комплементарность	359,768	265,428	294,36	133,304
Двоичные деревья	734,364	274,404	520,676	434,456
Регулярные выражения	158,792	439,208	834,824	260,46

Относительно неплохие результаты показывает Java, однако при выполнения некоторых заданий иногда имеет неоправданный расход памяти. Python и Ruby временами выполняются дольше, чем PHP. Для разработки серверной части приложения целесообразно выбрать PHP.

Необходимо определиться с системами управления базами данных, поскольку база данных – хранилище на сервере для различных типов данных, которое достаточно сильно влияет на производительность и стабильность системы.

Так как от базы данных, в первую очередь, требуется надёжность и высокая скорость работы, есть смысл говорить о разработке таковой на производительном языке программирования. Это позволило бы разработчикам полностью осознать принцип работы СУБД и, как следствие, сделать её гарантированно качественной и надёжной.

Рассмотрим три основные серверные СУБД: MySQL, SQLite, PostgreSQL [8].

Таблица 3. Сравнение серверных СУБД

Достоинства	Недостатки
MySQL	
Простота в работе. Богатый функционал. Безопасность. Масштабируемость. Скорость.	Есть ограничения. Проблемы с надёжностью. Медленно развивается.
SQLite	
Состоит из одного файла. Использует язык SQL. Подходит для разработки и тестирования.	Отсутствие системы пользователей. Отсутствие возможностей увеличения производительности. Разрешён только один процесс записи в промежуток времени.
PostgreSQL	
Большое количество дополнений. Объектность.	Низкая производительность.

MySQL, имея довольно высокую скорость и безопасность в работе, также обладает большим количеством функций обработки данных.

Исходя из анализа способов организации серверной части, в качестве операционной системы будет выступать FreeBSD. Уверенно можно сказать, что в идеале необходимо использовать Apache и nginx в связке: nginx для отдачи статического контента и передачи внешних запросов внутреннему ПО (Apache с дополнениями). Надёжное хранилище структурированной информации обеспечит MySQL.

Проектирование системы. Построение базовых концепций

Весь функционал реализован в виде веб-сайта. Сначала посетитель попадает на страницу приветствия, описывающую сервис. Там же расположены ссылки на страницы входа и регистрации.

После входа или регистрации пользователь видит персональный рабочий стол (также в виде

веб-страницы). Некоторые базовые программы, такие как «Настройки» и «Магазин приложений» уже установлены. На использование приложений накладываются некоторые ограничения.

Безопасность *рабочей среды.*

В целях создания безопасной рабочей среды, любое изменение в облаке должно производиться только с помощью функции API. Целесообразно создать отдельный интерпретатор программного кода, так как использование JavaScript для написания пользовательских программ и его встраивание непосредственно в код страницы не обеспечивает безопасность рабочей среды.

Безопасность *хранилища.*

Основные требования к алгоритму шифрования на стороне клиента – производительность и надёжность. Как показывает практика и тесты, наиболее подходящим является алгоритм AES-256. Он существует уже достаточно долго (создан в 1998 году [9]) и успел зарекомендовать себя в качестве быстрого и надёжного алгоритма.

Помимо обычного шифрования может возникнуть потребность в необратимом шифровании (хешировании). Для таких целей целесообразно использовать SHA-512, однако он имеет слишком большую длину и его лучше использовать исключительно для обеспечения безопасности. В случае проверки целостности файлов с помощью контрольных сумм можно применить CRC-32, так как этот алгоритм выдаёт достаточно короткий результат. Проверка целостности информации может пригодиться при запросе пользователем файлов из облака.

Ограничения *на использование приложений.*

По уровню привилегий приложения можно разделить на два класса – системные и пользовательские. Системные приложения не подлежат удалению и имеют повышенные привилегии. Скорее всего, они будут реализованы на «чистом» JavaScript.

Функционал *системы.*

В первую очередь, система должна предоставлять платформу для запуска программ. В её функции входит:

- интерпретация пользовательских программ
- предоставление функций API для приложений
- шифрование передаваемых данных

Помимо того, система должна взаимодействовать с пользователем (с помощью графической оболочки) и иметь в своём составе компоненты для настройки оболочки. Они представляют собой системные приложения.

Выводы

Основной целью является исследование процесса разработки расширяемого приложения для работы с облачным хранилищем. Под расширяемостью предполагается предоставление возможности написания пользовательской программы с использованием API (программный интерфейс приложения), предоставляемого сервисом. Рабочей средой приложения будет являться современный веб-браузер (Chrome, Opera, Firefox).

В процессе рассмотрения структуры облачных систем и технологий обработки информации были рассмотрены такие основные моменты, как:

- устройство облачных систем.
- технологии обработки информации средствами веб-браузера.
- организация серверной части приложения (хранилища): выбор программного обеспечения, наилучшей их компоновки.

Выбрана наиболее подходящая конфигурация для серверных задач. Так, в качестве операционной системы выбрана FreeBSD, так как имеет высокую стабильность и надёжность. Веб-сервером будет комбинация apache и nginx. Базой данных будет выступать MySQL, благодаря хорошей скорости работы и функциональности. В качестве языка программирования для разработки серверной части выбран PHP.

Литература

1. Яндекс.Диск [Электронный ресурс] – 2017 – Режим доступа: https://disk.yandex.ua/ – Загл. с экрана.

2. Dropbox [Электронный ресурс] – Режим доступа: https://www.dropbox.com/ru/

3. Кантор И. Современный учебник Javascript. [Электронный ресурс] – 2017 – Режим доступа: https://learn.javascript.ru/intro – Загл. с экрана.

4. Причины выбрать Debian [Электронный ресурс] – 2017 – Режим доступа: https://www.debian.org/intro/why_debian.ru.html – Загл. с экрана.

5. Почти объективно на тему «чем FreeBSD лучше Linux» [Электронный ресурс] – 2017 – Режим доступа: http://eax.me/freebsd-vs-linux/ – Загл. с экрана.

6. Apache vs Nginx: практический взгляд [Электронный ресурс] – 2017 – Режим доступа: https://habrahabr.ru/post/267721/ – Загл. с экрана.

7. Apache или IIS – сравнение и преимущества [Электронный ресурс] – 2017 – Режим доступа: http://wiki.merionet.ru/servernye-resheniya/3/apache-ili-iis/ – Загл. с экрана.

8. SQLite vs MySQL vs PostgreSQL: сравнение систем управления базами данных [Электронный ресурс] – 2017 – Режим доступа: http://devacademy.ru/posts/sqlite-vs-mysql-vs-postgresql/ – Загл. с экрана.

9. Advanced Encryption Standard [Электронный ресурс] – 2017 – Режим доступа: https://ru.wikipedia.org/wiki/Advanced_Encryption_Standard – Загл. с экрана.

Когутенко А. А., Плотникова С. В. Разработка кроссплатформенной рабочей среды с облачным хранилищем данных. *В данной статье рассмотрены этапы создания кроссплатформенной рабочей среды с облачным хранилищем данных. Изучены особенности организации облачных хранилищ данных. Обоснован выбор операционной системы для сервера, проанализированы доступные способы обработки информации в веб-браузере (JavaScript и Flash). Рассмотрено применение и предназначение, достоинства и недостатки основных веб-серверов (Apache HTTP Server, Internet Information Services (IIS) и nginx), серверных языков (Hypertext Preprocessor (PHP), Python, Java, Ruby on Rails), серверных систем управления базами данных (MySQL, SQLite, PostgreSQL) с целью выбора оптимальных инструментов для разработки и обеспечения работоспособности среды. Описана организация системы, определены ключевые моменты политики безопасности (безопасность рабочей среды, безопасность хранилища данных), основной функционал системы.*

Ключевые слова: *WebOS, облачное хранение данных, веб-браузер, расширяемое веб-приложение, JavaScript, Debian, Apache, nginx, MySQL. PHP.*

Kogutenko A. A., Plotnikova S. V. Development of cross-platform working environment with the cloudy depository of information. *The stages of creation of cross-platform working environment with the cloudy depository of information are considered in this article. The features of organization of cloudy depositories of information are studied. The choice of the operating system for a server is grounded, the accessible methods of treatment of information in a web-browser are analysed (JavaScript and Flash). Application and destiny, dignities and lacks of basic web-servers (Apache HTTP Server, Internet Information Services (IIS) and nginx), server languages (Hypertext Preprocessor (PHP), Python, Java, Ruby on Rails), server control systems by databases with the purpose of choice of optimum instruments for development and providing of capacity of environment is considered. Organization of the system is described, the key moments of policy of safety (safety of working environment, safety of depository of information), basic functional of the system, are certain.*

Keywords: *WebOS, cloudy data storage, web-browser, extended web-appendix, JavaScript, Debian, Apache, nginx, MySQL. PHP.*

Статья поступила в редакцию 25 апреля 2018 г.
Рекомендована к публикации доцентом Зори С. А.