

Разработка сервиса виртуализации мобильных приложений в браузере

И.Н.Старицкий

Волгоградский государственный университет

Аннотация: В работе рассмотрен процесс создания системы облачного доступа к приложению. Эта система позволяет получить доступ к любому программному обеспечению прямо в браузере. В работе первостепенная роль отведена эмуляции именно мобильных приложений, так как они, в отличие от десктопных аналогов, в настоящий момент развиваются наиболее стремительно.

В первой части работы описан процесс проектирования и непосредственной разработки серверной части системы облачной виртуализации приложений в браузере. Указаны проблемы и трудности, возникшие в процессе работы.

Вторая часть работы посвящена дальнейшим планам и разработкам. А именно, рассмотрены пути виртуализации мобильных приложений, и даны объяснения, почему для аппаратной виртуализации выбрана платформа Cubieboard.

Ключевые слова: облачные вычисления, виртуализация, аппаратная виртуализация, сервис приложений.

Введение

В последние годы главным направлением развития IT отрасли являются облачные вычисления. Постоянно этот термин упоминается в средствах массовой информации, большие компании заявляют о внедрении этих технологий в своей инфраструктуре. И уже ни одна техническая конференция не обходится без доклада про Cloud computing. В настоящее время лидерами этой отрасли являются такие гиганты, как Google с сервисом App Engine, Microsoft с Azure и Amazon с EC2, который входит в инфраструктуру AWS [1, 3].

Не смотря на это, небольшие компании тоже смогли выйти на рынок облачных технологий и успешно там обосноваться. Самые успешные из них – это Dropbox и Digital Ocean [2].

Перечисленные выше компании предоставляют хранилище данных на удаленных серверах, инфраструктуру (IaaS), платформу (PaaS) или уже конечное приложение (SaaS). Все эти новые технологии переносят нас в

будущее, где все данные хранятся и обрабатываются на удаленных серверах, а компьютер является только устройством вывода и ввода информации [4, 5].

Пока до этого ещё далеко, и в нашем настоящем нам постоянно приходится сталкиваться с обычным, не облачным программным обеспечением. Перенести его на удаленные машины и предоставить его для работы конечному клиенту по модели SaaS - совершенно нетривиальная задача. Для того чтобы импортировать в Web такую программу, как Autodesk 3D Max, потребуется большой штат дополнительных разработчиков, так как функционал программы писался десятилетиями. Для мобильных ПО все еще сложнее, тут есть и проблемы с покрытием интернета и с задержкой данных. Но в целом процесс перехода в облака неотвратим и появляется все больше и больше решений. Чтобы уже сейчас использовать мобильное или десктопное ПО удаленно на любом устройстве, и разрабатывается система облачного доступа к приложению.

В работе рассматривается проблема виртуализации мобильных приложений, так как этот рынок развивается наиболее стремительно.

Разработка клиентской части

Клиентом в данном случае являлась программа, получающая изображение экрана удаленного компьютера и взаимодействующая с ним по протоколу RFB. Управление осуществляется путем передачи информации о нажатой на клавиатуре клавише, или о перемещении мыши.

Для реализации управления удаленного устройства в окне браузера, необходимо обеспечить передачу изображения с сервера компьютера по средствам протокола WebSocket.

WebSocket — протокол связи поверх TCP-соединения, предназначен для обмена данными между веб-клиентом и сервером в режиме реального времени. В настоящее время протокол стандартизирован W3C [6,7].

Протокол WebSocket имеет две URI схемы, ws: (нешифрованное соединение) и wss: (шифрованное соединение).

Для реализации клиента системы облачного доступа к приложению была использована именно эта технология. Несомненным её плюсом является, по сути, революционный подход использования приложения, основанного на парадигме HTTP. Больше не требуется ждать ответа от сервера и можно отправлять, а также принимать запросы полностью асинхронно и симметрично.

Для примера работы VNC клиента использующий технологию WebSocket (рис.1), было взято потоковое изображение рабочего стола Windows, обрезано на несколько частей и отправлено в сессию WebSocket.

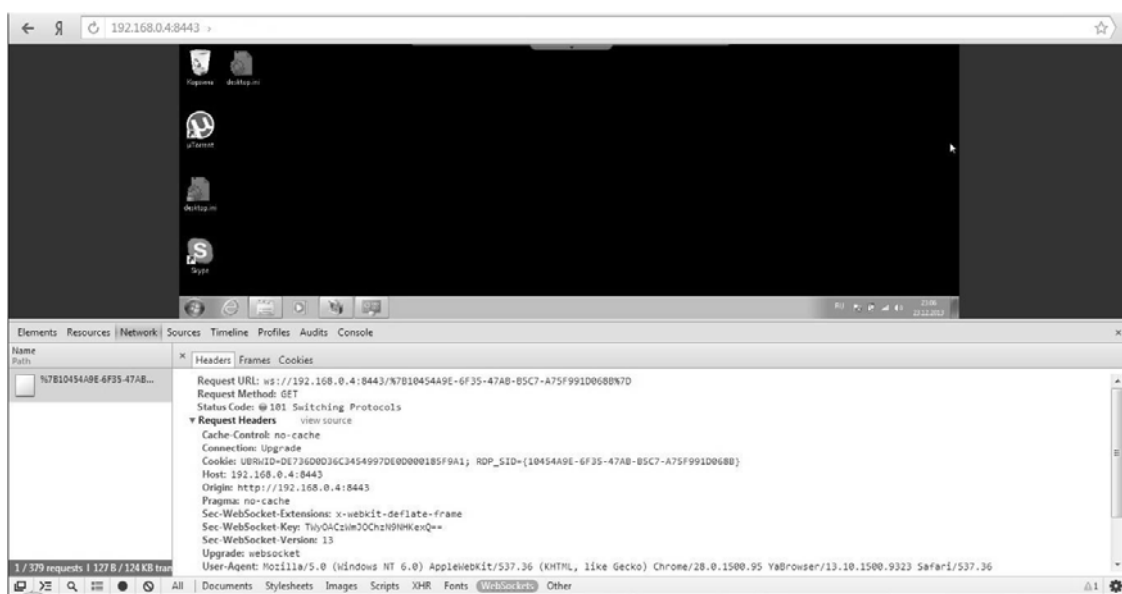


Рис. 1. – Пример работы VNC клиента по технологии WebSocket

Для передачи изображения в VNC клиенту используется Binary Frames, в котором хранятся порезанные части изображения. На рис.2 можно увидеть передачу Binary Frames в браузер. Помимо приема изображения от сервера, для полноценного удаленного доступа требуется ещё и передача данных о передвижении мыши и о введенных символах.

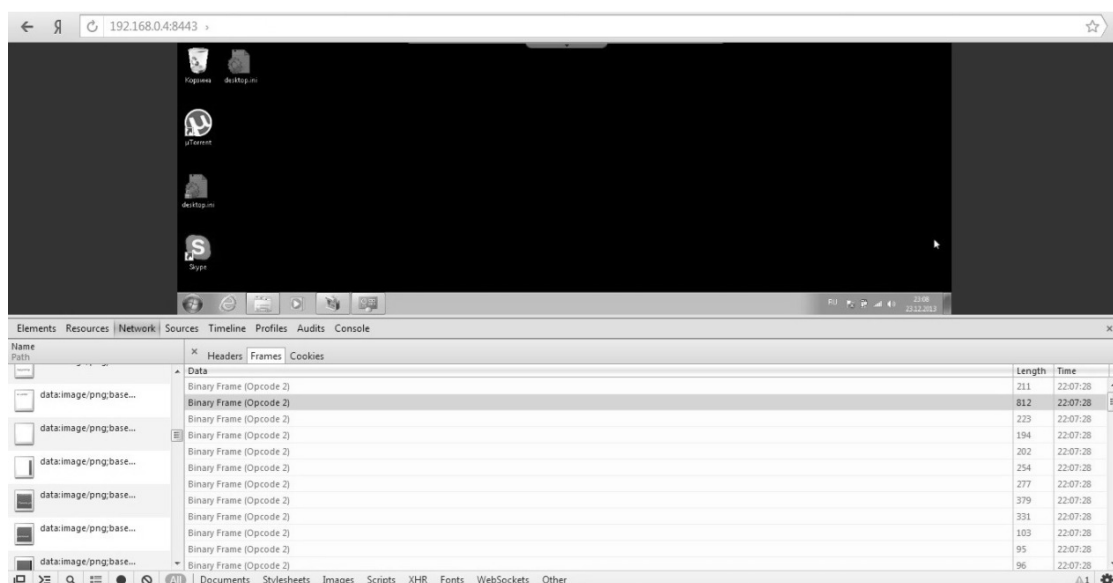


Рис. 2. – Прием клиентов изображения от сервера

WebSocket позволяет решать все задачи, возникающие в результате работы с удаленным рабочим столом внутри браузера. При этом имеет более высокую скорость работы по сравнению с обычной AJAX HTTP передачей данных. Еще одним плюсом WebSocket является то, что он сохраняет преемственность с основными web-технологиями, тем самым упрощается процесс понимания технологии его работы.

Разработка серверной части

В процессе работы над проектом было испробовано множество способов эмулирования Android операционной системы программно.

И в итоге основным решением оказалось использование виртуальной машины с гипервизором. При проведении исследовании нагрузок на вычислительную машину получалось, что на один слепок операционной системы приходится 30% ЦП шестиядерного AMD FX-6300 (рис.3).

Процессы	ИД пр...	Потоки	ЦП	Среднее для ЦП
<input type="checkbox"/> Образ				
<input type="checkbox"/> player.exe	3076	36	21	18.09
<input type="checkbox"/> VBoxHeadless.exe	4048	36	16	16.39
<input type="checkbox"/> explorer.exe	3936	46	7	0.14
<input type="checkbox"/> SearchIndexer.exe	5912	14	3	0.05
<input type="checkbox"/> chrome.exe	8532	14	2	0.19
<input type="checkbox"/> System	4	139	2	1.46
<input type="checkbox"/> Системные прерывания	-	-	1	0.88
<input type="checkbox"/> audiodg.exe	316	17	1	0.61
<input type="checkbox"/> chrome.exe	2848	15	1	0.51

Рис. 3. – Загрузка ЦП для одной виртуальной машины

Из рис. 3 видно, что потребляемые гипервизором ресурсы достаточно высоки, из-за этого облачный проект по виртуализации Android в данной модели не сможет иметь широкого распространения и развития, так как не выгодно содержать такие вычислительные мощности. Соответственно нужен совершенно иной подход.

Полноценным решением всех описанных выше проблем является Cubieboard. Это микрокомпьютер с процессором ARM Cortex A8, специально разработанный под операционные системы Linux и Android. Стоимость его \$50, что 2 раза меньше, чем стоимость потребляемых ресурсов виртуальной машины. С помощью этого устройства можно сделать сервис аппаратной виртуализации Android, где на каждом устройстве будет предустановлена ОС, и доступ к ней будет даваться пользователю по требованию[8-10].

В итоге общая схема серверной части системы облачного доступа к приложению, представляет из себя веб-сервер (рис.4) к которому обращается веб-браузер пользователя, Linux сервер, который обеспечивает всю логику доступа к устройствам, и сами устройства Cubieboard, которые передают информацию о своём статусе, доступности или недоступности.

Для работы с программой каждому пользователю будет выделено 5 минут, в итоге 100 Cubieboard могут за сутки обслуживать 28800 уникальных

пользователей. Кластер с восьмью Cubieboard показан на рис. 4.

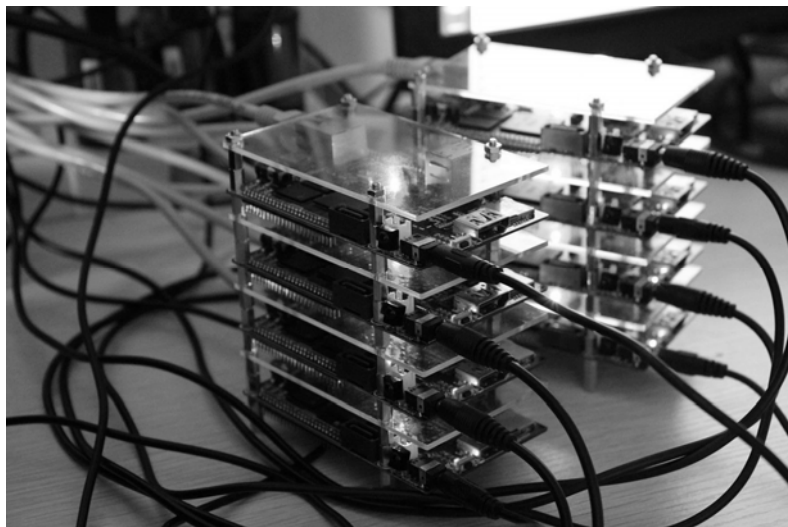


Рис. 4 .– Кластер Cubieboard

Использование аппаратной виртуализации поможет отказаться от гипервизора и создать инновационный сервис облачной виртуализации мобильных приложений.

В процессе работы и была реализована аппаратная виртуализация с использованием кластера из машин Cubieboard.

Был разработан прототип административной панели управления работой с облачным сервисом. Программный интерфейс показан на рис.5.

В административной панели можно найти информацию о последних визитах в систему, о работе аппаратной виртуальной машины, есть возможность подключиться к удаленному управлению. Также реализован функционал генерации HTML кода для вставки на сайт, с доступом к конкретной виртуальной машине.

Полученный прототип передавал картинку и осуществлял удаленное управление Android операционной системы. Но основная проблема заключалась в том, что изображение подгружалось очень медленно, из-за того что устройство передавало картинку покадрово.

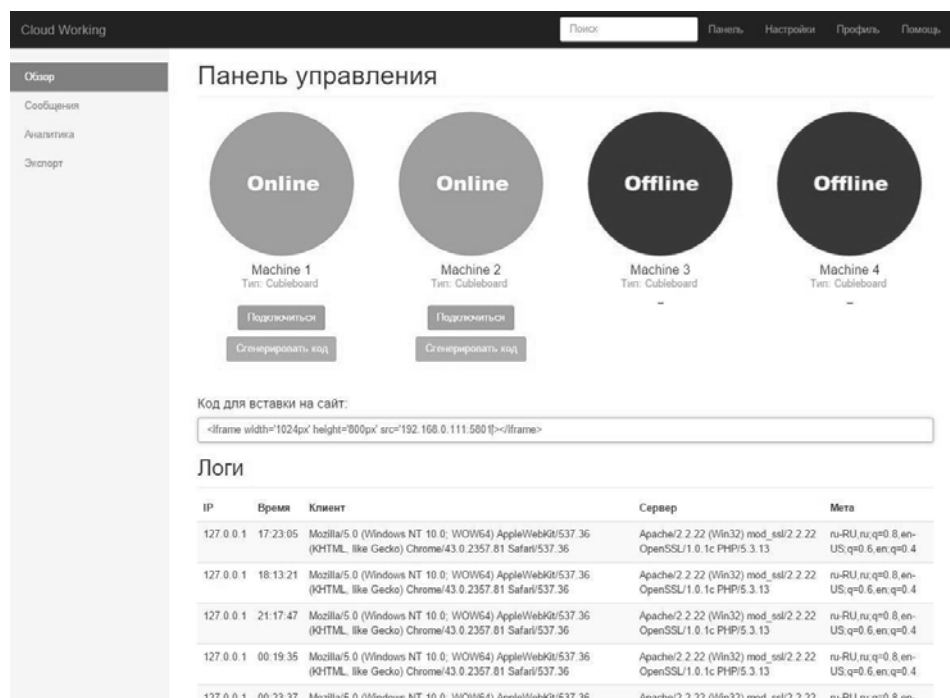


Рис. 5. – Программный интерфейс Cubieboard

Такая низкая скорость работы не соответствовала требованиям, изначально предъявляемыми к будущему продукту. Ведь сама суть состояла в том, чтобы предоставить технологию, при которой работа с приложением в веб-браузере ничем не отличалась от работы с реальным приложением на мобильном устройстве.

Соответственно потребовалась альтернативная технология, которая поможет осуществлять кодирование и декодирование фреймов, чтобы увеличить скорость передачи видеоизображения. В настоящий момент на основе технологии libjreg удалось получить картинку с устройства достаточной плавности, работы в этом направлении продолжаются.

Литература

1. Brodtkin J. Open source fuels growth of cloud computing, software-as-a service. Network World, 2012. 432 p.

2. Joseph M. Proceedings of the 1st ACM symposium on Cloud computing. N. Y.: ACM, 2010. 402 p.
3. Lee G. Cloud Computing: Principles, Systems and Applications. L.: Springer, 2010. 379 p.
4. Prajapati K., Sarkar, A. Trust Management Model for Cloud Computing Environment// India. International Conference on Computing Communication and Advanced Network 2013. pp. 24–30.
5. Ullah S. TCloud: A Dynamic Framework and Policies for Access Control across Multiple Domains in Cloud Computing // International Journal of Computer Applications. 2013. V. 62(2). pp. 49–51.
6. Берёза Н.В. Современные тенденции развития мирового и российского рынка информационных услуг // Инженерный вестник Дона, 2012, №2 URL: ivdon.ru/magazine/archive/n2y2012/758/.
7. Пономарева Е.И. Совершенствование процесса обработки данных при помощи облачных вычислений // Инженерный вестник Дона, 2012, №1 URL: ivdon.ru/magazine/archive/n1y2012/628/.
8. Риз Д. Облачные вычисления. СПб.: БХВ-Петербург, 2011. 288 с.
9. Фингар П. Dot.cloud: облачные вычисления – бизнес-платформа XXI века. М.: Акваринная книга, 2011. 256 с.
10. Широкова Е.А. Облачные технологии // Современные тенденции технических наук: материалы международной заочной научной конференции (г. Уфа), 2011. С. 30-33.

References

1. Brodtkin J. Open source fuels growth of cloud computing, software-as-a service. Network World, 2012. 432 p.
 2. Joseph M. Proceedings of the 1st ACM symposium on Cloud computing. N. Y.: ACM, 2010. 402 p.
-



3. Lee G. Cloud Computing: Principles, Systems and Applications. Springer, 2010. 379 p.
4. Prajapati K., Sarkar, A. International Conference on Computing Communication and Advanced Network, 2013. pp. 24–30.
5. Ullah S. International Journal of Computer Applications, 2013, V. 62(2). pp. 49–51.
6. Bereza N.V. Inzhenernyj vestnik Dona (Rus), 2012, №2. URL: ivdon.ru/magazine/archive/n2y2012/758/.
7. Ponomareva E.I. Inzhenernyj vestnik Dona (Rus), 2012, №1. URL: ivdon.ru/magazine/archive/n1y2012/628/.
8. Riz D. Oblachnye vychislenija. SPb [Cloud computing]. BHV-Peterburg, 2011. 288 p.
9. Fingar P. Dot.cloud: oblachnye vychislenija – biznes-platforma XXI veka [Dot.cloud: cloud computing – business platform XXI century]. M.: Akvamarinovaja kniga, 2011. 256 p.
10. Shirokova E.A. Sovremennye tendencii tehniceskikh nauk: materialy mezhdunarodnoj zaochnoj nauchnoj konfeencii (g. Ufa), 2011. pp. 30-33.