

Оглавление

Технологии и алгоритмы

Progressive Downloads and Rendering / Stoyan Stefanov (Yahoo).....	3
Профилирование памяти в приложениях на Python / Антон Грицай (Андромеда)	5
Компиляция скриптов PHP / Алексей Романенко (РБК).....	6
Сервер-агрегатор на python (для Xscript/FEST) / Андрей Сумин, Михаил Сабуренков и Павел Труханов (HeadHunter).....	7
Приемы разработки высоконагруженных приложений на Twisted / Андрей Смирнов (Qik)	8
Microsoft Ajax Minifier — автоматическая оптимизация JavaScript и CSS для веб-сайтов высокой производительности / Евгений Чигиринский (Microsoft)	9
Incubation project: Social Monitoring Tool codename Looking Glass / Patrice Pelland (Microsoft)	10
Быстрое развертывание шаблонов и статики в Mail.Ru / Николай Кондратов (Mail.ru)	11
Tarantool/Silverbox: высокопроизводительная база данных в оперативной памяти / Юрий Востриков (Mail.ru)	12

Архитектуры

Scaling to Hundreds of Millions of Requests: What Worked and What Didn't / James Golick.....	13
Секреты Фейсбука: как выдержать 50 миллионов запросов в секунду / Robert Johnson (Facebook).....	14
О чем стоит подумать, приступая к разработке высоконагруженной системы или top-10 ошибок, которые совершаются еще до начала разработки / Артем Вольфтруб (Грамант)	15
Как мы строим CDN в России / Ярослав Городецкий	17
Оптимизация одного из топовых приложений для социальной сети ВКонтакте: 1000 запросов в секунду на Rails / Максим Лапшин.....	18
Extreme Cloud Storage on FreeBSD / Андрей Пантюхин.....	19
Использование OMQ для построения low latency-распределенных систем / Андрей Охлопков, Алексей Ермаков (Global Hedge Capital Group)	20
Интеграция открытых технологий и взаимодействие со сторонними проектами в условиях высоких нагрузок / Олег Илларионов (ВКонтакте)	21
Shared Personalization Service — How to Scale to 15K RPS / Patrice Pelland (Microsoft)	22
Практическое создание крупного масштабируемого web 2.0 проекта с нуля / Дмитрий Бородин (Сонетика.ру, Лице-Мер)	23

Системное администрирование

Мониторинг: XXI век / Смирнова Алиса, Дмитрий Никоненко (Яндекс).....	25
Некоторые аспекты влияния сходимости протокола BGP на доступность сетевых ресурсов / Александр Азимов (Highload Lab)	26
Динамика DDoS-атак в России / Александр Лямин (Highload Lab)	28
Тандемные DDoS-атаки / Проблематика уязвимостей в спецификации TCP/IP (фундаментальные уязвимости) / Артем Гавриченко (Highload Lab)	29
Performance tweaks and tools for Linux / Joe Damato.....	31
Cloud APIs: обзор API западных провайдеров и API Scalaxy / Нат Гаджибалаев (Scalaxy)	32

Смежные технологии

Native Client / Евгений Эльцин (Google)	33
Yota Access: Система для раздачи абонентам информации о загрузке базовых станций / Кирилл Коринский (Yota).....	34
2.5D игры и особенности разработки многопользовательских игр / Глеб Полушкин (SM&Partners).....	35

Тестирование

Нагрузочное тестирование без границ / Юрий Ковалёв (Performance Lab)	36
Сервис нагрузочного тестирования ddosme.ru / Иван Самсонов (Scalaxy)	37

Базы данных и системы хранения

Developing PostgreSQL Performance / Simon Riggs (PostgreSQL)	38
Managing Replication of PostgreSQL / Simon Riggs (PostgreSQL).....	39
The Magic of Hot Streaming Replication / Bruce Momjian (PostgreSQL)	40
Rapid Upgrades With Pg_UpgradeBruce Momjian (PostgreSQL) / Bruce Momjian (PostgreSQL).....	41
Сравнительный анализ хранилищ данных / Олег Царев, Кирилл Коринский (Percona, Yota)	42
Мы решили написать собственное хранилище данных / Илья Космодемьянский (Интелотек групп).....	43
Sphinx для высоконагруженных и масштабируемых проектов / Вячеслав Крюков (Ivenco)	44
Сравнение, тестирование и миграция из обычных индексов в real time индексы Sphinx Search / Ярослав Ворожко (Ivenco)	45
Масштабируемая система голосования на базе PostgreSQL PgQ / Сергей Нековаль (Грамант)	46
Организация хранилища с vast sky / Дмитрий Лоханский (Scalaxy)	47
InnoDB: архитектура транзакционного хранилища / Константин Осипов (Oracle).....	48

Технологии и алгоритмы

Progressive Downloads and Rendering

Stoyan Stefanov (Yahoo)

Воспользовавшись преимуществами таких продуктов, как YSlow и PageSpeed, что вы можете сделать со своим приложением дальше? Пора задуматься о том, как улучшить скорость и время доступа. Мы все знаем, как обманывает нас зрение, когда дело касается времени и продолжительности загрузки. Как же убедить пользователя в том, что приложение действительно быстро загружается и реагирует на запросы? Ключевое слово — «прогресс». Когда компоненты страницы загружаются параллельными потоками, и нет никаких проблем в других местах, страница грузится быстрее. А когда страница быстрее обновляется, это не только внушает пользователю уверенность и дает гарантии того, что приложение прекрасно работает и завершает процессы, а также, что приложение работает быстро и мгновенно реагирует на запросы.

На этом докладе вы также узнаете:

- Роль «прогресса» (загрузки, обновления, усиления) в производительности интерфейса;
- Как избегать неожиданных проблем, таких как загрузка иконок и комментариев;
- Предварительная загрузка CSS и JavaScript без исполнения;
- CSS и высокие нагрузки;
- Трюки, которые используют такие гиганты, как Yahoo, Amazon, Google, для обеспечения лучших скоростей.

О докладчике

Стоян Стефанов (Stoyan Stefanov, phpied.com, [@stoyanstefanov](https://twitter.com/stoyanstefanov)) — разработчик интерфейсов компании Yahoo! Search. Создатель smush.it — онлайн-оптимизатора изображений и архитектор YSlow 2.0. Автор книг по объектно-ориентированному JavaScript, *JavaScript Patterns*, *Speed Matters*, учредитель *Even Faster Web Sites* и *High-Performance JavaScript*, спикер *Velocity*, *Ajax Experience*, *JSConf*.

Профилирование памяти в приложениях на Python

Антон Грицай (Андромеда)

1. Суть проблемы. В двух словах о том, что такое утечки памяти, откуда они могут браться. Для чего, кроме борьбы с явными утечками, используется профилирование?
2. Как утечки могут возникать в средах исполнения с автоматическим управлением памятью, в частности в CPython? Темные стороны работы сборщика мусора в CPython.
3. Ну и фиг с ним. Даже если мое приложение течет, у меня есть несколько возможностей сделать так, чтобы это не сильно мешало жить. В каких случаях это применимо, а в каких нет?
4. Профилирование памяти: если п.3 все же не подходит. Какие существуют инструменты в мире Python?
5. Профайлер Neary: общие идеи и принципы работы. В двух словах о его шизофреническом командном интерфейсе.
6. Заключение. Что нужно делать и чего стоит избегать, чтобы утечки не появлялись? Каков простейший сценарий профилирования утечки?
7. За кадром. Управление памятью в c-extensions: на стыке ручного и автоматического. Работа сборщиков мусора в альтернативных средах исполнения — Jython, Unladen, PyPy.

Целевая аудитория доклада

Доклад будет интересен тем, кто разрабатывает на Python и на других динамических языках. Не только для web, но и десктопные приложения, а также сетевые сервисы / системные демоны. Тема практически не освещена в Рунете, несмотря на свою актуальность.

О докладчике

Программирую на Python пятый год. Работаю в холдинге «Андромеда» (<http://andromeda.ru>), занимаюсь разработкой новой версии системы онлайн-мониторинга мобильных и стационарных объектов. Система написана главным образом на Python.

Компиляция скриптов PHP

Алексей Романенко (РБК)

Речь в докладе пойдет о возможных способах компиляции php-скриптов в native code. После выхода компилятора HipHop от Facebook эта тема стала снова актуальной.

Будет сделан обзор существующих решений (Roadsend, phc, HipHop), их отличия, ограничения и способы использования. Также планируется провести тестирование и сравнительный анализ различных способов запуска скриптов: mod_php, fast-cgi, phc, hiphop. После этого можно будет подвести итог, насколько использование компиляции скриптов оправдано по сравнению со стандартной схемой оптимизации php+APC (или другого акселератора) и имеет ли это направление дальнейшее развитие.

Возможно, имеет смысл сравнить компиляторы для других интерпретируемых языков: python, perl.

О докладчике

Руководитель отдела технических разработок в компании РБК.

Сервер-агрегатор на python (для Xscript/FEST)

Андрей Сумин, Михаил Сабуренков и Павел Труханов (HeadHunter)

Проблемы

- Надо разделять монолитное приложение на части или зачем нам сервер-сборщик.
- Concurrency.

Решения

Xscript. Решение от Яндекса. Полностью решает обе поставленные задачи, но в процессе выявился ряд неудобств:

- XML программирование;
- Не заточен для HTTP;
- Разработка «далеко».

Frontik. Реализован на базе Tornado, интересные моменты:

- Процессинг страницы;
- Стадии: Page, XSL, POST;
- SyncGroups;
- Doc;
- Future;
- Минусы: Lock на XSL, память под XSL, макаронный код.

Бонусы:

- Графики производительности;
- Обязка всего проекта;
- Легкая интеграция.

Приемы разработки высоконагруженных приложений на Twisted

Андрей Смирнов (Qik)

В докладе пойдет речь о конкретных проблемах и их решениях, а также о подходах в разработке нагруженных приложений на Python/Twisted. Многие из описанных решений подойдут и при разработке на других похожих фреймворках, таких как Ruby/EventMachine и Perl/POE.

1. Запуск и шедулинг многих однопоточных процессов на одном сервере.
2. Key-value storage и приемы работы с ним.
3. Обслуживание сотен тысяч соединений на одном сервере.
4. HTTP-сервисы и балансировка нагрузки, локализация нагрузки.
5. Сбор статистики, интеграция с системой мониторинга.
6. Шина обмена сообщениями на примере AMQP.
7. Поиск и устранение memory leak.
8. Оптимизация по времени отклика и пропускной способности.
9. Мифы и правда о Python как языке разработки нагруженных приложений.

Microsoft Ajax Minifier — автоматическая оптимизация JavaScript и CSS для веб-сайтов высокой производительности

Евгений Чигиринский (Microsoft)

Доклад об особенностях разработки и развития одного из самых посещаемых сайтов в мире — www.msn.com и методах обеспечения его быстродействия путем использования утилиты Microsoft Ajax Minifier.

О докладчике

Евгений Чигиринский — сотрудник компании Майкрософт с 2003 года, принимал участие в разработке продуктов для девелоперов и моделинга, таких как Microsoft Visio, Visual Studio Class Designer, Visual Studio Team System Data и т. д. В настоящее время работает над платформой для веб-девелоперов для Microsoft Network (MSN).

Incubation project: Social Monitoring Tool codename Looking Glass

Patrice Pelland (Microsoft)

Silverlight 4 application using MEF and MVVM design patterns hooked up to WCF services and SQL. The talk will also cover the architecture of the same application designed to be in the cloud on Windows Azure.

О докладчике

Patrice Pelland is a principal developer manager at Microsoft with more than a dozen years of experience designing software. He is also an experienced mentor and an instructor of programmers at all skill levels.

Быстрое развертывание шаблонов и статики в Mail.Ru

Николай Кондратов (Mail.ru)

Доклад о работе со статикой и шаблонами на основных проектах Mail.Ru (Почта, Мир, Фото, Видео). Организация работы верстальщиков и разработчиков клиентской части – от разработки до тестирования и выкладки.

- Устройство шаблонов в Mail.Ru
- Общая организация системы выкладки
- Тестовое окружение и организация работы верстальщика
- Миграция с CVS на git
- Внедрение новой схемы раскладки и тестирования
- Особенности настройки серверов, участвующих в раскладке
- Использование FuseFS
- Автоматизация сборки шаблонов и статики
- Развитие системы и использование p2p технологий

О докладчике

Кондратов Николай – технический руководитель почтовой службы (Mail.Ru)

Tarantool/Silverbox: высокопроизводительная база данных в оперативной памяти

Юрий Востриков (Mail.ru)

Необходимость in-memory баз данных:

- Высоконагруженные плоские таблицы;
- Скорость разных типов памяти.

Tarantool framework:

- Устройство SLAB аллокатора;
- Массаракш: код наизнанку;
- Persistence;
- Hot Standby.

Клиентские интерфейсы:

- Протокол IProto/Silverbox;
- Эмуляция memcached.

Результаты:

- Производительность;
- Дальнейшие планы.

Архитектуры

Scaling to Hundreds of Millions of Requests: What Worked and What Didn't

James Golick

All over the web, people are discussing the latest scalable database or cloud computing platform. Claims are made about scalability, reliability, and performance. But by definition, most of this hype is simply conjecture. Its authors rarely have the experience to support their claims.

In the process of scaling our ruby, rails, and scala app to 150 million monthly pageviews, we've tested a lot of these technologies and ideas. We've put several popular and supposedly scalable NoSQL database solutions in to heavy production use. We've deployed both cloud computing instances and physical hardware. Some things worked. Others failed spectacularly.

In this talk, I'll share our experiences, good and bad. These are some of the topics we'll cover:

- **Virtualization and Cloud Computing:** Virtualized hardware is the default choice for new deployments. That makes sense - it's easy to get started and you only pay for what you need. But when (if ever) does it make sense to move to physical hardware? It might be sooner than you think.
- **NoSQL:** NoSQL is all the rage these days. There are quite a few solutions, nearly all of them advertised as fast, scalable, reliable. But what are NoSQL databases really like to run in production?
- **Designing for High Availability:** Everybody talks about machines failing, but software tends to fail more often. We should design our systems to be resilient to spikes in service latencies, garbage responses, and altogether failures.

Секреты Фейсбука: как выдержать 50 миллионов запросов в секунду

Robert Johnson (Facebook)

За 6 лет Facebook прошел путь от идеи, рожденной в комнате общежития, до популярнейшей социальной сети в мире, с более чем 500 миллионами активных пользователей. Этот невероятный рост породил множество технических требований. Вот только некоторые из них: 100 млрд просмотров в день, 50 млрд хранящихся фотографий, и кеш, содержащий 2 трлн объектов и обрабатывающий сотни миллионов запросов в секунду.

Я расскажу о технологиях, необходимых для запуска сайта таких масштабов, в том числе о конкретных технологиях, созданных нами, некоторых принципах, применяемых нами в работе с масштабированием, и уроках, полученных нами во время этих разработок. Также я расскажу о наших новейших разработках в области масштабирования — пропуска социального контекста из сети через платформу Facebook.

О докладчике

Бобби Джонсон (Bobby Johnson) — руководитель отдела разработки в Facebook, где он разрабатывает технологии по экономически эффективному масштабированию инфраструктуры и оптимизации производительности для миллионов пользователей Facebook.

За время работы докладчика в компании количество пользователей увеличилось в 50 раз и достигло 100 млрд просмотров страниц в день.

Ранее Бобби работал в компании ActiveVideo Networks, где руководил распределенными системами и командой разработчиков set-top software. Он работал с широким спектром инженерных задач: от робототехники до embedded-систем в веб-приложениях. Он получил степень научного бакалавра инженерных и прикладных наук в Калифорнийском технологическом университете.

О чем стоит подумать, приступая к разработке высоконагруженной системы или top-10 ошибок, которые совершаются еще до начала разработки

Артем Вольфтруб (Грамант)

Согласно данным статистики, на начало 2010 года аудитория интернет-пользователей в России составляет около 43 млн человек, или 37 % населения страны. Гигантский потенциал этого рынка, а также его быстрый рост привели к тому, что многие компании в спешном порядке стремятся занять эту нишу, сделав свой интернет-проект, который перевернет мир.

Долгое время сдерживающим фактором для выхода на рынок интернет-приложений являлась сложность привлечения аудитории. После того как крупнейшие социальные сети открыли API для разработки приложений, порог входа снизился до минимума. Усилий одного-единственного разработчика зачастую достаточно для того, чтобы приложение собрало миллионную аудиторию.

Пример «Веселого фермера» показал, что на этом рынке можно неплохо зарабатывать, поэтому в создание приложений для социальных сетей начали вкладывать деньги. Похожая картина наблюдается и с другими интернет-проектами, которые не связаны с социальными сетями, но также рассчитаны на широкую аудиторию.

Разработка интернет-систем имеет свою специфику. Если руководить проектами следует старым проверенным приемам, результат, скорее всего, будет отрицательный. Кроме того, многие ошибки совершаются по причине того, что руководители проектов, плохо ориентируясь в стеке технологий, пытаются максимально контролировать действия разработчиков.

В этом докладе мне бы хотелось рассмотреть несколько наиболее типичных, с моей точки зрения, ошибок, большинство из которых совершаются еще до начала разработки. Примеры взяты из реальной жизни.

«У нас есть своя IT-команда, но она сильно загружена в ближайшие три месяца. Мы рассчитываем, что за это время вы напишете первую версию системы, которую мы дальше будем развивать своими силами».

- Цикл разработки высоконагруженной системы;
- Как правильно передать разработку другой команде.

«В первую версию системы должно войти N фич. У нас есть еще несколько минорных пожеланий, но их можно будет реализовать после выпуска первой версии».

- Что в действительности должно входить в первую версию системы;
- Бета-версия как основной инструмент для формирования требований.

«Система должна быть масштабируемой. Нам нужен подробный план того, как мы будем справляться с нагрузками, когда система вырастет со 100 000 пользователей до 10 000 000».

- Возможно ли правильно рассчитать нагрузку;
- Где обычно возникают узкие места.

«Производительность системы будет проверяться путем проведением полноценного нагрузочного тестирования перед сдачей проекта».

- Можно ли доверять результатам нагрузочного тестирования;
- В каких случаях оно невозможно.

«Что значит приемлемый уровень отказоустойчивости? Система должна работать безотказно!»

- Когда действительно нужна отказоустойчивость;
- Магия чисел, или сколько девяток нужно вашей системе.

«Зачем нам система мониторинга? Если система сломается, это и так все увидят»

- Для чего действительно нужен мониторинг;
- Как организовать мониторинг и не утонуть в потоке информации.

«Согласно последним обзорам, производительность фреймворка XYZ выше, чем ZYX. Давайте разрабатывать систему с использованием XYZ».

- Когда разработчики имеют право самостоятельно выбирать платформу для разработки;
- Действительно ли производительность XYZ выше, чем ZYX.

«Наши IT-шники не разбираются в вашей системе. Напишите нам максимально подробную пошаговую инструкцию, как ее устанавливать и поддерживать».

- Кто должен заниматься поддержкой;
- Можно ли разделить ответственность за сопровождение системы.

«Мы нашли баг в системе, вы можете прислать нам последнюю версию, мы выложим ее сегодня ночью».

- Что дает правильно организованное проектное окружение;
- Формальные процедуры для поддержки системы: бюрократия или лекарьство от неприятностей?

«Зачем переписывать код, который был написан всего пару месяцев назад. У нас еще куча фич, которые нужно реализовать».

- Кто должен расставлять приоритеты;
- За чей счет должны осуществляться переделки.

Целевая аудитория

Доклад будет интересен руководителям проектов, специалистам, которые формируют требования, менеджерам продуктов, а также представителям разработчиков, которые участвуют в переговорах с заказчиками.

О докладчике

Артем Вольфтруб, руководитель разработки компании «Грамант». Более 10 лет занимаюсь разработкой программного обеспечения. Эксперт в области веб-технологий, электронной рекламы и финансовых систем. Выступал с докладами на конференциях HighLoad 2008, HighLoad 2009, RIW 2009, РИТ++ 2010.

Как мы строим CDN в России

Ярослав Городецкий

1. Что такое CDN и зачем он нужен (краткий ликбез);
2. Классификация CDN по способу расположения серверов:
 - архитектура Akamai-style (расстановка серверов по сетям интернет-провайдеров);
 - архитектура LimeLight-style (подключение к провайдерам в точках обмена трафиком);
 - анализ «за» и «против».
3. Классификация CDN по способу распределения нагрузки:
 - DNS;
 - HTTP Redirect;
 - анализ «за» и «против».
4. Классификация CDN по способу распространения контента:
 - «классическая» архитектура (точка – многоточка);
 - P2P;
 - анализ «за» и «против».
5. Классификация CDN по способу нахождения кратчайшего пути до пользователя:
 - метрики BGP;
 - другие метрики;
 - анализ «за» и «против».
6. Что из всего вышеперечисленного правильно применять в российских условиях, или как строим свою CDN мы.
7. Для чего мы делаем все это:
 - сервисы, которые предоставляются на CDN;
 - API для управления этими сервисами для клиентов.

Оптимизация одного из топовых приложений для социальной сети ВКонтакте: 1000 запросов в секунду на Rails

Максим Лапшин

С первого РИТа бытует миф, что «рельсы медленные». Реальность отличается.

Проблема не в рельсах, а в плохих продуктах.

В докладе я расскажу, что было предпринято для увеличения пропускной способности приложения для ВКонтакте с 9 сек. на запрос и 30 000 запросов в сутки до 25 мс на запрос и 60 млн запросов в сутки.

Рецепт в правильном шардинге, использовании RabbitMQ для развязывания компонент и постоянном профилировании всех компонент.

Extreme Cloud Storage on FreeBSD

Андрей Пантюхин

В России готовится к запуску уникальный массовый медиапроект. Детали пока не афишируются, но нет секрета в том, что к технической части были выдвинуты беспрецедентные требования. Одна из важнейших подсистем — файловое хранилище. О нем и пойдёт речь в докладе.

К системным разработчикам был выдвинут стандартный набор требований:

- высокая общая скорость доступа — десятки гигабит в секунду;
- солидный объём основных данных — сотни терабайт;
- потеря данных недопустима;
- бесперебойная доступность;
- гео-распределённость с сомнительной внутренней связностью;
- краткие сроки разработки и ввода в эксплуатацию;
- соответствие консервативному бюджету;
- линейное масштабирование по всем характеристикам на два порядка;
- максимальная простота эксплуатации;
- continuous self-healing, on-demand self-desctruction.

Мы привлекли инженеров с большим опытом в предметной области, изучили комплексные и частичные решения от таких лидеров отрасли, как Sun/Oracle, Isilon, IBM, DataDirect Networks, NetApp, Panasas, Xyratex, Backblaze и других. Мы столкнулись с тем, что объективные российские реалии выдвигают к распределённым системам хранения данных такие требования, под которые не разрабатывалось ни одно из ведущих мировых решений. В итоге мы были вынуждены разработать и запустить собственное файловое хранилище.

В качестве ключевых технологий были использованы ОС FreeBSD и файловая система UFS2 и IP для коммутации. Для передачи данных пока хорошо себя показывает HTTP (nginx в качестве target), а прототип диспетчера данных работает на базе данных PostgreSQL и языках sh и python.

Хранилище находится на стадии раннего прототипа, но уже успешно решает производственные задачи. О подробностях реализации и ближайших планах — в докладе.

Использование 0MQ для построения low latency-распределенных систем

Андрей Охлопков, Алексей Ермаков (Global Hedge Capital Group)

1. Преимущества и недостатки построения многомодульных систем с использованием обмена сообщениями.
2. Протоколы очередей сообщений (XMPP, JMS, AMQP).
3. Why 0MQ? (низкоуровневость, децентрализованность, производительность, роутинг).
4. 0MQ API (общие принципы, различные типы сокетов и их взаимодействия, devices).
5. Недостатки (клиентские библиотеки, недостатки в API).
6. Case study: получение и обработка биржевых котировок.

О докладчиках

Андрей Охлопков, руководитель проектов.

2006 – 2009 — ЗАО «Акцент». Разработка приложения для автоматической торговли. Интеграция с PATSystems, СМЕ;

2009 — GHCG, Руководство проекта по разработке приложения для автоматической торговли. Интеграция с различными брокерами и вендорами данных.

Образование: МГУ им. Ломоносова, ВМиК.

Алексей Ермаков, - старший разработчик.

2006-2009 — Sytech, разработка информационно-аналитической системы «Арион»;

2009-2010 — GHCG, разработка системы автоматической торговли ценными бумагами.

Образование: МГУ им. Ломоносова, ВМиК.

Интеграция открытых технологий и взаимодействие со сторонними проектами в условиях высоких нагрузок

Олег Илларионов (ВКонтакте)

1. Создание высоконагруженного Jabber-сервера с нуля за месяц.
Почему мы решили не использовать существующие технологии, и как ведёт себя node.js под нагрузкой.
2. Кроссдоменное взаимодействие (Виджеты, IFrame-приложения).
Какими способами можно передавать информацию между родительским и дочерним окнами на разных доменах, не используя серверную сторону и не теряя кроссбраузерность.
3. О взаимодействии с внешним миром (Twitter, RSS и новые направления).

О докладчике

Студент ЛЭТИ (5 курс). В ВКонтакте занимаюсь самыми разными вещами, но в основном интересуют API, Jabber, Виджеты, экспорт в Twitter, импорт rss в группы.

Shared Personalization Service — How to Scale to 15K RPS

Patrice Pelland (Microsoft)

Covering the architecture of Web Services for the MSN Shared Personalization Service to scale to the load of MSN Home Pages around the globe. The services were designed to sustain peaks of 15K RPS.

О докладчике

Patrice Pelland is a principal developer manager at Microsoft with more than a dozen years of experience designing software. He is also an experienced mentor and an instructor of programmers at all skill levels.

Практическое создание крупного масштабируемого web 2.0 проекта с нуля

Дмитрий Бородин (Сонетика.ру, Лице-Мер)

- О чем не стоит думать, когда вы решили сделать web 2.0 проект. Неправильные вопросы новичка в web 2.0, имеющего огромный опыт программирования: какой выбрать фреймворк, базу данных, язык программирования;
- Важное отличие термина Highload от честного горизонтального масштабирования;
- Для построения web 2.0 проекта нужно думать о масштабировании, а не о Highload;
- Держать высокую нагрузку можно даже ужасным кодом, если реализовать масштабирование;
- Кто умеет строить масштабируемые web 2.0 проекты? Никто, кроме тех, кто уже имеет крупные социальные сети;
- Примеры типичных задач (страниц) в web 2.0, которые технически невозможно реализовать обычными подходами в программировании. Те самые причины, почему ваш стартап гарантированно обречен на провал, если не решать их с нуля и правильно;
- Обзор технологий программирования (администрирования), которые кардинально придется поменять при создании web 2.0 проекта. Забудьте много важного, что вы умели профессионально делать раньше;
- Основные известные тезисы для создания Highload-проекта;
- Основная задача, о которой нужно думать при создании горизонтально масштабируемого web 2.0 проекта, — как хранить и оперировать данными;
- Если выбирать между базами данных, что выбрать? Не слишком ли зазорно использовать MySQL?
- Выбор базы данных и языка почти ни на что не повлияет, проблемы масштабирования одни и те же. Используйте самые простые технологии — nginx, php-fpm, MySQL, Memcache. Опасность экспериментов с новомодными хранилищами данных;
- Важность тщательного изучения выбранных инструментов. Знание memcache на уровне команд set/get — это 0,5 % того, что надо. Знание наизусть документации и применение команд add/cas — это 5 % того, что надо знать. Memcache — это не кеш!
- Обзор уникальных возможностей Memcache в программировании: версионность сессий и их параллельная обработка, блокировки, параллельная правка больших массивов в одном ключе Memcache. Огромные минусы Kefata и ее альтернатива;
- Основной перечень проблем в большом веб 2.0 проекте. Как найти случающиеся затыки: интернет-канал, внутренний трафик, лимиты жестких дисков, журнал транзакций SQL, очереди, application-сервера, memcache, php/php-fpm и т. д.;
- Описание технических параметров проекта Лице-Мер, который занимает 6 место в Рунете в категории «Общение» по ежедневной посещаемости (крупнее только 3 соцсети, такие как ВКонтакте и Мой мир). Нагруз-

ка: 900 000 «уников» в сутки, трафик 400 Мбит/с, 20 000 SQL-запросов в секунду, около 80 слабых серверов;

- Применение на практике: nginx, php-fpm, php 5.3 (уже 1,5 года), Memcache, Redis, RabbitMQ, собственные патчи. Не существующий нигде, кроме нашей компании, работающий PECL модуль к REDIS для PHP 5.3;
- Методы обработки огромных потоков данных и поиск затыков;
- Плюсы и минусы выбранного нами хостера: дешевизна, выгода от оплаты голосами, проблемы (название хостера не публикую);
- Небольшая итоговая инструкция начинающим стартаперам по методу хранения и оперирования данными. При ее соблюдении, если внезапно ваш проект «выстрелит», вы сможете с минимум потерь успевать за ростом.

Системное администрирование

Мониторинг: XXI век

Смирнова Алиса, Дмитрий Никоненко (Яндекс)

Каждый час по всему миру из строя выходит не один десяток серверов.

Тысячи администраторов следят за их работоспособностью, однако поломки неизбежны. Как узнавать об этом своевременно, а лучше заранее? На помощь приходят математические методы, хорошо зарекомендовавшие себя в мониторинге эпидемий, землетрясений и многого другого. Мы расскажем о том, что общего между выходом из строя Интернет-сервиса и землетрясением. Вы узнаете, как из сложной математики получить простую и эффективную систему мониторинга и как она применяется в «Яндексе» сейчас.

Некоторые аспекты влияния сходимости протокола BGP на доступность сетевых ресурсов

Александр Азимов (Highload Lab)

Тема работы относится к актуальной области администрирования на междоменном сетевом уровне — моделированию процесса сходимости протокола BGP *1+. Данный тип моделирования активно применяется при разработке новых механизмов передачи сообщений BGP и при оценке скорости перестроения глобальной сети вследствие возникновения нештатных ситуаций.

Под временем сходимости протокола BGP понимается время перехода сегмента сети BGP-маршрутизаторов из одного согласованного состояния в другое вследствие изменения настроек BGP-маршрутизаторов. В рамках проведенного исследования были получены следующие оценки:

- Время сходимости протокола BGP вследствие объявления нового маршрута BGP-маршрутизатором пропорционально диаметру графа сети относительно рассматриваемого маршрутизатора;
- Время сходимости протокола BGP вследствие удаления маршрута BGP-маршрутизатором пропорционально гамильтону пути в графе относительно рассматриваемого маршрутизатора.

В рамках исследования был также проведен анализ работы механизма Flap Damping *2+. Данный инструмент BGP разработан для обнаружения, локализации и минимизации влияния на сеть АС нестабильных участков сети. В качестве признака нестабильности участка сети используется «мигающий маршрут»: повторяющиеся объявления и удаления маршрута к некоторому префиксу за короткий интервал времени. Используя полученные оценки времени сходимости протокола BGP, была рассчитана вероятность возникновения «мигающего маршрута» в кольце BGP-маршрутизаторов вследствие единичного удаления маршрута. В соответствии с полученными результатами, был сделан вывод о негативном влиянии механизма Flap Damping на сходимость протокола BGP.

Учитывая невозможность сбора репрезентативной статистики, для проверки теоретических оценок необходимо было найти или реализовать собственную систему моделирования маршрутизации протокола BGP. В рамках решения данной задачи было проведено исследование существующих систем моделирования маршрутизации BGP и предложена архитектура, позволившая одновременно использовать системы моделирования PRIME *3+ и CBGP *4+. В результате была разработана новая система моделирования маршрутизации BGP, превосходящая по своей функциональности существующие аналоги.

Результаты экспериментов с использованием разработанной системы моделирования маршрутизации BGP подтвердили корректность теоретических оценок времени сходимости BGP-маршрутизации.

1. RFC 4271, A Border Gateway Protocol 4 (BGP-4), [HTML] (<http://www.faqs.org/rfcs/rfc4271.html>).
2. RFC 2439, BGP Route Flap Damping, [HTML] (<http://www.faqs.org/rfcs/rfc2439.html>).

3. Parallel Real-time Immersive network Modeling Environment (PRIME), [HTML] (<https://www.primesf.net/bin/view/Public/PRIMEProject>).
4. CBGP, [HTML] (<http://cbgp.info.ucl.ac.be/>).

Динамика DDoS-атак в России

Александр Лямин (Highload Lab)

В докладе рассматривается развитие механизмов атак и изменение целей нападения. Обсуждается классификация типов атак и свойств ботнетов в ретроспективе прошедшего года.

Классификация типов атак: почему одной метрики, например, скорости, явно недостаточно. Свойства ботнета в сравнении с тем, какими мы видели их год назад. Новые цели и задачи нападающих: и причем здесь Яндекс?

Расследования DDoS, примеры и результаты. Технический подход к проблеме. Основные направления развития на следующий год: новые сервисы, новые методики, новые скорости.

Тандемные DDoS-атаки / Проблематика уязвимостей в спецификации TCP/IP (фундаментальные уязвимости)

Артем Гавриченко (Highload Lab)

Широко известная атака на отказ в обслуживании протокола TCP — SYN flood — хорошо изучена, существуют популярные методы борьбы с ней, включая, например, технологию SYN cookies [1]. Важным свойством этого вида атак является их влияние на модули отслеживания соединений (connection tracking). Даже при использовании SYN cookies каждый входящий TCP-сегмент с выставленным флагом SYN создаёт запись в базе отслеживаемых соединений, что спустя некоторое время может привести к переполнению базы и потере новых запросов на соединения.

Менее известная атака, носящая кодовое название FIN-WAIT-2 attack, эксплуатирует особенности алгоритма закрытия TCP-соединения. Данная атака подробно разбирается в [2]. Её основная цель — исчерпание памяти, используемой TCP-соединениями, находящимися в фазе FIN-WAIT-2 закрытия соединения. При определённых условиях соединение может находиться в этом состоянии продолжительное время, при этом все ресурсы, задействованные соединением, не будут доступны активным компонентам атакуемой системы.

В конце 2008 года исследователями компании Outpost24 была обнаружена уязвимость автомата работы TCP-соединения. Proof of concept этой уязвимости известен под именем Sockstress. DDoS-атака, основанная на этой уязвимости, предположительно функционирует за счёт имитации работы SYN cookies локально на каждом элементе ботнета и особенностей работы алгоритмов борьбы с перегрузкой, включённых в спецификацию протокола TCP. Хотя сам proof of concept не был опубликован в силу нежелания исследователей передавать в руки злоумышленников столь мощный инструмент для проведения DDoS-атак, ряд ведущих производителей сетевого программного и аппаратного обеспечения был с ним ознакомлен. Существуют работы, документирующие обнаруженные Outpost24 уязвимости, а также опубликованы рекомендации по борьбе с такими атаками от исследующих их корпораций, таких, как Red Hat, Cisco, Microsoft [3].

Рекомендации вендоров ПО и аппаратных решений по борьбе с атаками FIN-WAIT2 и Sockstress включают внедрение в файрвол механизма отслеживания соединений и фильтрацию вредоносного трафика на основе анализа устанавливаемых и установленных соединений.

Данный доклад описывает потенциальную тандемную атаку, включающую в себя одновременно SYN flood и одну из атак на FIN-WAIT-2 или Sockstress. Иллюстрируется эффективность такой атаки против большинства компьютерных систем, настроенных в соответствии с рекомендациями вендоров. Предлагаются альтернативные решения по борьбе с тандемной атакой.

Список литературы

1. Dave Dittrich, Some TCP/IP Vulnerabilities: Weaknesses, attack tools, defenses [HTML] (<http://staff.washington.edu/dittrich/talks/agora/index.html>).

2. CPNI Technical Note 3/2009: Security Assessment of the Transmission Control Protocol [PDF] (<http://www.cpni.gov.uk/Docs/tn-03-09-security-assessment-TCP.pdf>), 2009.
3. CVE-2008-4609 - CVE - Common Vulnerabilities and Exposures (CVE) (<http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-4609>), 2008.

Performance tweaks and tools for Linux

Joe Damato

Доклад разделен на две части. В первой части мы рассмотрим особенности различных настроек производительности ядра Linux и стандартные драйвера. Разговор пойдет о некоторых малоизвестных настройках ядра, подстройка которых поможет наиболее эффективно использовать аппаратные средства железа.

Во второй части доклада разговор пойдет о полезных инструментах для изучения и диагностики ошибок в процессах. Мы пробежимся по различным инструментам для Linux (strace, ltrace, oprofile, gdb, и другие) и разберем, как их можно использовать для получения представления о том, на что тратятся ресурсы и время обработки запросов.

Если останется время, мы изучим пути «посмертного» анализа как инструмента лучшего понимания того, что пошло не так.

Целевая аудитория

По итогам каждый участник сможет понять, как ему оптимизировать настройки производительности к загрузенности системы; сможет лучше представить, как работают приложения и по какой причине произошел сбой.

О докладчике

Джо Домато (Joe Damato) — системный хакер, специализирующийся на низкоуровневом взломе, в особенности I/O, производительности, тестирования, масштабирования, безопасности и взлома kernel.

Автор ведет блог (<http://timetobleed.com>), в котором описывает релизы кода, патчи для интерпретаторов MRI Ruby и делится мыслями о низкоуровневом программировании. Джо добавил поддержку выполнения динамического компоновщика ltrace в Linux. Также он пишет memprof Ruby level memory profiler (<http://github.com/ice799/memprof>), который помогает отслеживать пользователям дыры в своих приложениях. Джо работает в MRI и REE Ruby VM-ом, выпуская патчи по оптимизации процессов на Ruby 1.8, и работает над Fibers для Ruby 1.8.

Cloud APIs: обзор API западных провайдеров и API Scalaxy

Нат Гаджибалаев (Scalaxy)

Многие провайдеры на рынке облачной инфраструктуры предоставляют свои API для использования либо для разработки, либо для администрирования серверной инфраструктуры. Это одно из преимуществ облачного хостинга — возможность управлять серверами через API предельно просто. В Скалаксии мы тоже создали специальный API для работы с инфраструктурой.

Заставить java-приложение работать в рамках одной виртуальной машины и править размер этой виртуальной машины из кода на java, запрашивая и освобождая ресурсы на лету? Это уже возможно.

В докладе я расскажу, какие API представлены на мировом рынке, зачем они были созданы, как с ними работать. Мы сравним API Скалаксии с ведущими мировыми аналогами, покажем, чего можно добиться, используя API уже сейчас, поговорим про развитие этой технологии в будущем и ее приложения уже сейчас.

СМЕЖНЫЕ ТЕХНОЛОГИИ

Native Client

Евгений Эльцин (Google)

Google Native Client (<http://code.google.com/p/nativeclient/>) — это исследовательский проект с открытым исходным кодом, целью которого является использование native кода в web-приложениях с сохранением ожидаемых от web-приложений переносимости и безопасности. Важнейшим свойством проекта является поддержка технологий, ориентированных на производительность, обычно недоступных в существующих системах программирования для web-приложений. Это такие технологии, как multithreading, расширения системы машинных команд наподобие x86 SSE, встроенные функции компиляторов и ассемблерные вставки. Все в целом представляет собой открытую архитектуру, спроектированную с учетом поддержки существующих web-стандартов. Мы рассчитываем на активное участие сообщества в тестировании и развитии данной технологии. Целью наших усилий является новая жизнь существующих desktop-приложений в web, появление принципиально новых web-приложений и возможность гибко перераспределять нагрузку между клиентом и сервером.

Целевая аудитория доклада:

разработчики web-приложений с высокой нагрузкой на клиента (например, игры, в том числе 3d), а также желающие использовать в web-приложении код на C/C++.

Yota Access: Система для раздачи абонентам информации о загрузке базовых станций

Кирилл Коринский (Yota)

Сетью Yota пользуются уже больше 600 тысяч человек. Часто люди хотят знать не только качество радио-сигнала (RSSI/CNIR), но и загрузку базовой станции, услугами которой они пользуются. Я расскажу, как создавался и работает данный сервис.

2.5D игры и особенности разработки многопользовательских игр

Глеб Полушкин (SM&Partners)

1. Распределение логики между клиентом и сервером:
 - Какая логика нужна?
 - Какую логику отдать клиенту?
 - Какую логику отдать серверу?
 - Кто сильнее: сервер или клиент?
2. Допустимые упрощения в построении игрового мира:
 - упрощения графики игрового мира;
 - упрощения, допустимые для редактирования;
 - метод обоснованного обмана пользователя;
3. Персонажи: как не испортить ими идеальный игровой мир:
 - выбор варианта сборки персонажа под случай оценка трудозатрат на 1 миллион персонажей;
 - клиентская сборка векторного персонажа;
 - серверная сборка векторного персонажа;
 - клиентская сборка растрового персонажа;
 - серверная сборка растрового персонажа;
4. Масштабируемость серверной части:
 - общие возможности SmartFoxServer по нагрузкам;
 - кластер с общей базой данных;
 - кластерное решение от Terracotta;
 - SmartFoxServer Amazon EC2 AMI.

Тестирование

Нагрузочное тестирование без границ

Юрий Ковалёв (Performance Lab)

1. Существует много факторов, которые могут помешать нагрузочному тестированию.
2. Например, отсутствие поддержки клиент-серверного протокола инструментами нагрузочного тестирования.
3. В докладе рассмотрены два таких случая и подробно разобраны технические подходы к решению:
 - возможные способы перехвата и дальнейшего использования трафика (расширенное логирование, снифер, реверс-инжиниринг jdbc-драйвера);
 - универсальное решение с использованием технологии Citrix XenApp.

Сервис нагрузочного тестирования ddosme.ru

Иван Самсонов (Scalaxy)

Растущим веб-проектам необходимо не просто быть уверенными, что приложение работает, а точно знать, какой рост аудитории приложение выдержит, чтобы понять, когда пора заняться рефакторингом и оптимизацией архитектуры. Иными словами, кроме юнит-тестирования и интеграционных тестов, необходимо также проводить нагрузочное тестирование проекта.

В этом докладе мы поговорим о том, какие инструменты нагрузочного тестирования сейчас доступны, каких результатов можно достичь с их помощью, а так же представим вам наш новый сервис нагрузочного тестирования: ddosme.ru. Расскажем, как он построен, с какими сложностями мы столкнулись в его разработке, покажем, что он умеет и как с ним работать, какие результаты можно получить с его помощью.

- Краткий обзор существующих сервисов тестирования;
- Обзор интерфейса и функционала ddosme;
- Архитектура сервиса, почему он качественно и дешево тестирует;
- Работа с сервисом, небольшое howto;

Базы данных и системы хранения

Developing PostgreSQL Performance

Simon Riggs (PostgreSQL)

A short history of the development of performance features in PostgreSQL. Talk will cover low level techniques, design, use case analysis, current status and review of short term (9.1) and long term further opportunities (9.2+).

Managing Replication of PostgreSQL

Simon Riggs (PostgreSQL)

Why replicate? A look at the various use cases for replication. How replication works, when and how it doesn't work and what to do. Configuration tools and options to make replication work for you. Covers streaming replication, Hot Standby and future options.

The Magic of Hot Streaming Replication

Bruce Momjian (PostgreSQL)

This talk explores the much-anticipated Postgres 9.0 features of hot standby and streaming replication. It explains how these features work, how to configure them, and their current limitations. It includes a hands-on demonstration that can be done either by the instructor or by students.

Rapid Upgrades With Pg_Upgrade

Bruce Momjian (PostgreSQL)

Pg_Upgrade allows data to be transferred between major Postgres versions without a dump/restore. It does this by transferring the user data and version-dependent data separately. This talk explains the internal workings of pg_upgrade and includes a pg_upgrade demonstration.

Сравнительный анализ хранилищ данных

Олег Царев, Кирилл Коринский (Percona, Yota)

Существует разные системы для хранения данных: как классические SQL решения, так и модные NoSQL.

Поиск серебряной пули вытекает из ограничений реального проекта. Ограничения зависят как от окружения проекта (кадры, бюджет, сроки), так и от формулировки задачи.

В нашем докладе мы продемонстрируем связь между ограничениями, структурами, алгоритмами, и реализациями различных хранилищ данных.

Вы решили написать собственное хранилище данных

Илья Космодемьянский (Интелотек групп)

Хранение данных. Откуда данные берутся и куда могут пропасть:

- как приходят к идее написания своего хранилища, чем могут не устраивать существующие: типичные случаи;
- постановка задачи.

Идеальный мир, безотказное хранилище:

- атомарность и хранение данных;
- 2PL, граф сериализуемости, связанные теоремы.

Реальный мир, отказы и падения. Разновидности и последствия:

- восстановимость;
- изоляция;
- транзакции;
- управление одновременным взаимодействием и его виды;
- управление одновременным взаимодействием на объектных и реляционных структурах.

Производительность и масштабирование:

- бойтесь масштабирования!
- распределенные транзакции, сопутствующие алгоритмы;
- управление распределенными транзакциями: варианты реализации;
- очереди;
- CAP-теорема.

SQL-интерфейс:

- вкус и цвет, интеграция с системой-потребителем;
- NoSQL: большая подмена понятий.

Нерассмотренные важные моменты:

- сеть и производительность;
- мониторинг;
- поддержка.

Возвращаемся к постановке задачи.

Sphinx для высоконагруженных и масштабируемых проектов

Вячеслав Крюков (Ivinco)

В этом докладе изложен опыт разработки системы полнотекстового поиска на основе Sphinx для масштабируемого и высоконагруженного проекта BoardReader.com.

Рассматриваемые темы:

- Архитектура системы полнотекстового поиска.
- Масштабируемость проекта, многоуровневая распределенность индексов Sphinx.
- Производительность запросов Sphinx.
- Влияние факторов сбалансированности и обновления данных на производительность.
- Обеспечение согласованности с индексами Sphinx данных при их обновлении.
- Меры для повышения производительности и надежности.
- Перспективные Sphinx фичи.

О докладчике

Опыт работы: web-разработчик с 2000 г., работал в компаниях SpyLOG, Percona, в настоящее время web-разработчик в компании Ivinco.

Сравнение, тестирование и миграция из обычных индексов в real time индексы Sphinx Search

Ярослав Ворожко (Ivinco)

В докладе рассмотрена проблема индексации данных в Sphinx версии 0.9.9 и решение этой проблемы с помощью real time-индексов в Sphinx версии 1.10.

Опыт тестирования и миграции на RT индексы был пройден в проекте LJSeek.com. LJSeek.com — это поисковая система по LiveJournal страницам, на данный момент проиндексировано около 60 млн записей.

В докладе рассмотрены такие аспекты, как:

- проблемы обычных индексов;
- новые Real Time индексы;
- сравнение производительности обоих типов индексов;
- демонстрация и конфигурация RT индексов;
- схемы миграции из обычных индексов в RT индексы;
- пример миграции на основе проекта LJSeek.com;
- проблемы миграции, которые возникли в проекте LJSeek.com

О докладчике

Опыт работы в веб-разработке начиная с 2005 года, специализируется на поисковых технологиях, в частности на Sphinx Search, с 2008 года.

Работал в таких компаниях, как Varien-Magento, Persona и в настоящее время работает в компании Ivinco.

Масштабируемая система голосования на базе PostgreSQL PgQ

Сергей Нековаль (Грамант)

Доклад посвящен построению системы голосования для японского видеохостинга с трафиком порядка 10 млн событий в сутки. Рассматривается применение Skytools PgQ как готового и удобного в использовании инструмента агрегации событий, с упором на практические вопросы, проблемы и тонкости.

Предварительные тезисы:

1. Что требуется от системы голосования: виды событий, подсчет голосов, статистика.
2. Какие сложности и как это сделать.
3. Обзор Skytools PgQ. Асинхронность, формат событий, хранение событий, ротация таблиц, агрегация, механизм retry.
4. Построение очередей событий для системы голосования.
5. Обработка и агрегация событий.
6. Production-система как регулируемый конвейер. Подстройка PgQ. Некоторые варианты настройки PostgreSQL.
7. Как это масштабируется.

О докладчике

Работает в компании «Грамант» системным архитектором, занимается проектированием интернет-приложений, а также аудитом самых разнообразных production-систем.

Организация хранилища с vast sky

Дмитрий Лоханский (Scalaxy)

Для стабильной работы «облака», нам было необходимо построить систему хранения данных достаточно быструю, отказоустойчивую, с репликацией данных в реальном времени. При этом решение должно быть легко масштабируемым.

На базе VastSky мы построили систему хранения данных, отвечающую нашим требованиям:

- Достаточно высокая скорость работы;
- Репликация данных в реальном времени;
- Отказоустойчивость всей системы, гарантия сохранности данных и бесперебойности работы;
- Масштабируемость;

Мы расскажем про архитектуру нашей системы хранения, тонкости настройки, мониторинга и масштабирования подобных систем.

InnoDB: архитектура транзакционного хранилища

Константин Осипов (Oracle)

Доклад освещает архитектуру всемирно известного транзакционного хранилища MySQL с точки зрения разработчика систем хранения данных. На примере широко используемой реализации освещаются принципы и проблемы обеспечения надёжности хранения, реализации параллельного доступа к данным и механизма блокировок, подходы к решению задач резервного копирования и восстановления данных.

В докладе рассматривается:

- основы теории реляционных транзакционных систем;
- физический формат данных InnoDB;
- структуры данных в оперативной памяти, их назначение;
- процессы и задачи;
- основные конфигурационные параметры.