Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего профессионального образования

**«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»**

Институт математики и компьютерных наук
Кафедра математической экономики

ВЕБ-СИСТЕМА ВИЗУАЛИЗАЦИИ, АНАЛИЗА И МОНИТОРИНГА РАБОТЫ ПРОГРАММ

Допустить к защите

зав. кафедрой
математической экономики
кандидат физико-математических
наук
Магаз Оразкимович Асанов

Квалификационная работа на степень магистра наук
по направлению «Прикладная информатика»
студента группы МГПИ-2
Согомоняна Михаила Сергеевича

Научный руководитель
кандидат технических наук
Авербух Владимир Лазаревич

Екатеринбург
2013

РЕФЕРАТ

Согомонян М.С. ВЕБ-СИСТЕМА ВИЗУАЛИЗАЦИИ, АНАЛИЗА И МОНИТОРИНГА РАБОТЫ ПРОГРАММ, дипломная работа: стр.40, рис. 21, табл. 2, библ. 5 назв., приложений 1.

Ключевые слова: СБОР И ХРАНЕНИЕ ДАННЫХ О РАБОТЕ ПРОГРАММ, ВИЗУАЛИЗАЦИЯ И АНАЛИЗ БОЛЬШИХ ОБЪЕМОВ ДАННЫХ.

Рассматривается проблема сбора и анализа данных о работе программ различного типа. Выявляются недостатки существующих решений проблемы, выдвигаются требования к решению, призванному их устранить.

В результате разработано серверное программное обеспечение, предназначенное для накопления и хранения данных о работе программ и предоставляющее визуальные средства для их анализа. Также созданы простые и удобные в использовании клиентские библиотеки, служащие для доставки данных о работе программ на сервер.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 4
Описание проблемы и ее актуальность 4
Краткий обзор состояния проблемы 4
Постановка задачи и полученный результат 8
ОПИСАНИЕ РАБОТЫ ПРОГРАММЫ 9
Общий принцип устройства системы 9
Отправка событий 10
Просмотр событий 12
Условия и оповещения 15
Внутренние события системы 17
ДОПОЛНИТЕЛЬНЫЕ СРЕДСТВА ВИЗУАЛИЗАЦИИ СТАТИСТИКИ 19
Инструмент фильтрации событий 20
Количество событий. Распределение количества событий между сериями 21
Отклонение скорости потока событий от среднего значения 22
ДЕТАЛИ РЕАЛИЗАЦИИ 25
Средства для разработки 25
Установка системы на сервере разработчика 25
Принцип именования событий. Алгоритм преобразования списка имен в дерево 26
ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ 30
Пример №1. Анализ востребованности функции «Поиск изображения» 30
Пример №2. Контроль процесса резервного копирования базы данных 32
Пример №3. Выявление хакерских атак 35
ЗАКЛЮЧЕНИЕ 37
СПИСОК ПУБЛИКАЦИЙ 38
СПИСОК ЛИТЕРАТУРЫ 39
ПРИЛОЖЕНИЕ 1 40

ВВЕДЕНИЕ

Описание проблемы и ее актуальность

Разработчики программного обеспечения на определенном этапе передают свое творение конечным пользователям или размещают его на сервер. В процессе эксплуатации программы у разработчика возникает естественный интерес, связанный с тем, как используется его программа, а именно:

- Какие функции в ней востребованы;
- Как часто они используются;
- Каким числом людей;
- С какими параметрами они запускаются;
- Частота возникновения ошибок и их суть.

Обладая такого рода информацией, создатели могут понимать работу своих программ, проверять корректность этой работы и принимать решения о дальнейшем развитии программы.

Аспекты визуализации процесса и параметров работы программ в научном плане изучает область «визуализация программного обеспечения» [1], которая особенно активно развивается на западе [2].

Настоящая работа описывает наш опыт разработки в области сбора и визуализации статистики работы программ.

Краткий обзор состояния проблемы

Проблема сбора и визуализации работы программ является актуальной как для обычных (настольных или мобильных) приложений, исполняющихся на устройствах пользователей, так и для клиент-серверных систем.

На практике, сбор информации о работе серверного приложения осуществляется путем создания записей о происходящих в нем событиях в специально отведенном текстовом файле – логе. Например, в лог веб-сервера

может записываться информация как о работе самой системы (сведения об ошибках или предупреждениях), так и о поведении пользователей (список запрошенных пользователем ресурсов, продолжительность сеанса, IP-адрес компьютера пользователя).

При активном использовании сервера пользователями или очень подробном журналировании или просто продолжительном времени работы программного обеспечения, в логе становится очень много записей, что создает трудности для его чтения и анализа. Более того, если программу обслуживает несколько серверов, количество логов, подлежащих исследованию, возрастает, что в еще большей степени усложняет анализ данных.

Наиболее распространенным решением этой проблемы является написание программы, которая:

1. Осуществляет извлечение логов с серверов, обслуживающих серверное приложение, выполняет их объединение и хранение.
2. Предоставляет инструмент фильтрации данных.
3. Отображает информацию в текстовом или графическом виде, удобном для анализа.

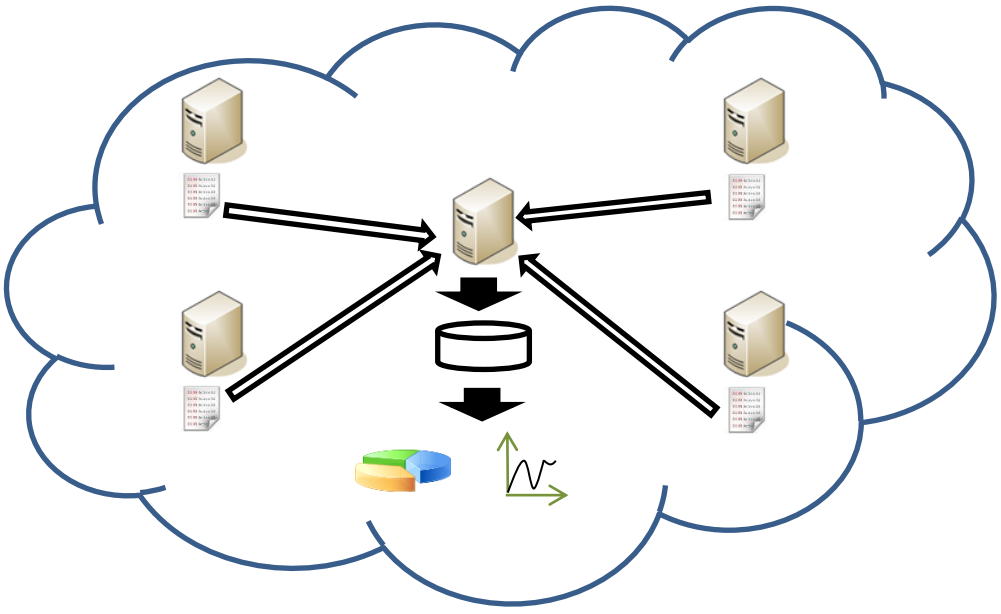


Рис 1. Схема устройства сбора, хранения и визуализации данных о работе серверной программы.

Сбор статистики о работе настольного приложения представляет собой более сложную задачу: необходимо организовать сбор и хранение данных о работе программы на компьютере пользователя, обеспечить доставку этих данных на удаленный сервер, предназначенный для их хранения и анализа. При этом на сервере должна работать программа, которая:

1. Принимает и хранит поступающую информацию.
2. Предоставляет инструмент фильтрации данных.
3. Отображает информацию в текстовом или графическом виде, удобном для анализа.

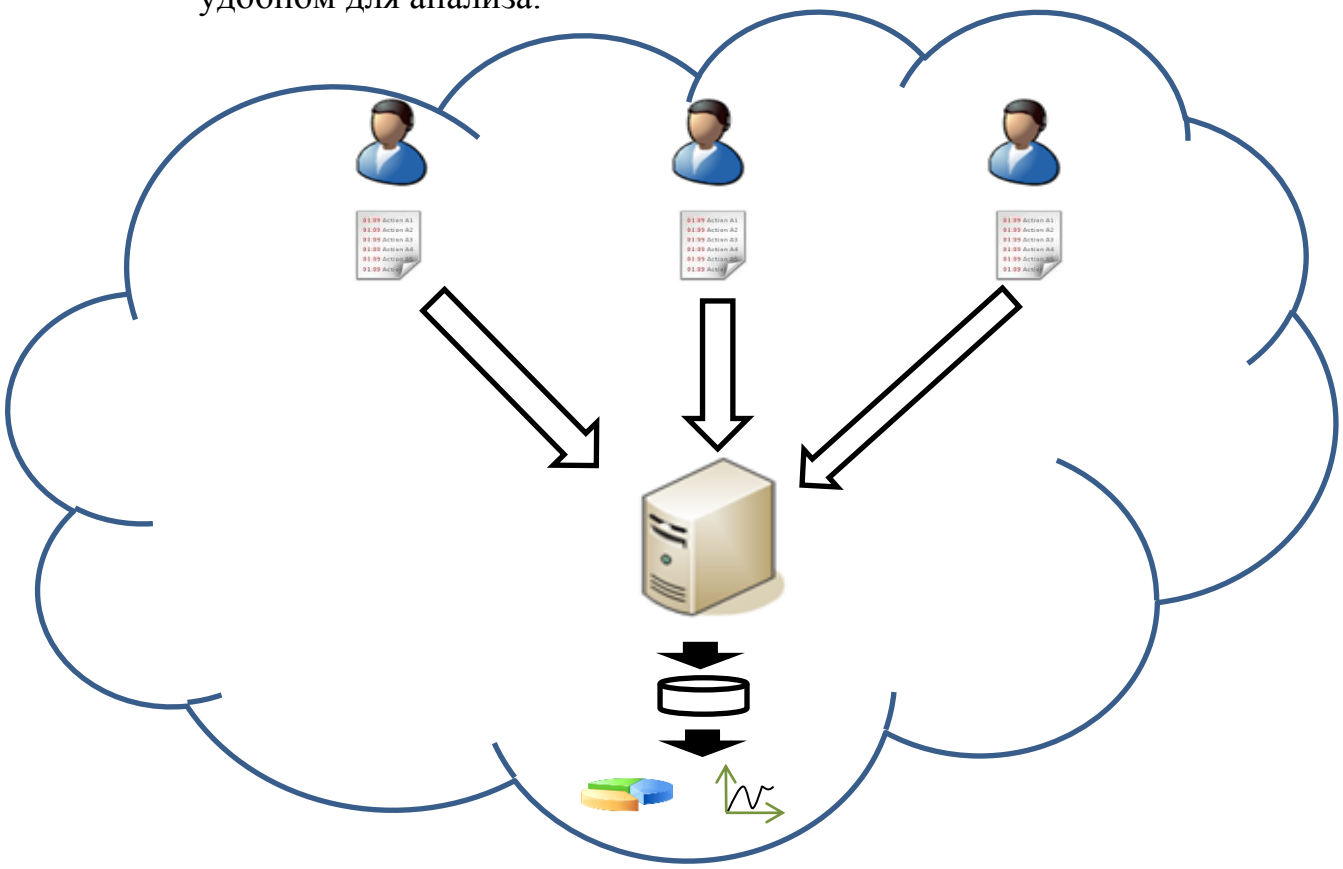


Рис 2. Схема устройства сбора, хранения и визуализации данных о работе настольной программы.

Таким образом, реализация сбора, хранения и визуализации статистики работы программ является нетривиальной многоплановой задачей и требует от разработчика значительных затрат времени и других ресурсов. Однако на рынке программного обеспечения присутствует ряд готовых решений поставленной задачи. Такие веб-сервисы как Loggly (www.loggly.com),

Loggr (www.loggr.net), MixPanel (www.mixpanel.com) предоставляют своим пользователям дисковое пространство своих серверов для хранения данных, а также мощные средства их фильтрации и визуализации, доступные через веб-интерфейс. От пользователя требуется лишь настроить своё приложение таким образом, чтобы данные о его работе автоматически поступали на сервер соответствующего веб-сервиса (Рис. 3). Другим известным решением является программный комплекс Splunk (www.splunk.com), который устанавливается на сервер разработчика, осуществляет сбор данных о работе указанных разработчиком программ и предлагает богатый арсенал средств для анализа.

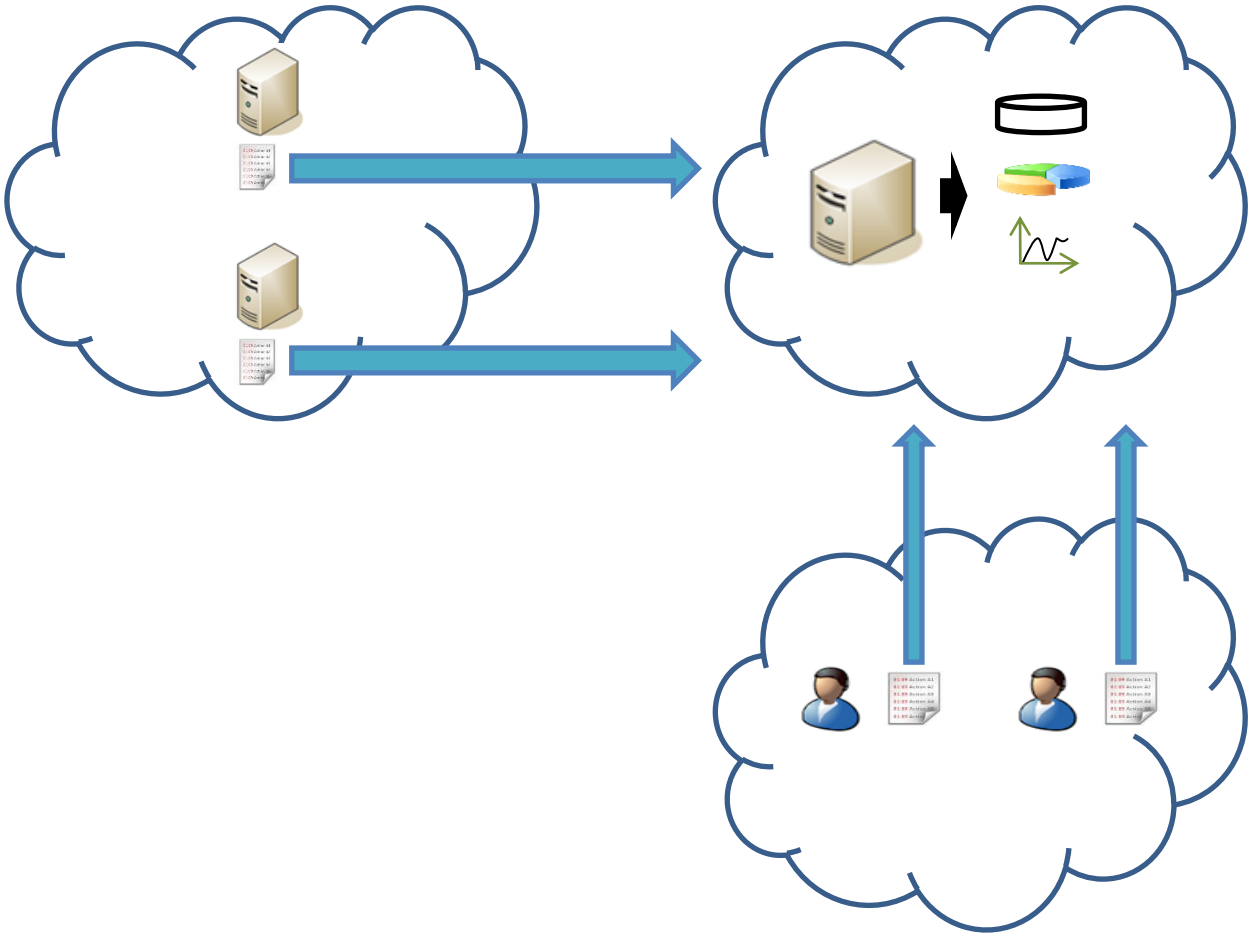


Рис. 3. Отправка данных о работе серверной и настольной программ на сервер стороннего разработчика для хранения и последующего анализа.

Большинство решений, представленных на рынке, являются коммерческими продуктами. Стоимость лицензии на использование подобного программного обеспечения измеряется в сотнях и тысячах

долларов, в то время как бесплатная лицензия существенно ограничивает его функциональность. Бесплатного решения, достаточно функционального и удобного в применении, автором данной работы обнаружено не было.

Также стоит отметить, что использование сторонних продуктов в большинстве случаев связано с передачей данных на сервер другой компании, что неприемлемо в случае конфиденциальности этих данных. Наличие полноценного решения, распространяемого по свободной лицензии, позволило бы разработчику не только запустить такое программное обеспечение на собственном оборудовании, тем самым обеспечив конфиденциальность данных, но и расширить его возможности путем изменения исходного кода.

Постановка задачи и полученный результат

В результате работы над созданием решения, лишенного недостатков конкурентов, была получена система Evented, предназначенная для визуализации, анализа и мониторинга работы других программ. В ее основе лежит возможность:

1. Программным путем наполнять себя информацией;
2. Графически отображать накопленную информацию.

Среди отличительных черт системы можно выделить:

- Открытый исходный код;
- Свободное распространение;
- Простота в применении.

Также обеспечена поддержка следующих возможностей:

- Механизм гарантированной доставки статистики;
- Мощный инструмент структуризации и фильтрации накопленной информации;
- Гибкая настройка визуализации статистики;
- Система условий и оповещений.

ОПИСАНИЕ РАБОТЫ ПРОГРАММЫ

Общий принцип устройства системы

Система Evented предоставляет возможность программным путем наполнять себя информацией (данными о работе других программ) и графически ее отображать.

Работу пользователя с системой Evented можно условно разбить на два этапа:

1. Наполнение системы данными о работе программ пользователя.
2. Анализ накопленных данных при помощи графических средств.

Наполнение информацией происходит путем передачи HTTP-запроса к серверной части Evented. Каждый такой запрос формирует "событие". Событие имеет имя, численное значение, комментарий, дату и время, а также набор произвольных атрибутов вида имя-значение. Таким образом, данные о работе программ хранятся в Evented в виде событий – неделимых структурных единиц информации. Имя события обычно отражает тип данных, которые данное событие содержит. События с одним и тем же именем формируют серию.

По HTTP, через веб-браузер доступен список сформировавшихся серий. Пользователь может выбрать одну или несколько серий, чтобы посмотреть соответствующие события на графике.

Таким образом, есть программа, работающая на сервере, которая с одной стороны позволяет по определенному протоколу события в себя закладывать, а с другой стороны по HTTP же, через веб-браузер визуально их просмотреть.

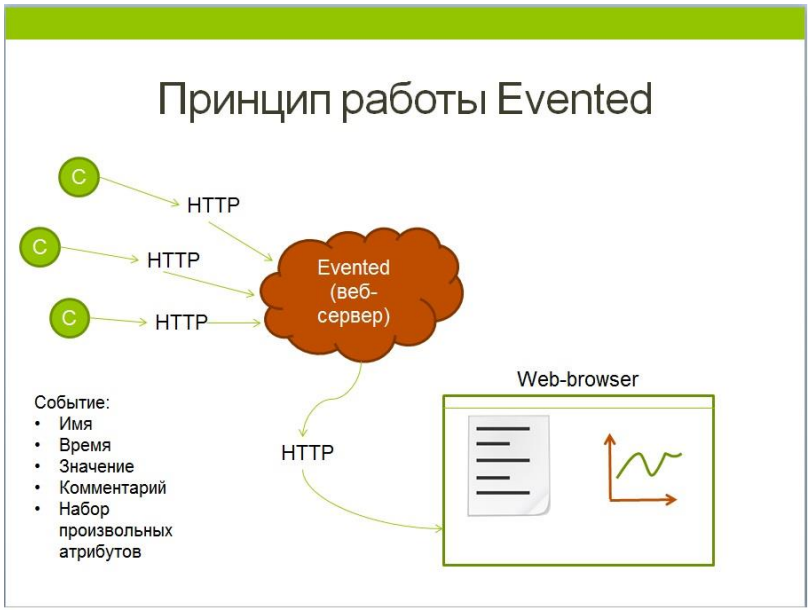


Рис. 1. Принцип работы системы Evented.

Отправка событий

Программист, используя протокол HTTP или специальные поставляемые с системой библиотеки, внедряет в исходный код своих программ функции для формирования событий. Например, это может быть событие «пользователь запустил программу», «пользователь использовал функционал X», «пользователь загрузил файл размером Y», «программа завершена, время работы составило Z минут», и так далее. Таким образом, программист сам указывает, какие элементы логики работы программы его интересуют.

На данный момент вместе с системой поставляются библиотеки для следующих языков программирования:

- Ruby
- Javascript

Каждая такая библиотека представляет собой программный модуль, реализующий функцию push – функцию формирования и отправки события. Разработчик, подключив соответствующую библиотеку к своему проекту, встраивает в исходный код вызовы функции “push” с указанием необходимых параметров. При каждом вызове функции формируется

событие, определяемое пользовательскими параметрами, и отправляется на сервер, на котором работает система Evented. Передача события происходит путем выполнения HTTP-запроса методом POST.

Формат вызова функции “push”, реализованной в библиотеке для языка Ruby, описан в Листинге 1.

Evented.push имя_события, значение_события, [хэш_атрибутов]

Листинг 1. Формат вызова функции “push”.

Обязательными к указанию являются параметры «имя события» и «значение события». Хэш параметров содержит пары («имя», «значение») и служит как для явного указания параметров отправки, так и для задания атрибутов события. Если в паре («имя», «значение») «имя» является зарезервированным (ключевым) словом, то такая пара задает значение параметра отправки. Иначе пара задает имя и значение атрибута. Если параметры отправки явно не заданы, будут использованы значения по умолчанию. Подробная информация о параметрах отправки событий приведена в Приложении 1.

Стоит отметить, что в библиотеке для языка Ruby отдельным элементом проработана логика отложенной записи событий. Такая запись может потребоваться при временном отсутствии связи с Evented. В этом случае события накапливаются в специальный файл, а затем в фоновом режиме отправляются по мере появления возможности.

Еще одним способом отправки события является выполнение HTTP запроса методом GET к специальному ресурсу сервера Evented. При этом необходимо включить в URI, к которому происходит обращение, два обязательных параметра – имя и значение добавляемого события. Пример реализации такого подхода приведен в Листинге 2.

```
<html>
    <head>
    </head>
    <body>
        <img src="http://localhost:5000/push_static/test?value=2" />
    </body>
</html>
```

Листинг 2. Встраивание в HTML-страницу элемента, загрузка которого влечет добавление события с именем «test» и значением «2».

Таким образом, разработчику предоставлены удобные и простые средства для отправки данных о работе его программ. Такие данные, в виде событий, попадают на веб-сервер системы Evented и становятся доступными для анализа.

Просмотр событий

Для анализа накопленных данных пользователь использует веб-интерфейс системы Evented через веб-браузер. Логика работы пользователя с веб-интерфейсом заключается в циклическом выполнении двух шагов:

1. Выбор серии событий для отображения.
2. Просмотр и взаимодействие с графиком событий серии.

В связи с этим, для удобства пользователя веб-интерфейс системы визуально разделен на две области – область серий событий и область графика.

Область серий событий (Рис. 2) состоит из следующих элементов:

- Список серий событий
- Окно настроек списка

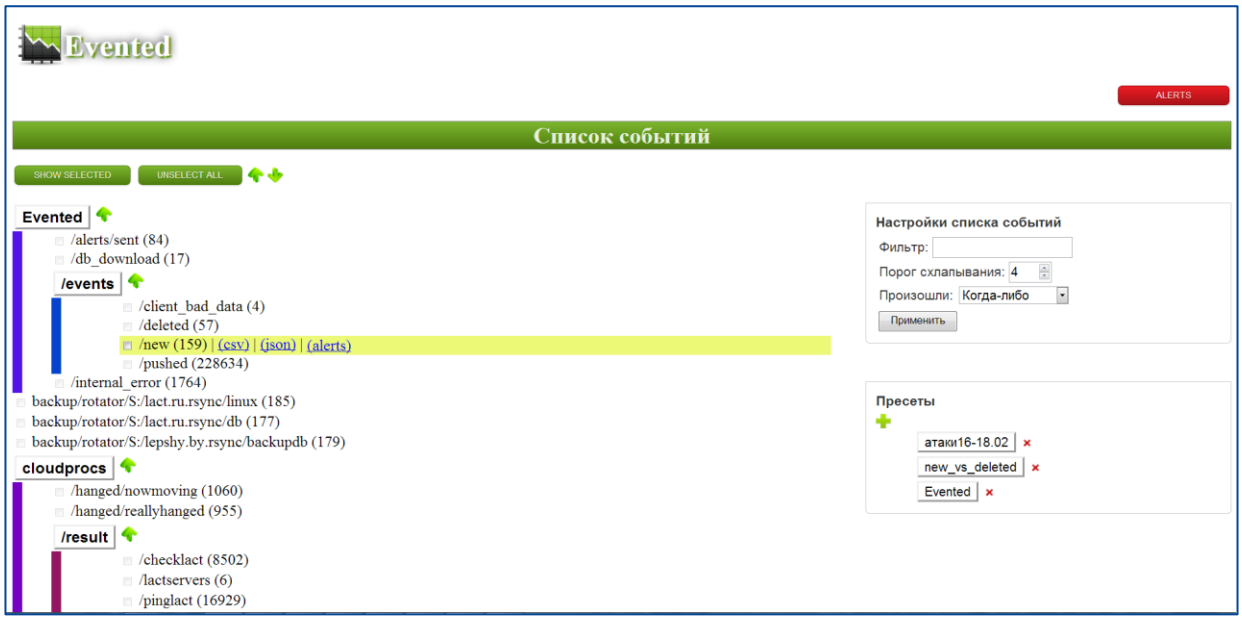


Рис. 2. Область серий событий. Слева расположен список серий событий, справа – настройки списка.

Список серий имеет древовидную структуру, образованную при помощи вычисления общих префиксов в именах серий (описание соответствующего алгоритма приведено в разделе «Детали реализации»). Такой подход, на наш взгляд, улучшает читаемость списка. При помощи настроек списка можно отфильтровать серии по включению именем серии заданной пользователем подстроки, а также указать, чтобы список содержал лишь те серии, события которых происходили на заданном пользователем промежутке времени.

Область графика событий (Рис. 3) имеет следующие компоненты:

- График событий выбранной серии (-ий)
- Окно настройки отображения графика и группировки данных
- Таблица с текстовым представлением данных событий

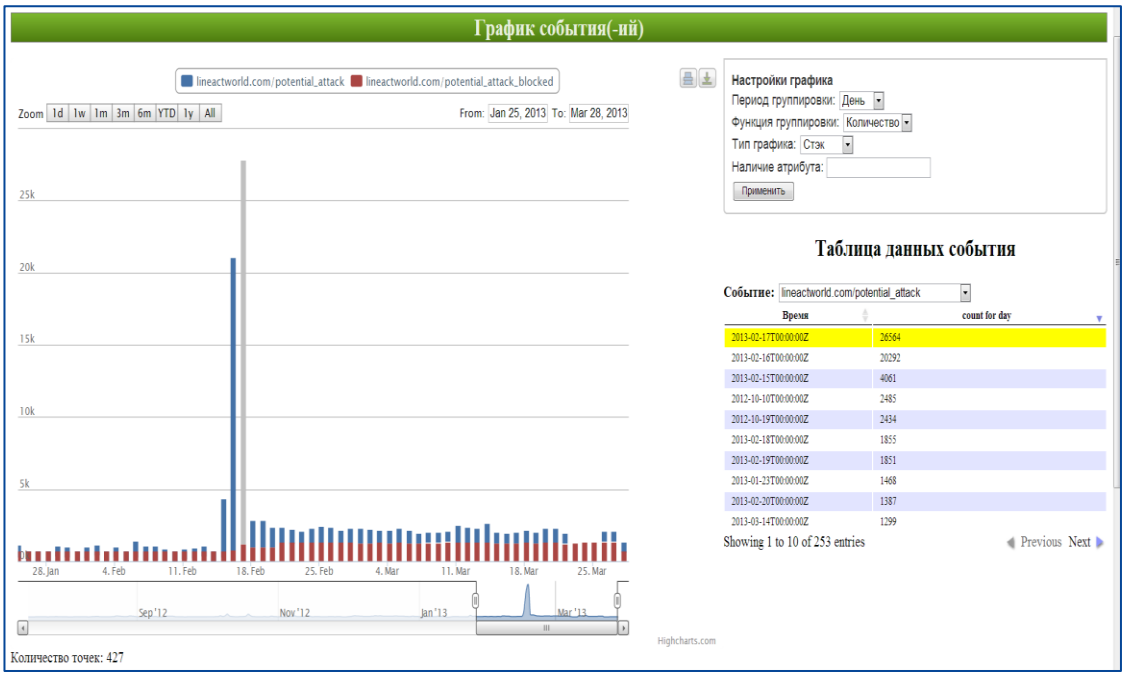


Рис. 3. Область графика. Слева расположен график событий выбранной серии, справа – настройки отображения и группировки данных, а также таблица с текстовым представлением данных событий.

Точки на графике – события выбранной серии. Значение по оси абсцисс – время, когда событие произошло, значение по оси ординат – численное значение, связанное с событием. При помощи окна настроек графика можно выбрать вид отображения событий – в виде точек, соединенных линиями, или в виде столбцов. Также, в настройках присутствует возможность включения группировки событий на предопределенных отрезках времени (год, месяц, день, час). В таком случае для событий на каждом из отрезков вычисляется численное значение по выбранному пользователем принципу: количество точек, сумма/среднее/максимум/минимум значений событий.

Таблица с текстовым представлением данных событий содержит следующие колонки:

- Время, когда событие произошло;
- Численное значение;
- Комментарий к событию (отсутствует в случае группировки событий);

- Список атрибутов и их значений (отсутствует в случае группировки событий).

Записи таблицы могут быть отсортированы по каждой из колонок.

Для удобства анализа таблица и график событий синхронизированы, то есть, если выделить событие (точку) на графике, то отобразится и подсветится соответствующая ему строка таблицы. И наоборот, если выделить событие (строку) в таблице строку таблицы, то соответствующее ему событие на графике подсветится.

Условия и оповещения

В ходе эксплуатации системы было обнаружено, что интерес представляет не только накопление и визуализация, но и автоматическая оценка происходящих событий.

Для этого система научена оценивать вновь приходящие события с заданным именем, проверяя их значения на соответствие заданным условиям. Своевременная сигнализация и визуализация различных ситуаций позволяет оперативно оценивать ситуацию или выявлять сбои в работе программ.

Схема работы механизма условий и оповещений изображена на Рисунке 4.

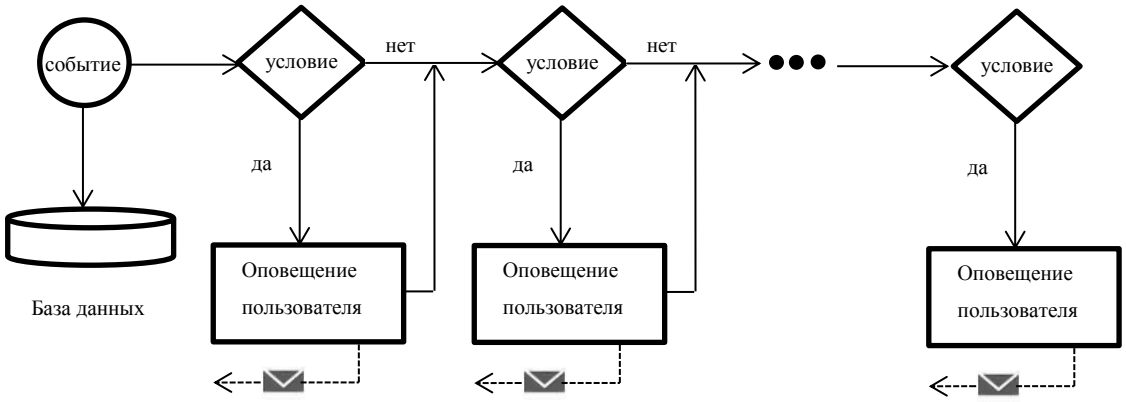


Рис. 4. Вновь поступившее в систему событие помещается в базу данных и проходит проверку на соответствие условиям, заданным пользователем.

Просмотреть существующие условия и создать новое можно при помощи веб-интерфейса системы. Чтобы задать условие, пользователю необходимо воспользоваться специальной формой создания нового условия и определить в ней следующее:

- Имя условия. Служит для отражения характера условия.
- Имя события. Определяет те события, для которых будет осуществляться проверка данного условия.
- Код условия на языке Ruby. Код может обращаться к данным как вновь пришедшего события, так и уже накопленных в базе данных системы, и при этом возвращать значение «истина» или «ложь».
- Адрес электронной почты, на который будет высылаться оповещение в случае, если код условия вернул значение «истина».

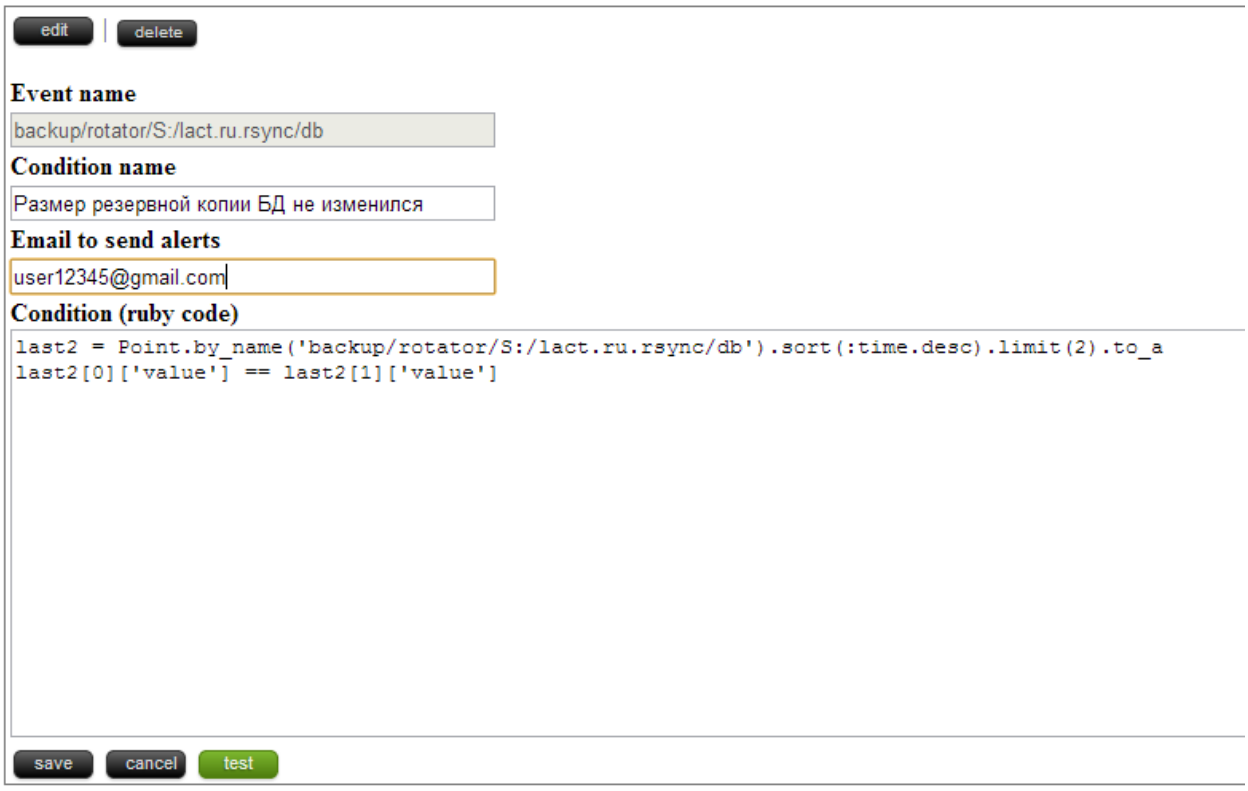


Рис. 5. Пример задания условия.

Если код условия, введенный пользователем, некорректен или возвращает значение, тип которого отличен от булева, система выдаст

соответствующее предупреждение. Система также позволяет выполнить проверку любого корректного условия путем нажатия кнопки «test».

Механизм условий и оповещений работает в экспериментальном режиме. Возможность задания условия в виде исполняемого кода, очевидно, создает уязвимость в системе, но с другой стороны, позволяет описывать принципиально разные условия. На данном этапе проводится анализ создаваемых пользователями условий с целью выделения среди них групп по признаку похожести. В дальнейшем планируется на основании накопленных знаний о таких группах сформировать множество шаблонов условий и заменить текущий способ задания условий на способ задания путем выбора шаблона условия и конкретизации его параметров.

Внутренние события системы

Поскольку система Evented предназначена для анализа работы программ, она с тем же успехом может быть использована для анализа собственной работы.

События, инициируемые системой Evented и помещающиеся в нее саму, назовем «внутренними», а все остальные – «внешними». Имена внутренних событий имеют префикс «Evented/».

Список внутренних событий системы приведен в Таблице 1.

Таблица 1. Внутренние события системы Evented.

Имя события	Когда происходит	Какие данные содержит
Evented/events/client_bad_data	Когда в систему поступает внешнее событие, данные которого не соответствуют правильному формату.	Комментарий к такому внутреннему событию содержит текстовое представление данных «проблемного» внешнего события.
Evented/events/deleted	Когда пользователь удаляет серию событий (все события с заданным именем).	Комментарий к внутреннему событию содержит имя удаленной серии и IP-адрес удалившего ее пользователя.
Evented/events/new	Когда в систему поступает событие при условии, что в системе нет событий с таким же именем.	Комментарий к событию содержит имя нового события.
Evented/events/pushed;	Каждый раз, когда в систему поступает внешнее событие.	Численное значение такого внутреннего события равно количеству минут, прошедших с момента формирования внешнего события (в программе пользователя) до момента попадания в систему Evented. Комментарий содержит имя внешнего события и IP-адрес отправителя.
Evented/internal_error;	Каждый раз при возникновении обработанного исключения в работе системы.	Комментарий содержит стек ошибки.
Evented/alerts/sent	Когда система в связи с выполнением некоторого условия высылает пользователю оповещение.	Атрибуты внутреннего события содержат информацию о выполненном условии (имя условия, имя события, адрес почты).
Evented/graph_viewed	Когда пользователь просматривает график какого-либо события.	Атрибуты внутреннего события содержат имя отображенного события и параметры визуализации.

ДОПОЛНИТЕЛЬНЫЕ СРЕДСТВА ВИЗУАЛИЗАЦИИ СТАТИСТИКИ

В рамках научно-исследовательской практики решалась задача по развитию возможностей средств визуализации, предоставляемых веб-интерфейсом системы Evented. С одной стороны, более функциональный, тонкий и гибкий инструмент визуализации событий серии позволил бы пользователю проводить более глубокий анализ данных и, соответственно, делать качественно новые выводы. С другой стороны, необходимо предоставить пользователю общую информацию о хранящихся в системе данных. Так, некие информативные показатели, вычисленные для данных в целом, а также характеристики серий событий, представленные в удобном для анализа и сравнения виде, могут указать пользователю курс дальнейшего, более детального изучения данных.

В теории визуализации существует направление, изучающее свойство визуального представления информации способствовать генерации у пользователей инсайтов (от англ. insight – проникновение в суть, озарение, внезапная идея) [3]. По наличию именно такого свойства можно судить о качестве визуализации. В то время как понятие «инсайт» не имеет формального определения, основные характеристики инсайта можно перечислить:

- Комплексность. В процесс генерации инсайта вовлечены все или большая часть данных, а не индивидуальные значения.
- Глубина. Инсайт нарастает с течением времени, накапливаясь и становясь глубоким. Инсайт часто порождает новые вопросы и, следовательно, новые инсайты.
- Качественность. Инсайт не может быть точным. Он всегда содержит долю неопределенности и часто субъективен.
- Внезапность. Инсайт часто непредсказуем, случаен и имеет творческую составляющую.

- Релевантность. Инсайт тесно связан с контекстом, связывает данные с существующими знаниями о предметной области и придает им релевантный смысл.

Целью данного исследования стало развитие существующих средств визуализации и внедрение новых таким образом, чтобы не только повысить качество и скорость решения частных, простых задач в области анализа данных, но и обеспечить понимание пользователем общей картины происходящего (общей структуры и распределения данных, тенденций изменения потока данных) путем стимулирования выработки инсайтов.

В результате исследования и с учетом пожеланий пользователей системы Evented, было выработано и решено несколько практических задач.

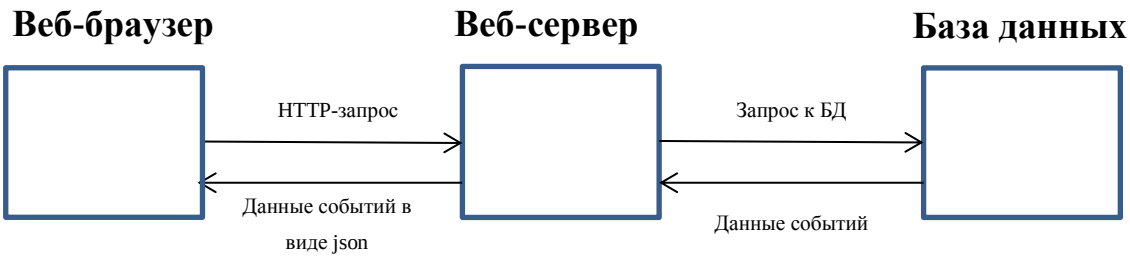
Инструмент фильтрации событий

Зачастую при просмотре графика событий пользователя интересуют не все события выбранной серии, а лишь те, удовлетворяющие заданному критерию. Значит, необходимо иметь возможность применять к событиям различные фильтры. Ниже приведены наиболее востребованные из них:

- Численное значение события меньше, больше или равно заданного;
- Комментарий к событию содержит заданную подстроку;
- Событие имеет заданный атрибут или атрибут с заданным значением.

Таким образом, перед нами встает задача создания инструмента фильтрации событий на графике согласно предъявленным выше критериям.

При нажатии в веб-интерфейсе Evented кнопки «показать график серии», веб-браузер пользователя отправляет HTTP-запрос к серверной части системы. Та, в свою очередь формирует запрос к базе данных для извлечения событий с учетом пользовательских настроек (имя серии событий, параметры группировки). Далее, извлеченные данные отправляются клиенту и используются для построения графика.



Идея решения данной задачи тривиальна и состоит в преобразовании запроса к базе данных таким образом, чтобы при извлечении событий учесть требования, наложенные пользователем на численное значение события, комментарий, список атрибутов и список значений атрибутов.

Задать настройки фильтрации пользователь может в области графика событий в окне настроек (Рис. 1).

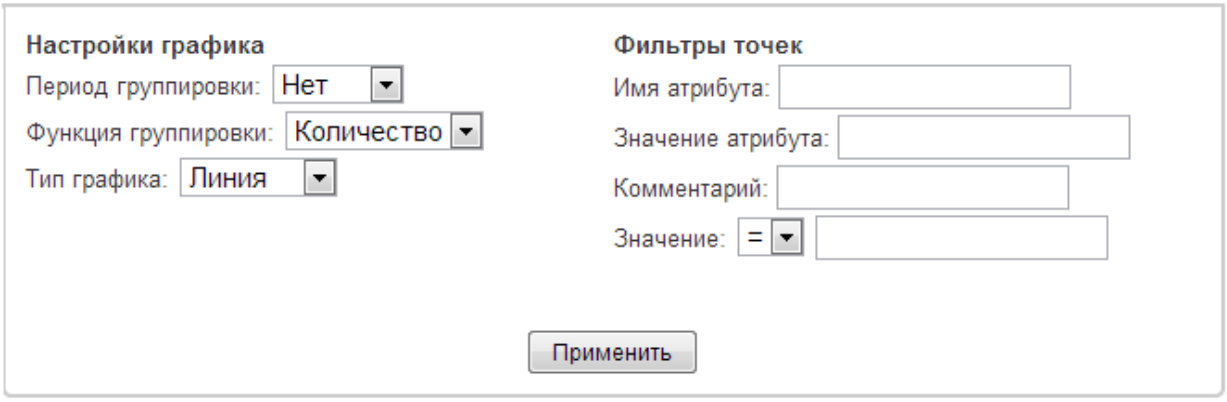


Рис. 1. Обновленное окно настроек области графика событий.

Количество событий. Распределение количества событий между сериями

Область серий событий дополнена новым информационным окном, отражающим:

- Общее количество событий, хранящихся в системе;
- Количество событий, удовлетворяющих фильтру, заданному в настройках списка серий событий;
- Количество событий на заданном в настройках интервале времени (с учетом фильтра);

- Распределение количества событий между сериями из списка серий событий. Реализовано в виде круговой диаграммы, на которой каждой из серий соответствует некоторый сектор.

На рисунке 2 представлен пример информационного окна «Количество событий» при заданных пользователем настройках:

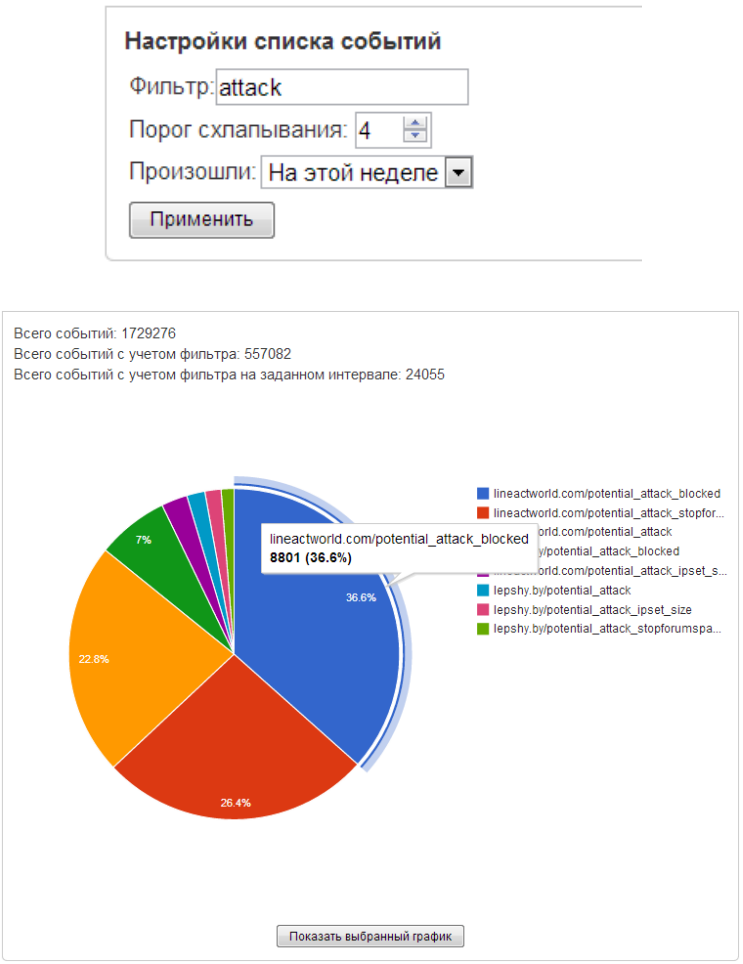


Рис. 2. Окно настроек списка серий событий и информационное окно «Количество событий».

В окне также присутствует кнопка быстрого перехода к просмотру графика серии событий, соответствующей выбранному на диаграмме сектору.

Отклонение скорости потока событий от среднего значения

Для каждой серии событий S и отрезка времени $[t_н, t_к]$ можно вычислить скорость потока событий данной серии на заданном отрезке времени по формуле:

$$V(S, t_н, t_к) = \frac{N(S, t_н, t_к)}{t_к - t_н}$$

где N(S, $t_н$, $t_к$) – количество событий серии S, случившихся на отрезке $[t_н, t_к]$.

Средняя скорость потока событий серии S вычисляется по формуле:

$$V(S)_{средн} = V(S, T_н, T_к)$$

где $T_н$ и $T_к$ – моменты появления самого первого и последнего событий серии S соответственно.

Текущая скорость потока событий серии S – скорость, вычисленная для отрезка времени, выбранного в окне настройки списка серий событий. Отклонение скорости потока событий серии S от среднего значения вычисляется по формуле

$$\text{Отклонение} = \frac{V(S)_{тек} - V(S)_{средн}}{V(S)_{средн}} \times 100\%$$

Каждый раз при изменении настроек списка серий событий для каждой из серий в списке вычисляется отклонение. Полученные пары (имя серии, отклонение) разбиваются на две группы: группа с положительными значениями отклонений и группа с отрицательными значениями. Элементы внутри групп сортируются по убыванию модуля значения отклонения, и для десяти первых элементов каждой из групп строится столбиковая диаграмма, отображаемая в области серий событий (Рис. 3).

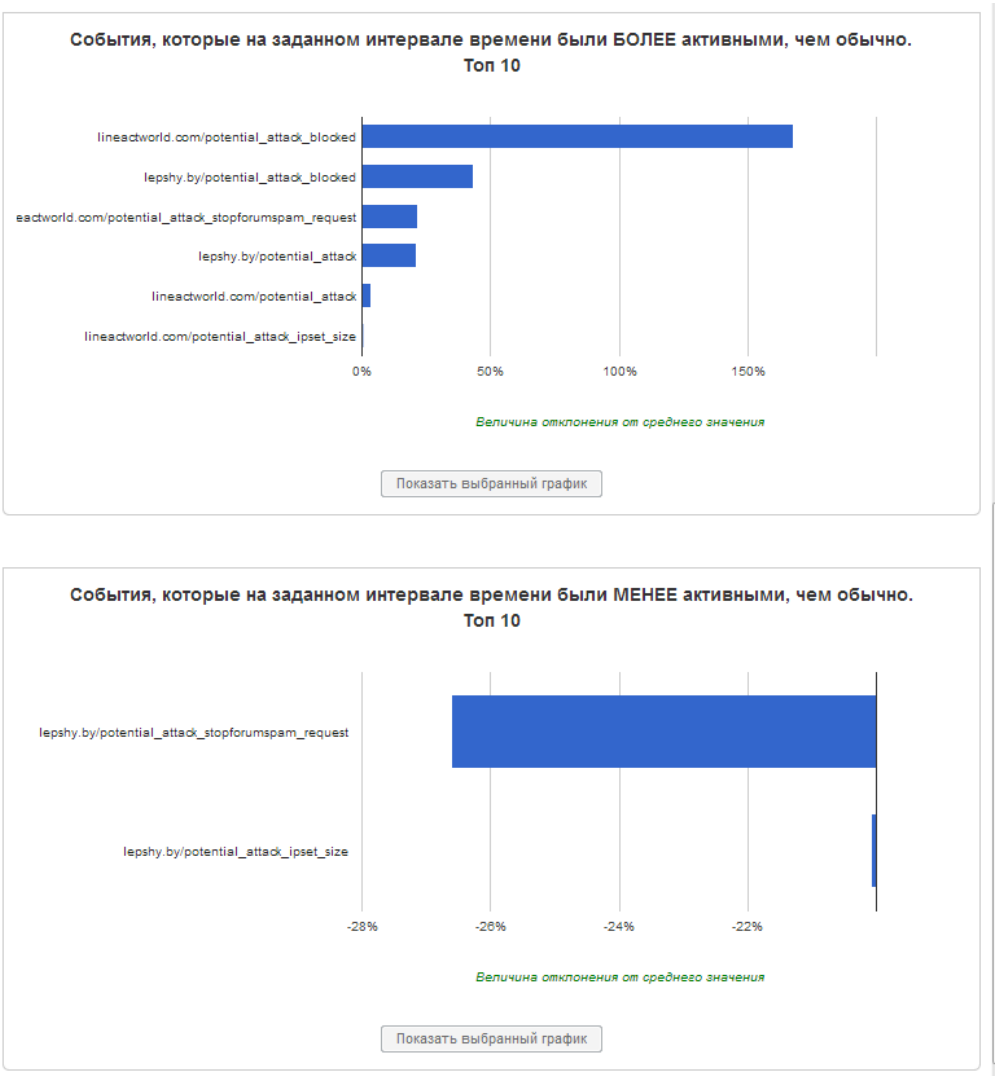


Рис. 3. Столбиковые диаграммы отклонений текущих скоростей потоков событий от средних значений.

Полученные диаграммы отражают, какие события на интересующем пользователя отрезке времени были более/менее активными, чем обычно. Наличие события с большим значением отклонения может послужить сигналом о существовании проблемы в работе пользовательской программы, о появлении нештатной ситуации или о стремительном росте/падении популярности некоторого функционала пользовательского приложения.

ДЕТАЛИ РЕАЛИЗАЦИИ

Средства для разработки

Для реализации и исполнения кода серверной части системы были использованы следующие программные и технические средства:

- Язык программирования Ruby v1.8.7;
- Программный каркас Sinatra v1.3.4, предназначенный для разработки веб-приложений;
- Документо-ориентированная СУБД MongoDB v2.4.3;
- Библиотеки (HTML5/JavaScript) Highstock, Google Charts и DataTables для создания интерактивных графиков, диаграмм и таблиц.

Установка системы на сервере разработчика

Чтобы подготовить сервер для запуска системы необходимо выполнить следующие шаги:

- Установить интерпретатор языка Ruby версии 1.8.7 (http://www.ruby-lang.org/en/).
- Установить RubyGems – менеджер пакетов для языка Ruby (http://rubygems.org/).
- Установить Ruby DevKit (http://rubyinstaller.org/add-ons/devkit/).
- Установить набор необходимых для работы системы библиотек (gem’ов).
- Скачать и распаковать архив с СУБД MongoDB в произвольную директорию (http://www.mongodb.org/downloads # >= v2.0.7).

Исходный код системы Evented и более подробная информация по ее установке доступны по адресу https://bitbucket.org/msogomonyan/evented

Для запуска системы требуется:

1. Запустить 'mongod.exe' с указанием пути для хранения данных MongoDB.
2. Выполнить команду «ruby main.rb» или пакетный файл «run_server.cmd».

Принцип именования событий. Алгоритм преобразования списка имен в дерево

Принцип именования событий, поступающих в систему Evented, целиком и полностью определяется разработчиком программ, которые генерируют эти события. Однако автор данной работы рекомендует использовать следующий способ именования, основанный на опыте работы с системой и обеспечивающий информативность получаемых имен:

1. Имя состоит из нескольких составляющих – слов и названий. Составляющие разделяются символом «/».
2. Первая составляющая является именем проекта (приложения), адресом или доменным именем сервера, с которого поступает событие.
3. Последующие составляющие несут смысл категорий или подкатегорий, к которым относится событие.
4. Последняя составляющая идентифицирует событие внутри иерархии, образованной категориями.

Пример имен событий, образованных данным способом:

- lineactworld.com/delayed_job/finished
- lepshy.by/admin/payment/normal
- lepshy.by/admin/payment/short

По мере того как количество различных имен (серий событий) растет, их список становится неудобочитаемым. Впоследствии был разработан

алгоритм, преобразующий список имен в древовидную структуру, более удобную для чтения.

Часто среди имен событий можно выделить группы, внутри которых имена имеют общий префикс. При этом тривиальные группы (группы из одного элемента) не рассматриваются. Префиксом может служить первая или несколько первых составляющих имени, разделенных символом «/». Например, имена

- **lepshy.by**/admin/payment/normal
- **lepshy.by**/admin/payment/short
- **lepshy.by**/delayed_job/finished

имеют общий префикс «lepshy.by» и образуют группу. Такую группу можно представить в виде дерева

Lepshy.by
- admin/payment/normal
- admin/payment/short
- delayed_job/finished

где корнем является общий префикс, а листьями – имена с «отсеченным» префиксом.

Процесс преобразования не закончен, поскольку среди листьев полученного дерева можно выделить еще одну группу имен с общим префиксом

- **admin/payment**/normal
- **admin/payment**/short

и, преобразовав ее в поддерево

admin/payment
- normal
- short

присоединить к корневому дереву «Lepshy.by»:

Lepshy.by
- **admin/payment**
  - normal
  - short
- delayed_job/finished

Теперь, поскольку среди братьев (листьев с одним и тем же отцом) нельзя выделить группы с общим префиксом, процесс построения дерева считается завершенным.

Описанный выше принцип построения дерева из списка имен событий реализуется в виде рекурсивного алгоритма. Каждый раз на вход алгоритму подается список имен и имя корня, к которому будут присоединены поддеревья, полученные в результате преобразования списка. В качестве начальных параметров подается список всех имен и пустой корень.

Алгоритм

Входные параметры: *список имен*, *корень*.

1. Текущим объявляется последний элемент *списка*. Осуществляется просмотр элементов *списка* с конца, при этом происходит образование групп следующим образом:
   1) Объявляется набор в новую группу.
   2) Текущий элемент добавляется в группу, для которой ведется набор.
   3) Для текущего элемента вычисляется наибольший общий префикс (НОП) с предшествующим ему элементом *списка* (НОП для первого элемента *списка* полагается равным пустой строке).
   4) Если НОП – непустая строка, то текущим объявляется предшествующий элемент и осуществляется переход в пункт 2).
   5) Набор в группу объявляется закрытым.
   6) Если не все элементы *списка* были распределены по группам, текущим объявляется предшествующий элемент и осуществляется переход в пункт 1).
2. Для каждой группы:
   a. Если группа тривиальна, то присоединяем ее единственный элемент к *корню*.

b. Иначе к *корню* присоединяется результат работы Алгоритма, запущенного со следующими параметрами:
    - В качестве *корня* выбирается наименьший непустой из НОП'ов, вычисленных для элементов данной группы;
    - Элементы группы, полученные «отсечением» от них выбранного НОП'а, образуют *список имен*.
3. Возвращаемое значение – *корень* (с присоединенными к нему на шаге 2. поддеревьями).

Характеристики Алгоритма для входных данных длинны n (количество имен в списке):

- Максимальная глубина рекурсии – n-1
- Временная сложность работы алгоритма с учетом рекурсии – $O(n^2)$
    - Сложность начальной сортировки имен – $O(n^2)$
    - Сложность пункта «1» – $O(n)$
    - Сложность пункта «2» (без учета рекурсии) – $O(n)$

ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ

Система Evented используется разработчиками таких программ, как:

- Веб-сервис «Конструктор сайтов LineAct» [4];
- Система научной визуализации «SharpEye» [5];
- Приложение для мобильных устройств «Mita».

Посмотрим, как мониторинг веб-сервиса «Конструктор сайтов LineAct» помогает анализировать его работу, выявлять проблемы и принимать оптимальные решения.

Пример №1. Анализ востребованности функции «Поиск изображения»

Пользователям конструктора сайтов предоставлена функция поиска изображения для иллюстрации материала (Рис. 1).

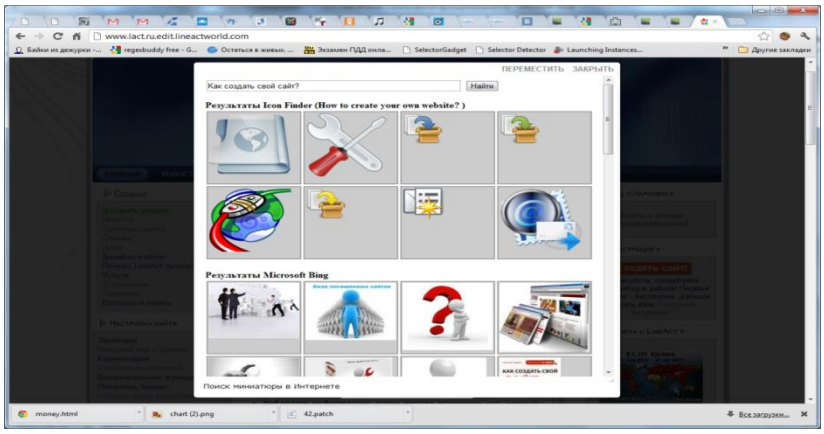


Рис. 1. Окно «Поиск изображения».

Поиск изображения осуществляется в сети интернет с помощью Bing Search API. Интерфейс поиска позволяет разработчику программный путем строить запросы к поисковой системе и получать результаты в таких форматах как XML и JSON.

Ситуация: весной 2012 года компания Microsoft сообщает о том, что планирует сделать Bing Search API платной услугой. Причем стоимость подписки будет зависеть от количества обращений к интерфейсу в месяц.

Задача: выяснить, насколько интенсивно используется функция подбора изображений, и решить, какую подписку оформить.

Решение: в функцию LineAct *engine/find_image* добавлено отслеживание её вызова (Рис.2).

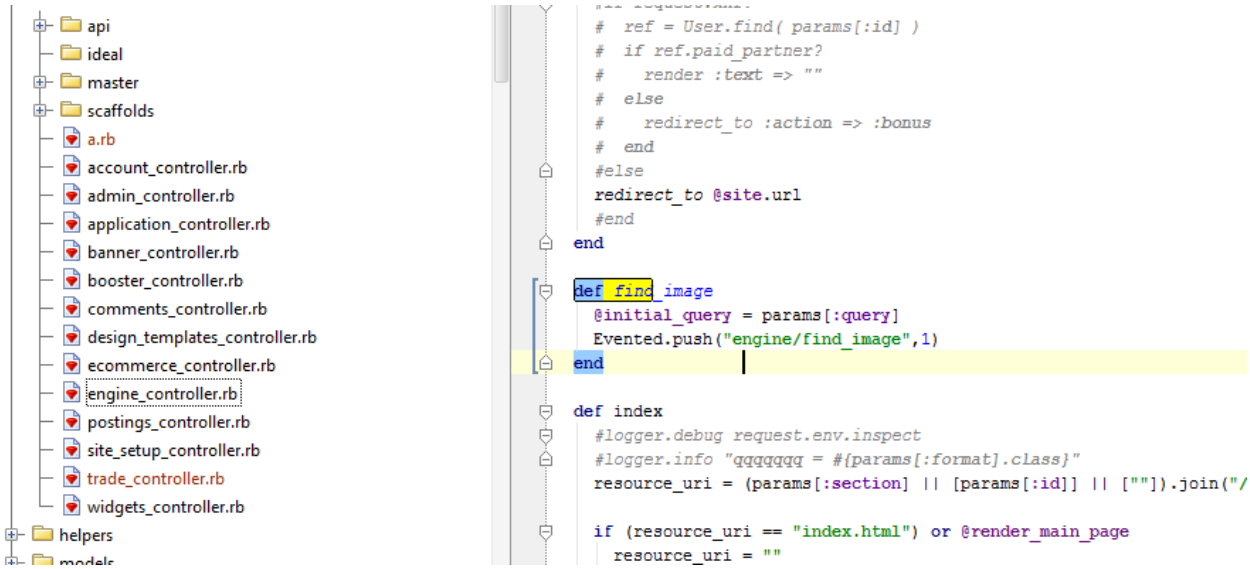


Рис. 2.

Благодаря отслеживанию запуска функции «поиск изображения» при помощи Evented, удалось установить, насколько востребована данная функция и сколько в среднем раз в месяц она используется (Рис. 3). Таким образом, был сделан вывод о том, какой тип подписки требуется оформить.

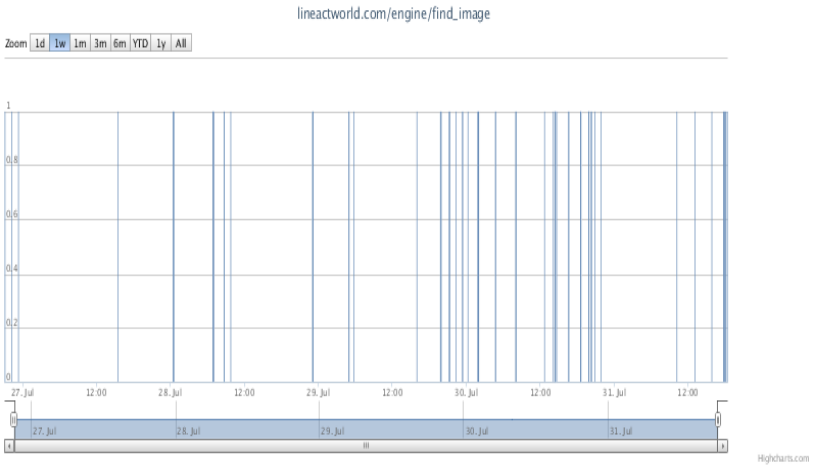


Рис. 3. График события «запуск функции поиска изображения».

Пример №2. Контроль процесса резервного копирования базы данных

Ситуация: в веб-сервисе LineAct на головном сервере установки ежедневно создается резервная копия базы данных. Затем посредством программы rsync, отвечающей за синхронизацию файлов и каталогов между двумя узлами сети, специальный хост, предназначенный для хранения резервной копии базы данных, обновляет локальную резервную копию и помещает ее в архив.

Задача: визуально контролировать процесс, чтобы знать, что он корректно функционирует.

Решение: для контроля над процессом решено отслеживать размер архива ежедневной резервной копии базы данных.

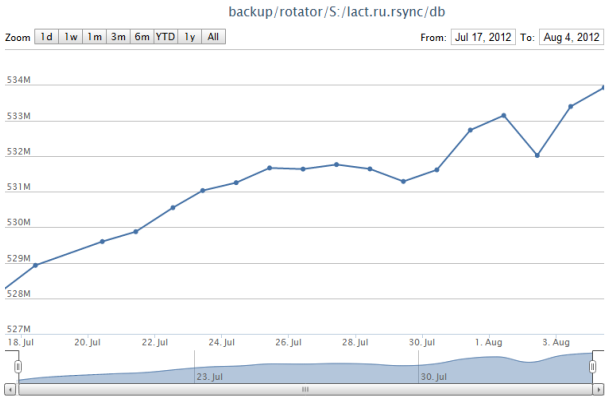


Рис. 4. График события «Размер резервной копии базы данных LineAct».

Поскольку данные сайтов, созданных при помощи конструктора, хранятся в базе данных «LineAct», изменение размера резервной копии отражает тот факт, что пользователи конструктора создают новые сайты, изменяют и дополняют контент существующих. Уменьшение размера резервной копии обычно связано с работой скриптов по удалению «старых» данных.

Сбой создания резервной копии базы данных

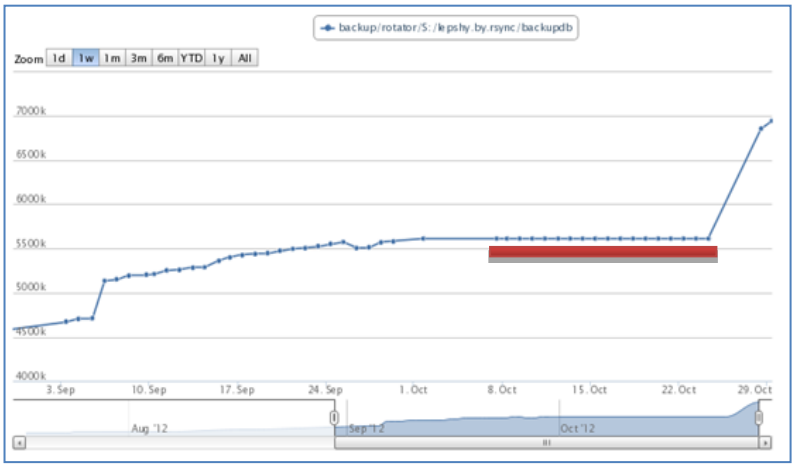


Рис. 5. Сбой создания резервной копии БД.

При помощи системы Evented удалось установить, что размер ежедневной резервной копии базы данных не менялся в течение нескольких дней (Рис.5). Поскольку на протяжении указанного времени веб-сервис корректно функционировал, факт о неизменности размера резервной копии базы данных послужил сигналом о наличии проблемы в процессе резервного копирования. В результате проблема была локализована и устранена.

Впоследствии для автоматического обнаружения подобной ситуации было создано условие оповещения «Размер резервной копии не изменился» (Рис. 6). В коде условия осуществляется проверка на равенство размеров текущей и предыдущей резервных копий, и в случае его выполнения на указанный пользователем адрес электронной почты высылается оповещение.

Рис. 6. Задание условия оповещения «Размер резервной копии не изменился».

Сбой работы rsync

При помощи Evented были обнаружены две точки неестественного скачкообразного роста размера резервной копии (Рис 7). Выяснилось, что работа программы rsync в обоих случаях прерывалась с ошибкой, вследствие чего в результирующее значение размера БД попадала сумма размеров предыдущей резервной копии и «полускачанных» изменений, что и послужило причиной неестественного роста.

Рис 7. Сбой работы программы rsync.

С целью автоматического обнаружения подобной ситуации было создано условие оповещения «Значительное изменение размера резервной копии» (Рис. 8). В коде условия осуществляется проверка на наличие существенной разницы между размерами текущей и предыдущей резервных копий. В случае выполнения условия на указанный пользователем адрес электронной почты высылается оповещение.

Рис. 8. Задание условия оповещения «Значительное изменение размера резервной копии».

Пример №3. Выявление хакерских атак

Ситуация: в веб-конструкторе сайтов LineAct за короткий промежуток времени значительно увеличилось количество пользователей и созданных сайтов. Система Evented наглядно отображает этот факт (Рис. 9). Выяснилось, что неестественный рост вызван действиями злоумышленников по автоматизированному созданию пользователей и сайтов. Создателями конструктора сайтов были разработаны программные средства для выявления и блокировки IP-адресов злоумышленников.

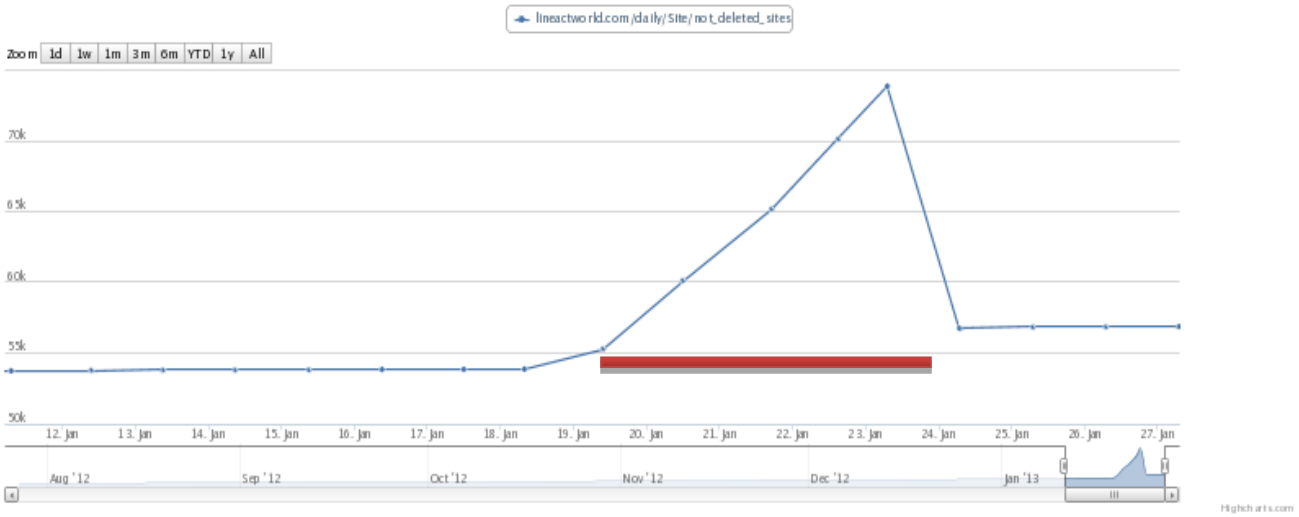


Рис. 9. Количество сайтов и результат вычистки.

Задача: визуально контролировать эффективность борьбы со злоумышленниками.

Решение: было принято решение контролировать число автоматически заблокированных IP-адресов (Рис. 10).

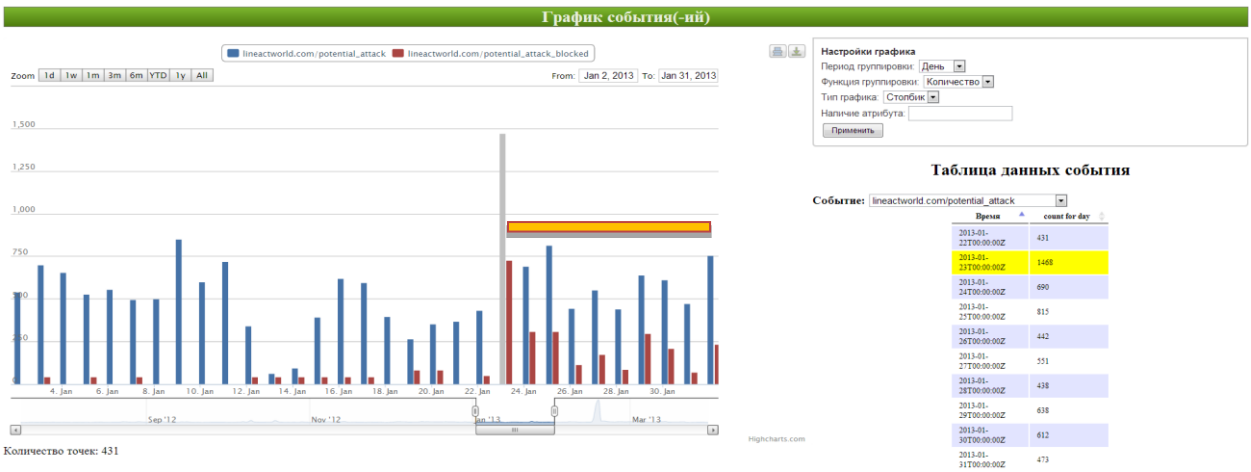


Рис. 10. Количество автоматически заблокированных IP-адресов (красные столбцы) и количество срабатываний датчиков авто-firewall (синие столбцы) в разрезе по дням.

Видно, что в конце января увеличилось количество атак на веб-сервис LineAct. Наряду с этим, прослеживается активность средств защиты от злоумышленников (рост количества заблокированных IP-адресов), спадающая по мере ослабевания атак.

ЗАКЛЮЧЕНИЕ

Таким образом, получено решение задачи сбора и визуализации данных о работе программ: разработаны средства для сбора данных – простые и удобные в использовании клиентские библиотеки, обеспечивающие надежную доставку статистики, а также система, служащая для накопления, хранения и анализа таких данных. При этом анализ может проводиться как пользователем, при помощи визуальных средств, доступных через веб-интерфейс, так и самой системой в автоматическом режиме с последующим оповещением в случае возникновения определенных пользователем ситуаций.

Система не столь мощна, как ее коммерческие аналоги, созданные крупными компаниями разработчиков, однако в отличие от них распространяется свободно и потому доступна каждому, причем не только для использования, но и для расширения ее возможностей и адаптации под собственные цели путем изменения исходного кода.

Летом 2012 года экземпляр системы был запущен в сети Интернет и стал доступен по адресу http://evented.lineact.com:5000/. Своевременная сигнализация и визуализация различных ситуаций не раз помогли ее пользователям, позволяя оперативно оценивать ситуацию или выявлять сбои в работе их программ.

Не прекращается поиск визуальных представлений и средств, позволяющих взглянуть на данные о работе программ с новой стороны. Также активно ведется разработка в направлении обеспечения эффективности работы с большим объемом информации: уже сейчас в системе хранится около двух миллионов событий, а их поток составляет двести тысяч событий в месяц.

СПИСОК ПУБЛИКАЦИЙ

1. Васёв П.А., Согомонян М.С., Возможности системы мониторинга работы программ Evented // СОВРЕМЕННЫЕ ПРОБЛЕМЫ МАТЕМАТИКИ: тезисы Международной (44-й Всероссийской) молодежной школы-конференции. Екатеринбург: Институт математики и механики УрО РАН, 2013, с. 218-220.

2. Васёв П.А., Согомонян М.С., Веб-система визуализации, анализа и мониторинга работы программ // Международная научная конференция Параллельные вычислительные технологии 2013. Национальный исследовательский Южно-Уральский государственный университет, 1-5 апреля 2013 г., г. Челябинск. Сборник Трудов. Том 2. С. 586.

СПИСОК ЛИТЕРАТУРЫ

1. Авербух В.Л., Байдалин А.Ю., Разработка средств визуализации программного обеспечения параллельных вычислений. Визуальное программирование и визуальная отладка параллельных программ. // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов, 2003, вып. 4., с. 68-80.
2. Martin Beck, Jonas Trumper, Jurgen Dollner. A Visual Analysis and Design Tool for Planning Software Reengineerings // Proceedings of the 6th IEEE International Workshop on Visualizing Software for Understanding and Analysis, VISSOFT 2011, Williamsburg, VA, USA, September 29-30, 2011, pp.~54-61.
3. Chris North, “Visualization Viewpoints. Toward Measuring Visualization Insight”, Virginia Polytechnic Institute and State University, 2006
4. Vasev Pavel, HUMAN-AWARE APPROACHES FOR WEBSITE CONTENT MANAGEMENT SYSTEMS // Proceedings of IADIS International Conference e-Society 2009, Barcelona, Spain, pp. 181-183.
5. П.А. Васёв, С.С. Кумков, Е.Ю. Шмаков, Функциональные возможности среды-конструктора систем научной визуализации SharpEye // Параллельные вычислительные технологии (ПаВТ'2013): труды международной научной конференции (1–5 апреля 2013 г., г. Челябинск). ISBN 978-5-696-04396-8. Челябинск: Издательский центр ЮУрГУ, 2013. С. 587.

ПРИЛОЖЕНИЕ 1

Формат вызова функции “push”, реализованной в библиотеке для языка Ruby, описан в Листинге 1.

Evented.push имя_события, значение_события, [хэш_атрибутов]

Листинг 1. Формат вызова функции “push”.

Таблица 1. Параметры вызова функции отправки события «push» (реализация для Ruby)

Имя параметра	Обязательный	Комментарий	Тип
имя_события	Да	Отражает имя события.	Строка
значение_события	Да	Отражает численное значение, связанное с событием. Будучи представленным в виде строки, является комментарием к событию.	Число или строка
хэш_атрибутов	Нет	Служит для указания параметров отправки или для задания атрибутов события.	Hash. Набор пар («имя», «значение»)

Таблица 2. Параметры отправки события (ключевые слова).

Имя параметра	Комментарий	Тип	Значение по умолчанию
time	Время, когда произошло событие.	Объект Time	Время сервера в момент формирования события.
server	Адрес сервера, куда отправляется событие.	Строка	"http://evented.lineact.com:5000"
push_mode	Режим отправки события	[‘normal’, ‘threaded’, ‘logged’, ‘local_log’, ‘nop’, ‘normal_retry’]	‘logged’

Таблица 3. Режимы отправки события

Режим отправки события	Описание
normal	Синхронный режим отправки события.
threaded	Асинхронный режим отправки события (отправка в параллельном потоке).
logged	Режим отложенной записи (запись в локальный файл, запуск демона, отправка в асинхронном режиме).
local_log	Запись в локальный файл.
nop	Отсутствие действия.
normal_retry	Десять попыток отправки в синхронном режиме