циональности. При этом вопросам безопасности и качества программного кода уделяется недостаточно внимания. В результате подавляющее большинство веб-приложений содержит уязвимости различной степени критичности.

Простота протокола HTTP позволяет разрабатывать эффективные методы автоматического анализа веб-приложений и выявления в них уязвимостей. Это значительно упрощает работу нарушителя, позволяя ему обнаружить большое число уязвимых веб-сайтов, чтобы затем провести атаку на наиболее интересные из них.

Кроме того, уязвимости некоторых типов допускают не только автоматическое выявление, но и автоматическую эксплуатацию. Именно таким образом производится массовое внедрение в веб-ресурсы вредоносного кода, который затем используется для создания бот-сетей из рабочих станций обычных пользователей сети Интернет. Возможность использования веб-приложений в качестве платформы для атаки на рабочие места пользователей сама по себе делает эти приложения привлекательной целью для нарушителя.

Таким образом, при подготовке атаки на информационную инфраструктуру компании нарушители в первую очередь исследуют ее веб-приложения. Недооценка риска, который могут представлять уязвимости в веб-приложениях, доступные из сети Интернет, возможно, является основной причиной низкого уровня защищенности большинства из них.

OWASP (некоммерческой организации Open Web Application Security Project) после своего исследования представила список десяти наиболее опасных, но, в то же время, распространенных уязвимостей в программном обеспечении для интернета и веб-сервисах [1]. По мнению OWASP, на эти уязвимости стоит обратить самое пристальное внимание как государственным, так и коммерческим организациям, желающим обезопасить себя и своих клиентов от хакеров. Все указанные уязвимости достаточно широко распространены, а использовать их под силу даже малоквалифицированным хакерам, поскольку соответствующие средства взлома легко найти в сети Интернет.

1) Injection (всякого рода инъекции, в т.ч. SQL, LDAP и т.д.).

2) Cross Site Scripting (не потерявший актуальности XSS).

3) Broken Authentication and Session Management (ошибки в архитектуре аутентификации и управления сессиями).

4) Insecure Direct Object References (незащищенные ресурсы и объекты).

5) Cross Site Request Forgery (CSRF).

6) Security Misconfiguration (небезопасная конфигурация окружения, различных фреймворков, платформы).

7) Failure to Restrict URL Access (несанкционированный доступ к функционалу, требующему особых привилегий – например, обход проверки с помощью двойного «//» в URL для получения доступа к управлению блогом в WordPress).

8) Unvalidated Redirects and Forwards (открытые редиректы, которые ведут к фишингу, HTTP Response Splitting и XSS).

9) Insecure Cryptographic Storage (небезопасное хранение важных данных).

10) Insufficient Transport Layer Protection (недостаточная защита данных при их передаче на транспортном уровне, например по HTTP вместо HTTPS).

Опасность некоторых из указанных уязвимостей представлена ниже:

1) SQL-инъекции – встраивание вредоносного кода в запросы к базе данных – наиболее опасный вид атак [2]. С использованием SQL-инъекций злоумышленник может не только получить закрытую информацию из базы данных, но и, при определенных условиях, внести туда изменения. Уязвимость по отношению к SQL-инъекциям возникает из-за того, что пользовательская информация попадает в запрос к базе данных без должной обработке: чтобы скрипт не был уязвим, требуется убедиться, что все пользовательские данные попадают во все запросы к базе данных в экранированном виде.

2) PHP-инъекция (англ. PHP injection) – один из способов взлома веб-сайтов, работающих на PHP, заключающийся в выполнении постороннего кода на серверной стороне [3]. Потенциально опасными функциями являются: * eval(), * preg_replace() (с модификатором «e»), * require_once(), * include_once(), * include(), * require(), * create_function(). PHP-инъекция становится возможной, если входные параметры принимаются и используются без проверки.

3) Cross-Site Scripting – это вид уязвимости программного обеспечения (Web-приложений), при которой, на генерированной сервером странице, выполняются вредоносные скрипты, с целью атаки клиента. Сейчас XSS составляют около 15 % всех обнаруженных уязвимостей. Долгое время программисты не уделяли им должного внимания, считая их неопасными. Однако это мнение ошибочно: на странице или в HTTP-Cookie могут быть весьма уязвимые данные (например, идентификатор сессии администратора). На популярном сайте скрипт может устроить DoS-атаку.

4) CSRF (англ. Cross Site Request Forgery – «Подделка межсайтовых запросов», также известен как XSRF) – вид атак на посетителей веб-сайтов, использующий недостатки протокола HTTP. Если жертва заходит на сайт, созданный злоумышленником, от её лица тайно отправляется запрос на другой сервер (например, на сервер платёжной системы), осуществляющий некую вредоносную операцию (например, перевод денег на счёт злоумышленника).

Таким образом, уязвимости в Web-приложениях по-прежнему остаются одним из наиболее распространенных недостатков обеспечения защиты информации. Проблема защищенности Web-приложений усугубляется еще и тем, что при разработке Web-приложений, зачастую не учитываются вопросы, связанные с защищенностью этих систем от внутренних и внешних угроз, либо не достаточно внимания уделяется данному процессу. Это в свою очередь порождает ситуацию, в которой проблемы ИБ попадают в поле зрения владельца системы уже после завершения проекта. А устранить уязвимости в уже созданном Web-приложении является более расходной статьей бюджета, чем при его разработке и внедрении.

Недооценка серьезности риска реализации угроз ИБ с использованием Web-приложений, доступных со стороны сети Интернет, возможно, является основным фактором текущего низкого состояния защищенности большинства из них.

**Список литературы**

1. Десять самых опасных уязвимостей в Web приложениях // http://www.securitylab.ru/news/212802.php.

2. SQL injection для начинающих. Часть1// http://habrahabr.ru/post/148151.

3. Основы php-инъекций для новичков // https://www.xaker.name/forvb/archive/index.php/t-13504.html

## РАЗРАБОТКА И ПРИМЕНЕНИЕ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ С ВИРТУАЛИЗАЦИЕЙ

Валова О.О., Мартышкин А.И.

*Пензенский государственный технологический университет, Пенза, Россия, e-mail: los@pgta.ru*

Существующие в настоящее время математические модели вычислительных систем (ВС) не пригодны для исследования ВС с виртуализацией, из-за отсутствия

возможности комплексно рассматривать не только потоки информации внутри модели, но и учитывать при этом изменения конфигурации самой модели в зависимости от случайных факторов – программно-аппаратных сбоев, так и заданных схем реконфигурирования системы, что характерно для ВС с виртуализацией.

К современным ВС и системам хранения предъявляются высокие требования:

- адаптируемость к быстро меняющимся задачам;
- гарантия заданного уровня производительности приложений;
- обеспечение требуемого уровня масштабируемости;
- простота эксплуатации и обслуживания;
- стоимость владения должна быть минимизирована.

Наиболее эффективным путем удовлетворения перечисленных требований является использование ВС с виртуализацией. Создание, верификация и анализ возможностей применения математических моделей ВС с виртуализацией ресурсов в образовательном процессе в качестве доступного инструмента для исследования ВС с виртуализацией, а также применения их реализаций – виртуальных серверов на практике, обуславливает актуальность и практическую ценность проводимого исследования.

Анализ публикаций показывает, что в мировой и отечественной практике для решения задач проектирования зачастую используются аналитические методы, требующие для своей реализации меньше вычислительных ресурсов и позволяющие решать как задачи анализа, так и задачи оптимизации параметров [3]. Причем получаемые средние значения характеристик модели, обычно не более чем на 10-15% отличаются от реальных, что вполне приемлемо при инженерных исследованиях.

В представленной работе для расчетов используется метод замкнутых стохастических систем массового обслуживания (СМО), математический аппарат которого уже разработан, но рассчитан на моделях с фиксированными параметрами [2]. В моделях ВС с виртуализацией необходимо учитывать возможность распределения процессорной мощности физического процессора, начиная от 10% с шагом в 1% прямо во время работы [4], см. рис. 1. Поэтому метод был адаптирован для задания и изменения числа каналов обслуживания в СМО в виде доли мощности канала обработки. В том случае, если заданное число каналов в СМО равно 1 – при расчете такая СМО считается одноканальной, иначе многоканальной.

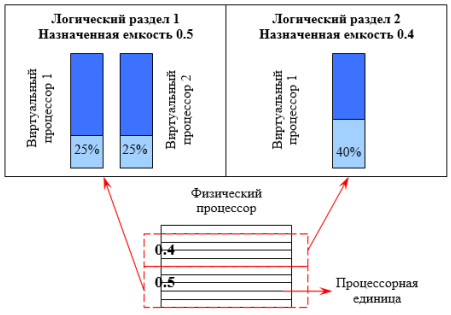



*Рис. 1. Разделение процессорной мощности*

Из-за модификации метода, потребовалось вычислять выражения со степенью с дробным показателем и дробным факториалом, для этого ряд формул был заменен на вычисления приближенных значений. Так, например, вычисление факториала заменено формулой Стирлинга [6], что позволило применить существующий метод расчета для моделей ВС с виртуализацией.

В рамках данного исследования в моделях предполагается наличие пула свободных ресурсов или пула виртуальных серверов – доноров, поскольку моделируется один виртуальный сервер в каждом случае, а не хост-платформа целиком.

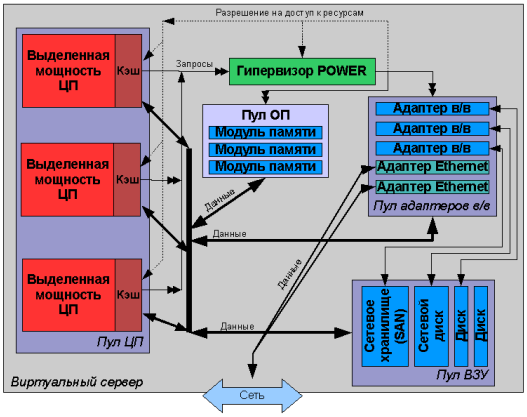



*Рис. 2. Структурная схема виртуального сервера*

Для реализации свойств адаптивности в моделях были использованы триггеры, которые отслеживают и корректируют мощность канала обработки в отдельно взятых СМО в зависимости от заданных условий [1], которые задаются непосредственно перед расчетом модели, путем назначения диапазона изменений и события срабатывания. В реальных ВС тот же самый функционал обеспечивается заданием политики управления ресурсами для менеджера управления ресурсами (PLM), см. рис. 3.

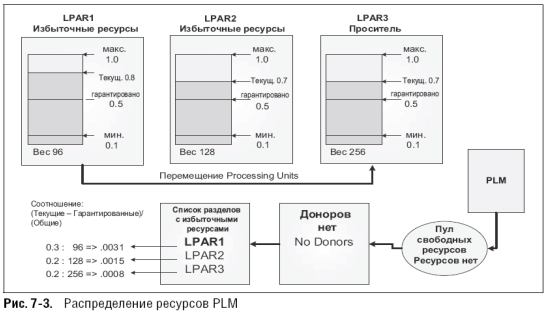



**Рис. 7-3.** Распределение ресурсов PLM

*Рис. 3. Распределение ресурсов PLM*

Также в модели задается вероятность сбоя в виде триггера псевдослучайных сбоев, который создает события времени выполнения модели и позволяет моделировать отказ тех или иных устройств в модели ВС. Эти события проявляются, как резкое сокращение мощности канала обработки в какой-либо СМО до минимального значения равного 0,1. Совсем исключить СМО, в которой произошел сбой, из модели не представляется возможным, поскольку это влечет за собой изменение размера матрицы передач, что в свою очередь исказит всю собранную до момента сбоя статистическую информацию. Для абстрагирования от привязки и измерения количества тактов процессоров для выполнения той или иной задачи, в моделях используются условные единицы времени обслуживания, задаваемые при расчетах, с учетом экспериментально полученных данных о времени обработки заявки с заданной трудоемкостью.

Отличительной особенностью разработанных моделей является возможность задания ресурсов в процентном соотношении. Адаптивные модели выгодны еще и тем, что в процессе выработки решения об изменении конфигурации используется информация, поступающая как от самой модели, так и от исследователя, в виде схем принятия решения [5]. В ходе исследования были получены простые и легко изменяемые модели для анализа.

Число заявок в модели равное 8 и 16 было выбрано экспериментальным путем, исходя из количества СМО в моделях, а также времени расчета модели.

Пример зависимости коэффициента загрузки от числа заявок в модели показан на рис. 4.

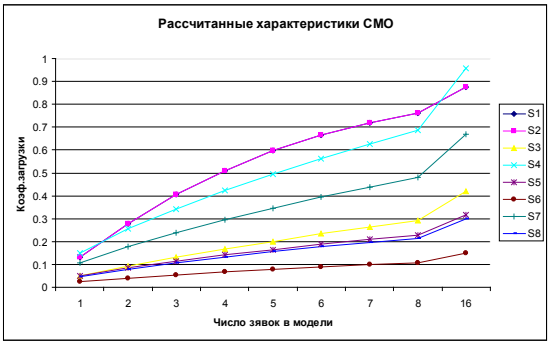



*Рис. 4. Динамика коэффициента загрузки от роста числа заявок*

Оптимизировать модели можно по следующим параметрам:

1) коэффициент загрузки, который показывает эффективность использования ресурса;

2) среднее время пребывания заявки в сети, поскольку, справедливо считать, что чем производительнее система, тем меньше среднее время пребывания заявки в сети или тем меньше время реакции ВС на запрос.

В ходе экспериментов с моделями было выявлено, что для оптимизации – минимизации среднего времени пребывания заявки в сети, необходимо временно увеличить выделенную процессорную мощность до 3,36. При этом коэффициент загрузки процессоров снижается, поскольку заявки начинают обрабатываться параллельно. Соответственно после обработки ресурсы могут быть возвращены обратно в пул свободных ресурсов. С помощью варьирования параметров ЦП, ОП и других компонентов модели, были получены следующие графики динамики коэффициента загрузки как отдельно по СМО, так средней по модели, см. рис. 5 а и 5 б.

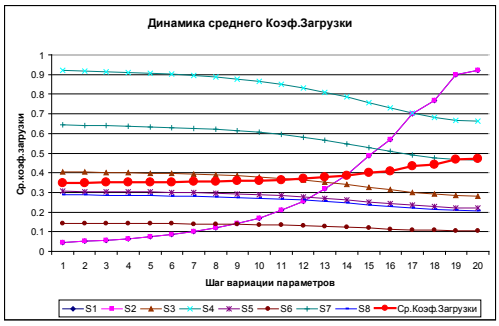



*Рис. 5 а*

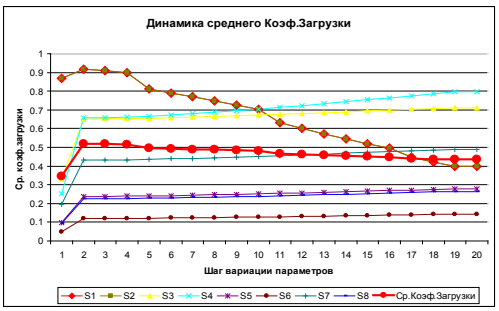



*Рис. 5 б*

На графиках видно влияние доступности дополнительных ресурсов ЦП: при изначально высокой доступности ресурсов снижаются временные затраты на обработку заявок, но при этом также уменьшается эффективность использования этих ресурсов. Таким образом, результаты исследования свидетельствуют о том, что оптимизация «на лету» при наличии пула свободных виртуальных ресурсов, позволяет наиболее эффективно решать задачу, возложенную на ВС, так и наиболее эффективно использовать аппаратные ресурсы, при условии их аренды в ситуациях с пиковой загрузкой, тем самым позволяет снизить общие затраты на аппаратные ресурсы.

Верификация моделей является важным этапом и была выполнена экспериментально, путем создания виртуального сервера на хост-платформе IBM System P и измерением его динамических характеристик. Для генерации нагрузки на сервере, другими словами имитации выполнения заявок по аналогии с моделями, использовалась утилита Stress. Для сбора статистики о работе виртуального сервера использовалась утилита NMON.

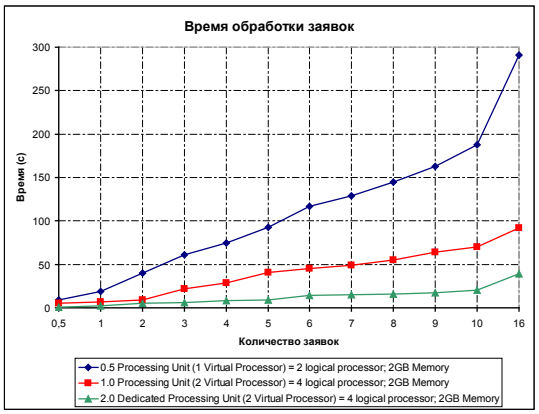



*Рис. 6. Время обработки заявок*

Максимальный результат по скорости обработки заявок достигается, когда число виртуальных процессоров равняется числу потоков в программе или заявке. Средняя скорость обработки заявок соответственно по трем конфигурациям равняется: 0,053, 0,141, 0,483 (заявок/сек). Сравнение результатов показывает схожее поведение модели и реальной системы, как по характеристикам загрузки, так и по использованию ресурсов, а также наглядно продемонстрировала эффект адаптивности к нагрузке, что в свою очередь свидетельствует о корректности использования разработанных моделей для изучения и моделирования ВС с виртуализацией. При этом есть и некоторые различия, которые объясняются различием в распределении входного потока заявок в моделях или пакета задач в реальной системе.

Результатами проведенных исследований являются:

1) Способ адаптации математического аппарата теории массового обслуживания для расчета моделей ВС с виртуализацией, который позволяет использовать проверенный временем математический аппарат с небольшой модификацией для совершенно другого типа ВС.

2) Тестовые модели ВС для различного вида нагрузок, которые были верифицированы экспериментальным путем, что позволило наглядно показать одинаковую динамику модели и виртуального сервера и доказать достоверность моделей.

3) Проведен анализ возможностей применения разработанных моделей и их реализаций – виртуаль-

ных серверов, что освещает важный аспект применения полученных результатов на практике и указывает возможные пути дальнейшего развития по теме моделирования в области виртуализации, в связи с выявленными недостатками технологии.

С практической точки зрения, реализацией моделей являются виртуальные сервера, которые во многих случаях гораздо удобнее использовать, нежели традиционные полностью аппаратные решения.

**Список литературы**

1. Клейнрок Л. Вычислительные системы с очередями. М.: Мир, 1979. 600 с.
2. Алиев Т.И. Основы моделирования дискретных систем. Учебное пособие. СПб: СПбГУ ИТМО, 2009. 363 с.
3. Моделирование информационных систем. / Под ред. О.И. Шелухина. Учебное пособие. М.: Радиотехника, 2005. 368 с.
4. Анника Бланк, Пол Кифер, Карлос Сальяве, мл., Герардо Валенсия, Джез Вейн, Армин М. Варда, «Технология Advanced POWER Virtualization в IBM System p5», Перевод А. Казаков, И. Легостаев, Д. Миронов, Москва, 2007.
5. Таха, Хэмди, А. Введение в исследование операций. 7-е издание: Пер. с англ. М.: Изд. дом «Вильямс», 2005. 912 с.
6. Свидетельство о государственной регистрации программы для ЭВМ №2008612745.

## ПРОБЛЕМЫ ИСПОЛЬЗОВАНИЯ ОБЛАЧНЫХ ТЕХНОЛОГИЙ ПРИ ИЗУЧЕНИИ СУБД MS SQL SERVER

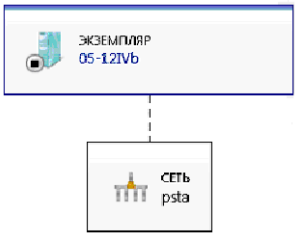
Жулев С.А., Ведюшкина А.Е., Артюшина Е.А.

*Пензенский государственный технологический университет Пенза, Россия, e-mail: los@pgta.ru*

В настоящее время широкое распространение получили облачные технологии. Облачное хранилище данных (cloud storage) – модель онлайн-хранилища, в котором данные хранятся на многочисленных распределённых в сети серверах, предоставляемых в пользование клиентам, в основном, третьей стороной или сервис-провайдером. Частное облако (private cloud) реализует модель развертывания облачных вычислений на имеющихся у конкретной организации вычислительных ресурсах и ресурсах хранения [1].

Сегодня облачные технологии применяются в различных сферах человеческой деятельности, в том числе – в образовании. При использовании частного облака в образовательных целях компьютерные ресурсы и мощности принадлежат локальной вычислительной сети (ЛВС) университета. В таком случае конечными потребителями сервиса являются студенческая группа и преподаватель учебной дисциплины.

Виртуальная машина (ВМ) – это программная и/или аппаратная система, эмулирующая аппаратное обеспечение некоторой целевой (target) платформы и исполняющая программы для target-платформы на host-платформе (host – платформа-хозяин). Для изучения клиент-серверной СУБД MS SQL Server в рамках дисциплины «Базы данных» была предложена следующая архитектура системной ВМ (рисунок):




*Архитектура виртуальной машины SQL Server*

Виртуализация MS SQL Server длительное время считалась невозможной, однако сейчас ясно, что она имеет некоторые преимущества перед развертыванием этой СУБД на физическом сервере. В частности, консолидация нескольких серверов SQL в виде виртуальных машин позволяет оптимально использовать ресурсы ЛВС университета.

Аппаратные и программные ресурсы ВМ, представленные в табл. 1 и 2, находятся под управлением диспетчера виртуальных машин MS System Center App Controller 2012 [2].

**Таблица 1**

Аппаратные характеристики виртуальной машины SQL Server

| №пп | Характеристика | Значение |
|---|---|---|
| 1 | Процессоры | 1 |
| 2 | Память | 512 МБ |
| 3 | Динамическая память | Максимум 1,00 ГБ |

**Таблица 2**

Программное обеспечение виртуальной машины SQL Server

| №пп | Название ПО | Вид ПО |
|---|---|---|
| 1 | MS Windows XP Professional | Операционная система |
| 2 | MS SQL Server 2008(R2) Express Edition | Система управления БД |
| 3 | MS SQL Server Management Studio Express | Среда для администрирования БД |
| 4 | Lazarus 1.2.4 for Windows 32/64 bit | Среда визуального программирования |

При проектировании серверной части приложения для работы с БД возникла следующая проблема зависимости от сервис-провайдера: конечные потребители сервиса (студенты) не могут переносить в частное облако файлы базы данных (*.mdf, *.ldf) со своих домашних компьютеров, поскольку у них отсутствуют права администраторов системы.

При разработке клиентского приложения с помощью кроссплатформенной среды Lazarus потребовалось установить дополнительную библиотеку dblib.dll в системную папку WINDOWS, а также решить проблему удаленного доступа к данным сервера.

Для настройки удаленных подключений необходимо: 1) разрешить удаленные подключения на экземпляре сервера SQL; 2) запустить службу SQL Browser с помощью диспетчера конфигурации SQL Server; 3) создать исключения в брандмауэре Windows для сервера SQL и обозревателя SQL Browser.

Таким образом, для решения указанных проблем необходимо либо предоставить всем конечным пользователям ВМ права системных администраторов, либо найти более эффективный вариант построения облачной инфраструктуры.

**Список литературы**

1. Лоридос П. Вознесение: приложения для облаков: [Электронный ресурс]. 1992-2014. URL: http://www.osp.ru/os/2010/06/13003733/
2. Диздаревич Д. Что такое App Controller 2012: [Электронный ресурс]. 1992-2014. http://www.osp.ru/win2000/2012/11/13033364/

## СРАВНИТЕЛЬНЫЙ АНАЛИЗ АЛГОРИТМОВ ОБРАБОТКИ СИГНАЛА РАДИОЛУЧЕВЫХ ТЕХНИЧЕСКИХ СРЕДСТВ ОХРАНЫ ДЛЯ ОБНАРУЖЕНИЯ ОБЪЕКТОВ НИЗКОЙ СКОРОСТИ ПЕРЕМЕЩЕНИЯ

Надобный А.М., Литвинская О.С.

*Пензенский государственный технологический университет, Пенза, Россия, e-mail: los@pgta.ru*

Радиолучевые технические средства охраны (РЛ ТСО) активно используются для охраны периметров. Низкая стоимость и высокая помехоустойчивость делают их наиболее эффективными для соблюдения целостности границ открытых территорий.