Polovko Ivan Yur'evich
Taganrog Institute of Technology – Federal State-Owned Autonomy Educational Establishment of Higher Vocational Education "Southern Federal University".
E-mail: ivan.polovko@gmail.com.
2, Chexova street, Taganrog, 347928, Russia.
Phone: +78634371905.

Abramov Evgeny
E-mail: abramoves@gmail.com.

УДК 004.732.057

Н.Е. Горло, О.Ю. Пескова

ПОДХОДЫ К ПОСТРОЕНИЮ ЗАЩИЩЕННОЙ СИСТЕМЫ УПРАВЛЕНИЯ КОНТЕНТОМ

Рассмотрены основные подходы к построению защищенной системы управления контентом с упором на разработку сайтов для поддержки электронных библиотек. Представлена разработанная система управления контентом электронных публикаций, которая обеспечивает функции регистрации, аутентификации, разграничения доступа, публикации и редактирования материалов, защищенное хранение пользовательских данных.

Электронные библиотеки; CMS; аутентификация и разграничение доступа.

N.E. Gorlo, O.Yu. Peskova

APPROACHES TO THE CONSTRUCTION OF PROTECTED CONTENT MANAGEMENT SYSTEM

The paper discusses the main approaches to the construction of secure content management systems, with emphasis on the development site to support digital libraries. The developed content management system of electronic publications, which provides the functions of registration, authentication, access control, publishing and editing of materials protected storage of user data.

Digital libraries; CMS; authentication and access control.

Введение. В наше время публикации все чаще появляются не на бумажных носителях, а в глобальной сети Интернет. Статьи располагаются на специализированных порталах, на сайтах учебных заведений, персональных страницах авторов. Достоинство электронной публикации прежде всего в том, что она дает возможность автору постоянно развивать, дополнять и править свой материал – в соответствии с собственным развитием и с критическими комментариями и рецензиями на выложенный в открытый доступ материал. Кроме того, электронная публикация дает возможность использовать элементы мультимедиа, перекрестные ссылки и другие элементы, позволяющие автору максимально полно и доступно представить свои идеи. Доступ к выложенным в сеть материалам открыт миллионам читателей благодаря поисковым сервисам. Поэтому не удивительно, что в сети постоянно появляются новые специализированные ресурсы, предназначенные для размещения и рецензирования научных публикаций (например, сайт научных публикаций сотрудников Российской академии наук www.ras.ru, журнал научных публикаций аспирантов и докторантов www.jurnal.org), и для каждого нового участника «научной паутины» возникает закономерный вопрос – строить ли сайт на основе уже существующей системы управления контентом (CMS) или разработать свой собственный движок.

Существует большое количество универсальных систем управления контентом, и во многих случаях их использование оправдано, но собственная CMS web-сайта обладает многими преимуществами – особенно для небольших систем, таких, например, как электронная библиотека учебного заведения или факультета. Ее архитектуру можно спроектировать в зависимости от предъявляемых требований и задач, а при необходимости дополнять впоследствии сторонними модулями или же собственными разработками. Большим плюсом является и сложность взлома собственной CMS (естественно, качественно выполненной), так как для взлома уникальной, нигде больше не установленной системы управления контентом злоумышленнику будет недоставать информации. Например, для распространенных движков существует богатый набор так называемых эксплоитов, с помощью которых злоумышленник может получить пароли от администраторской панели. Поэтому собственная CMS может оказаться надежнее, чем стандартная. Также необходимо задуматься и о системных ресурсах. Большинство функций стандартной CMS не используется, поскольку готовые движки рассчитаны на удовлетворение потребностей максимально широкого круга пользователей. Все неиспользуемые функции занимают место на диске, вынуждая пользователя покупать более дорогой хостинг.

В данной работе рассматриваются основные подходы к построению собственной системы управления контентом для библиотеки электронных публикаций, позволяющей авторам размещать на сайте свои публикации, управлять ими, получать рецензии и комментарии.

Выбор технологии и функционала системы электронных публикаций. Перед началом разработки необходимо решить, будет ли использована технология статических или динамических web-сайтов. Статический web-сайт – это набор связанных гиперссылками страниц, созданных на языке разметки HTML. На страницах статического web-сайта содержатся вместе и контент, и дизайн. У этого факта имеются негативные последствия, такие как трудоемкость публикации новых документов или редактирования существующих. Серьезной проблемой является и изменение архитектуры и представления web-сайта, так как этот процесс включает в себя изменение всех опубликованных страниц. Также на статических сайтах трудно осуществимы регистрация и аутентификация посетителей, назначение прав доступа пользователей к различным ресурсам web-сайта, возможность обратной связи. В таком случае не обойтись без программирования, так как включение скриптов решает не все проблемы. Динамические сайты для решения поставленной задачи более предпочтительны. Их принцип работы заключается в следующем: пользователь (и/или администратор) системы добавляет информацию на web-сайт, которая сохраняется в БД или файлах, затем посетитель web-сайта получает сохраненную информацию посредством web-браузера. Вид отображения информации зависит от представления (шаблона). Схема получения информации с динамического web-сайта показана на рис. 1.

Можно выделить следующие функциональные требования, необходимые и достаточные для большинства CMS электронных публикаций:

Для пользователя:

- регистрация и авторизованный доступ к ресурсу;
- публикация материалов как путем загрузки готовых файлов, так и с использованием встроенного редактора; возможность редактировать их (изменять текст, название, переносить в другой раздел), если у пользователя достаточен уровень привилегий;
- рецензирование статей, если у пользователя достаточен уровень привилегий;

- просмотр опубликованных статей, как открытых для всех, так и открытых только зарегистрированным пользователям, добавление комментариев к опубликованным материалам;
- поиск материалов по ключевым словам;
- подписка на новости;
- возможность восстановления пароля;
- редактирование личной информации.

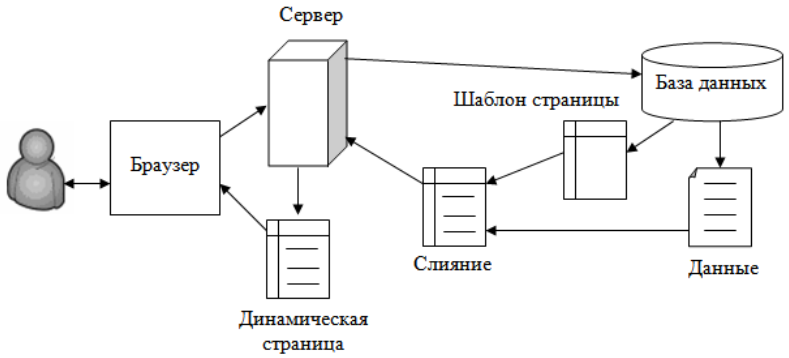

Рис. 1. Динамический сайт с CMS

Для администратора:

- авторизованный доступ к администраторской части;
- назначение прав доступа пользователей к различным ресурсам системы;
- добавление, удаление, редактирование пользовательских учетных записей;
- добавление, редактирование, удаление информации, содержащейся в БД, в частности добавление и удаление разделов публикаций, редактирование и удаление публикаций;
- администрирование системы.

Функции безопасности:

- защита паролей;
- защита от флуда и автоматической регистрации;
- проверка корректности введенных данных;
- защита БД от атак;
- защита автологина.

Каждая функция может быть реализована различным способом. Например, регистрацию пользователя целесообразно проходить в несколько этапов. Сначала пользователь заполняет регистрационную форму и проходит CAPTCHA-тест, позволяющий отсечь либо хотя бы резко сократить количество «роботов» при регистрации. Затем на его e-mail-адрес приходит письмо с кодом активации. После ввода кода активации в специальное поле на странице системы, пользователь получает доступ к ресурсам сайта.

На рис. 2 показана возможная функциональная схема системы электронных публикаций.

Взаимодействие с системой электронных публикаций осуществляется при помощи сверхтонкого клиента, в качестве которого выступает web-браузер. Обработку информации и формирования ответов (в виде HTML-страниц) будет осуществлять сервер подсистемы.

Разработанная нами система управления контентом представляет собой открытую систему с web-интерфейсом, выполненную по схеме двухуровневого программного комплекса, т.е. информация, находящаяся в базе данных, выбирается дополнительной программной утилитой (расширением web-сервера, в данном случае это интерпретатор языка динамического формирования HTML-страниц), формирующего окончательный HTML-код, который web-сервер передает клиенту, как показано на рис. 3.

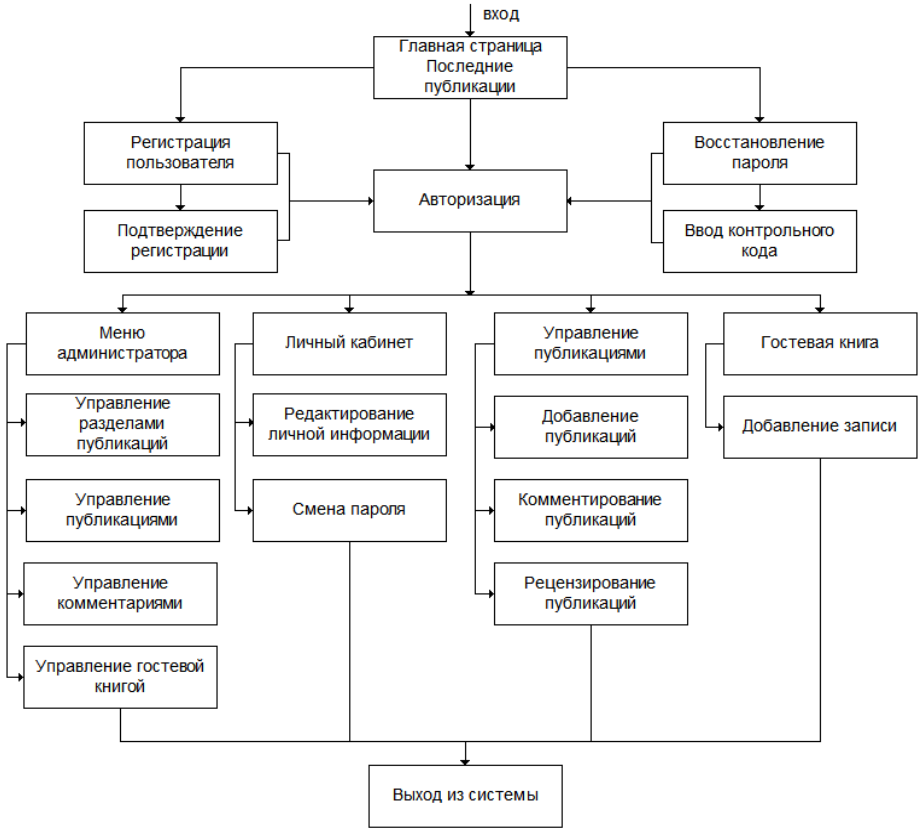

Рис. 2. Функциональная схема системы электронных публикаций

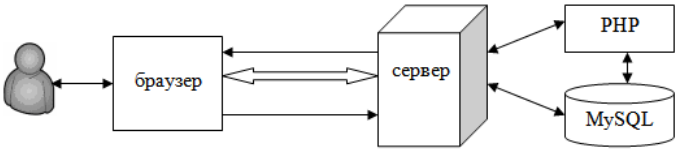

Рис. 3. Использование технологий динамического web-сайта

Реализация функций безопасности. Особое значение имеет надежная защита системы. Хотя конфиденциальных данных в нашей системе не так уж много (особенно, если все публикации сайта выложены в общий доступ), хакерская атака может привести к потере данных, вход от лица другого пользователя может привести к компрометации этого пользователя, в конце концов, сам сайт может быть «подменен». Рассмотрим реализацию основных функций безопасности, которые обязательно должны быть предусмотрены в CMS электронных публикаций.

1. Разграничение доступа. В нашей системе электронных публикаций предусмотрено 4 уровня привилегий, т.е. фактически реализована упрощенная мандатная модель разграничения доступа. Количество уровней может отличаться в разных системах, в зависимости от требований заказчика и функционала разрабатываемого сайта.

В зависимости от уровня пользователи имеют разный доступ к функциям и модулям CMS. Каждый модуль и функция CMS имеют свой уровень доступа (от 1 до 4). После авторизации пользователя его уровень доступа сравнивается с уровнем модуля или функции, к которым пользователь пытается получить доступ; если уровень пользователя больше уровня модуля, то пользователю разрешается доступ, а если меньше – запрещается.

2. Защита пароля. Шифрование. Никто, включая администратора, не должен знать пароль пользователя от его учетной записи. Поэтому для защиты пароля используется специальный алгоритм хеширования. Хеширование – это односторонняя функция, которая каждой введенной строке ставит в соответствие хеш-строку, длина которой никак не зависит от длины исходной строки и полностью определяется используемым алгоритмом хеширования (в нашем случае используется алгоритм хеширования md5). Односторонность хеширования означает, что по его результатам невозможно восстановить исходную строку, можно только подобрать коллизии, т.е. строки, хеширование которых дает нужный результат. На первый взгляд кажется невозможным перебрать огромное количество комбинаций, которое зависит только от длины пароля. Но у большинства пользователей пароль представлен цифрами, словарными словами и другими простыми комбинациями символов, поэтому задача упрощается. Кроме того, существуют сервисы, которые хранят терабайты хешей паролей. Поэтому в нашей системе пароль хешируется с добавлением нескольких случайных символов (так называемая «соль», хорошо знакомая по схеме аутентификации в Unix-системах). Соль можно вычислить, зная несколько настоящих паролей, но это крайне затруднительно. Взломать такой пароль методом перебора за разумное время без использования специализированных распределенных вычислительных систем практически невозможно.

Для минимальной защиты пароль должен быть не меньше 6-8 символов. Поэтому при регистрации должна проверяться длина пароля (минимальное значение устанавливается в файлах конфигурации).

3. Безопасность автологина. Для того чтобы при работе в браузере запоминался логин и пароль пользователя, используется механизм cookie. Это информация, запоминаемая браузером и предъявляемая для опознания по требованию сервера. Можно записать в cookie хэш пароля и при авторизации запрашивать пароль не из текстового поля, а сразу из cookie. Но любая информация, содержащаяся в cookie, уязвима. Поэтому нельзя заносить туда пароль даже в хешированном виде. Для решения этой проблемы сам хеш пароля будет храниться в базе, а для автологина будет использоваться хеш псевдаслучайной строки, который будет записываться в cookie и в специальное поле в таблице данных пользователя.

4. Защита от автоматических регистраций и спама. Для защиты от спама, отправления сообщений программами-роботами и множественных автоматических регистраций используется так называемая CAPTCHA.

CAPTCHA – это аббревиатура от выражения «Computer Aided Public Turing test to tell Computers and Humans Apart». Фактически, это тест Тьюринга для отличения человека от программы - задача, которую легко может выполнить человек, но которую невозможно или крайне трудно научить решать компьютер. Иногда CAPTCHA называют обратным тестом Тьюринга. Стандартный вид CAPTCHA – зашумленное случайное число, слово или иная надпись, которую пользователю

нужно прочитать и ввести прочитанный результат. Также существуют и другие реализации CAPTCHA. Например, требуется ввести не одно слово, а несколько или найти общий элемент, присутствующий на четырех изображениях, а также сказать о содержимом искаженного рисунка. Для людей с плохим зрением существуют звуковые тесты или задачи, требующие логического мышления.

На данный момент уже существует большой выбор среди производителей CAPTCHA. Этим занимаются как гиганты на рынке программного обеспечения, так и отдельные программисты. От квалификации производителя зависит вероятность обхождения теста. На качество защиты влияет правильное внедрение CAPTCHA. Часто очень хорошего качества тест можно легко обойти, потому что он используется без соблюдения политики безопасности.

Существует множество алгоритмов для распознавания CAPTCHA, поэтому одним из эффективных способов защиты является самодельная CAPTCHA. Писать анализатор для одного сайта бывает довольно хлопотно и накладно, так что некоторое время такая CAPTCHA будет являться хорошей защитой.

Алгоритм создания CAPTCHA в общем виде следующий:

1) создание строки из случайных символов;
2) создание пустого изображения или загрузка фона из файла;
3) вывод на этом изображении помех (случайные точки, линии);
4) вывод строки на этом изображении;
5) сохранение этой строки в сессии;
6) показ изображения.

Но может быть использован и один из стандартных вариантов построения CAPTCHA.

5. Защита от XSS и изменения формы. XSS, или так называемый «межсайтовый скриптинг», возникает, когда в генерируемые сервером страницы по какой-то причине попадают пользовательские скрипты. Специфика подобных атак заключается в том, что вместо непосредственной атаки сервера они используют уязвимый сервер в качестве средства атаки на клиента.

Злоумышленник публикует в формах на атакуемом сайте скрипт, например на языке JavaScript. Затем этот скрипт выполняется при открытии страниц пользователями web-сайта. Так как этот скрипт выполняется в браузере у пользователя, то он имеет доступ к информации в его cookie, а также может производить на сайте действия от имени пользователя, если тот авторизован, например, такие, как чтение, запись или удаление сообщений. В частности, если в форму для ввода пользовательских данных ввести код <script>alert(document.cookie)</script> и в системе отсутствует защита от подобных атак и такой текст появится в браузере в первозданном виде, то этот скрипт будет выполнен, и любой, кто откроет эту страницу, подвергнется XSS-атаке – будет получено содержимое файла cookie для данного сайта. А если вместо безобидного alert() поставить другой скрипт, который отправит эти данные на сайт-приемник (сниффер), то хакер сможет использовать их по своему усмотрению. Продемонстрированные действия ни в коем случае нельзя допускать, так как содержимое cookie – собственность пользователя и владелец сайта несет за нее ответственность. Кроме того, получив с помощью этой атаки нужный уровень доступа, хакер сможет удалить или изменить ценную информацию на самом сайте.

XSS может угрожать системе электронных публикаций, так как в ней предусмотрены регистрация и публикация информации пользователей, т.е. использование информации, вводимой пользователем. Также опасность может угрожать и администраторскому интерфейсу, так как в нем есть модули, предназначенные для просмотра данных, поступающих от посетителей сайта, например сообщения из форм обратной связи.

Основным способом противодействия XSS-атакам является обработка пришедших извне и публикуемых на сайте данных. Как правило, достаточно заменить символы "<" и ">" на "<" и ">" соответственно, а также кавычки и амперсанды с помощью функции htmlspecialchars(), при этом введенный посетителем текст теряет HTML-оформление, а содержащиеся в нем скрипты утрачивают вредоносность. В нашей CMS электронных публикаций обработка выполняется специальной функцией-оберткой htmlChars().

6. Защита от SQL-injection. SQL-injection используется для атаки web-сайтов, работающих с базами данных. Если в web-сайте имеется возможность вводить данные пользователем и если в SQL-запросах используются необработанные данные, то велика вероятность внедрения вредоносного SQL-кода.

SQL-запросы используются для извлечения, добавления, удаления и модифицирования информации из БД. Многие современные web-сайты используют скрипты и SQL для динамического формирования содержимого страницы. Без надлежащих мер защиты такой код может быть успешно выполнен на сервере.

Инъекция SQL является широко распространенным дефектом безопасности в Интернете, легко используется без специальных программ и не требует глубоких технических знаний. Использование этой уязвимости дает путь к большим возможностям, таким как:

- кража данных;
- отказ в обслуживании;
- подмена или уничтожение данных;
- другие случаи и намеренья.

Если есть необходимость (а она есть почти всегда) в текст запроса подставить данные, приходящие от пользователя, то очень высока вероятность, что в этом тексте окажутся несанкционированные SQL-команды. Это может нарушить работу приложения. Но самое страшное, что этим широко пользуются взломщики. Поэтому все данные, подставляемые в запрос, должны обязательно обрабатываться соответственным образом.

Продемонстрируем защиту CMS на примере.

В случае, если в системе имеются статьи только для просмотра зарегистрированными пользователями, то незарегистрированные не имеют доступа к ним.

Если не выполнять обработку данных хотя бы в одном запросе и если в тело запроса попадет не идентификатор статьи, а другая команда, то запрос изменится и результат будет непредсказуемым. Например, если в адресной строке набрать http://el/?page=main&rem=fulls&id=1+OR+1=1+--, то автоматически произведется доступ к закрытой статье. Доступ произойдет потому, что вместо нужного запроса получится совершенно другой, в котором есть команда OR 1=1, которая заставит вывести все статьи, а также экранирование конца запроса -- AND `access` = 0, что открывает доступ к закрытым статьям. Этот запрос выводит последнюю статью, и путем изменения идентификатора можно просмотреть любую статью. Для того чтобы избежать подобных проблем, числовые данные нужно приводить к типу integer функцией intval() или конструкцией (int).

Точно так же можно получить доступ в тело запроса, когда в запрос подставляются необработанные строковые данные. Дело в том, что они обрамляются апострофами, которые разграничивают данные и команды. Но если в тексте будет находиться апостроф, то все, что после него, попадет в тело запроса. Поэтому их нужно экранировать. Допустим, не происходит обработки данных, которые вводятся при авторизации пользователя. В таком случает, зная только один логин (а это не приватная информация), можно свободно получить доступ к любому аккаунту, вставив в поле «логин» текст «Логин’ OR 1=’1» . В итоге получится со-

вершенно другой запрос, где пароль становится необязательным атрибутом. Поэтому все внешние строковые данные должны обрабатываться функцией mysql_real_escape_string(), которая экранирует апострофы. Если данные обработаны, то введенный запрос не будет выполнен.

7. Защита от PHP-injection. PHP-инъекция предполагает выполнение постороннего кода на сервере web-сайта.

Для того чтобы осуществить такую атаку, нужно загрузить на сервер исполняемый файл. Возможным это становится тогда, когда при загрузке не проверяются расширения файлов. Механизм очень прост. Достаточно поместить в любое место скрипт eval ($_POST[‘hack’]), и тогда хакер спокойно загрузит свой скрипт с расширением php и получит полное управление сайтом. Для этого достаточно сделать форму типа

```
<form action="http://my cms/photo/30.php" method="post">
<textarea name="hack" cols="40" rows="10"></ exta-
rea><br>
<input name="" type="submit">
</form>
```

и ввести в поле формы текст unlink(‘../index/php’); после выполнения кода главная страница системы публикаций пропадет. Поэтому нужно обязательно проверять загружаемые файлы на допустимые типы загрузки.

8. Защищенность системных файлов. Одним из главных аспектов безопасности системы является правильно организованная структура, а также текущие и глобальные настройки конфигурационного файла. Необходимо предусмотреть защиту файлов конфигурации, чтобы хакер не проник в них и не внес нежелательные изменения, если, к примеру, случайно окажется включенной директива register_globals. Для этого необходимо установить константу define ('IRB_KEY', true) в файле index.php, которая отвечает за доступ к файлу. Каждый файл php должен содержать проверку на наличие этой константы. Если в адресной строке будет прописан путь до конфигурационного файла, то сервер выдаст ошибку и остановит выполнение скрипта. Файл .htaccess также должен быть правильно настроен. Должны быть отключены глобальные переменные и magic_quotes_gpc (магические кавычки), регистрация глобальных переменных, а также вывод ошибок на экран.

Заключение. В работе представлены основные подходы к построению систем управления контентом библиотеки электронных публикаций на примере разработки, которая обеспечивает функции регистрации, аутентификации, разграничения доступа, публикации и редактирования материалов, защищенное хранение пользовательских данных. В качестве практического результата работы была разработана собственная CMS электронных публикаций, отвечающая представленным требованиям.

Выделим следующие особенности и конкурентные преимущества разработанной CMS:

- возможность интеграций в структуру различных дополнительных модулей благодаря понятной структуре системы;
- использование готовых модулей для проектирования CMS другой направленности;
- простота шаблонов и их легкое настраивание;
- скорость исполнения и ресурсобережливость (в оперативную память грузятся только те файлы, которые действительно принимают участие в формировании данной страницы);
- прозрачность кода.

Кроме того, система управления электронными публикациями отвечает всем современным требованиям по безопасности и предоставляет защиту от основных угроз. Были проведены эксперименты, которые показали устойчивость разработанной системы к перечисленным угрозам и ряду сетевых атак.

Представленная система электронных публикаций является модульной системой, что позволяет с минимальной настройкой использовать проект для создания других подобных комплексов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. М. Дизайн: Управление сайтом [электронный ресурс] / Режим доступа: http://mdesign.ru/publications/cms/40b7504e10e58?start=, дата обращения: 03.12.2010, свободный. – Загл. с экрана.
2. Open Questions Blog Optimizer: CMS и все о них [электронный ресурс] / Режим доступа: http://www.oqbo.ru/read.php?block=25, дата обращения: 03.11.2010, свободный. – Загл. с экрана.
3. *Колисниченко Д.И.* Движок для вашего сайта. – СПб.: БХВ-Петербург, 2008. – 368 с.

Пескова Ольга Юрьевна
Технологический институт федерального государственного автономного образовательного учреждения высшего профессионального образования «Южный федеральный университет» в г. Таганроге.
E-mail: poy@tsure.ru.
347928, г. Таганрог, пер. Некрасовский, 44.
Тел.: 88634312018.

Горло Надежда Евгеньевна
E-mail: serebro@list.ru.

Peskova Olga Yur'evna
Taganrog Institute of Technology – Federal State-Owned Autonomy Educational Establishment of Higher Vocational Education "Southern Federal University".
E-mail: poy@tsure.ru.
44, Nekrasovskiy, Taganrog, 347928, Russia.
Phone: +78634312018.

Gorlo Nadezda Evgen'evna
E-mail: serebro@list.ru.

УДК 004.732.057

М.Л. Лопатин, О.Ю. Пескова

АНАЛИЗ ЗАЩИЩЕННОСТИ СОВРЕМЕННЫХ СИСТЕМ УПРАВЛЕНИЯ КОНТЕНТОМ

Представлено понятие систем управления контентом (CMS). Рассмотрены основные свободно распространяемые CMS, активно применяемые для создания и управления сайтами: Joomla!, WordPress, Drupal. Рассмотрены их основные характеристики, особое внимание уделено вопросам безопасности, таким, как особенности подсистем безопасности, наиболее критичные уязвимости, возможные атаки на CMS. Приведены рекомендации по настройке и применению рассмотренных систем управления контентом, позволяющие повысить защищенность клиентских систем, построенных на их основе.

Сетевые уязвимости; системы управления контентом (CMS); динамическое управление сайтом.