УДК 004.056.53

Сахарова С.В., Брицкий С.В., Гуржий Н.А.

ВОЗМОЖНЫЙ ПОДХОД ДЛЯ СОКРЫТИЯ ПУТЕЙ И ИМЕН ФАЙЛОВ В WEB-ПРОСТРАНСТВЕ

Описан подход, который может быть использован для сокрытия путей к файлам на web-сервере путем перенаправления запросов через .htaccess. Рассмотрен один из вариантов проникновения злоумышленника в систему с использованием уязвимости скрипта выдачи информации.

Ключевые слова: несанкционированный доступ, скрипт, копирование информации

Введение

С момента, когда пользователям Интернет была предоставлена возможность свободно обмениваться ссылками на файлы, прошло уже много лет. Методы, используемые для копирования файлов пользователями, постоянно меняются. Владельцы и администраторы различных *web*-ресурсов непрерывно занимаются поиском и разработкой методов для сокрытия путей к файлам. Сначала, это нужно было для ведения статистики операций, проводимых с файлами. Однако, в тот момент, когда информация о возможностях *PHP Injection* и *Mysql Injection* стала общедоступной, появилась необходимость приступить к поиску все новых и новых методов сокрытия путей и имен файлов в *web*-пространстве. Важно отметить, что в результате своей деятельности, злоумышленник с помощью специальных методов может не только навредить удалением файлов или папок, но и, что более важно, рассекретить и раскрыть важную информацию. Поэтому возникла задача, связанная с защитой информации, размещенной на сайтах от несанкционированного копирования.

Общепризнанным фактом является то, что право всегда отстает от общественных отношений. И это понятно, поскольку право тогда начинает регулировать отношения, когда они начинают нуждаться в этом. И Интернет стал удобной площадкой для нарушения авторских прав, а с развитием сети эти нарушения приобрели массовый характер. В данной работе показана попытка защитить авторские права на текстовую информацию с условием ее свободного чтения, но без возможности загрузки или редактирования. Есть сайт, на котором нужно содержать тематические материалы: книги, конспекты, материалы учебного характера. Пользователям, которым дана и видна ссылка на материал нужно иметь доступ только к чтению материалов, без возможности копировать и загружать их.

Основная часть

Как известно модули для отображения какой-либо информации, размещенной на сайте, связаны между собой. В случае если удается определить уязвимость одного из модулей, то вполне вероятно, что с ее помощью возможно осуществить взлом и проникновение ко всей системе в целом. Один из способов взлома *web*-ресурсов, работающих на *PHP*, заключающийся в выполнении постороннего кода на серверной стороне называют *PHP*-инъекция (англ. *PHP injection*). Приведем пример с *PHP Injection,* который был обнаружен в исходных кодах модуля одного из *web*-ресурсов. 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 *web*-сервером без проверки.

Предположим, что у нас есть уязвимый код, с помощью которого *web*-ресурс позволяет пользователю видеть или копировать определенную информацию.

Рассмотрим пример копирования какого-либо файла.

```
If (@$_GET['download'])
        $file=$_GET['download'];
if (file_exists($file)) {
header('Content-Description: File Transfer');
header('Content-Type: application/octet-stream');
header('Content-Disposition: attachment; filename='.basename($file));
header('Content-Transfer-Encoding: binary');
header('Expires: 0');
header('Cache-Control: must-revalidate, post-check=0, pre-check=0');
header('Pragma: public');
header('Content-Length: ' . filesize($file));
ob_clean();
flush();
readfile($file);
exit;
}
```

При использовании кода *download.php?download=config.php* злоумышленник получает файл конфигурации или любой другой интересующий его файл. Уязвимый код может быть не явным и являться частью другой подпрограммы, например менеджера файлов для администратора. Если злоумышленнику известны пути других, возможно более важных, файлов, (например, на этом же *web*-ресурсе могла размещаться открытая ссылка на файл «*/biblio/free_version_my_book1.pdf*»), в этом случае вся размещенная на этой ресусш информация становится доступной как для искажения так и для коирования.

Путем написания несложной программы *«bruteforce»* можно подобрать имена других файлов, которые лежат в директории «/*biblio*», чтобы получить информацию, не предназначенную широкому кругу лиц (личные разработки, тексты, графики и т.д.). Однако можно проследить явную связь между уязвимостью одного модуля и получением информации из другого, иногда даже не связанного с ним единой базой или кодом.

Сокрытие путей файлов с помощью *.htaccess* (сервер *Apache*). Файл *.htaccess* (от. англ. *hypertext access*) – это файл дополнительной конфигурации *web*-сервера *Apache* и подобных ему других серверов, позволяющий задавать большое количество дополнительных параметров и разрешений для работы *web*-сервера в отдельных каталогах, таких как, например, управляемый доступ к каталогам, переназначение типов файлов и т.д., без изменения главного конфигурационного файла.

Модуль *mod_rewrite* – это модуль для *web*-сервера *Apache*, который позволяет переписывать *URL* «на ходу». Допустим, имеется ссылка вида

http://test.name?id=123,

а хотим получить возможность использовать ссылку (с сокрытием путей):

http://test.name/article/123.

Для этого потребуется использовать файл *.htaccess* следующего содержания:

```
RewriteEngine on
RewriteRule ^article/([0-9]+)/? index.php?id=$1 [L]
```

Код *.htaccess* обладает большими возможностями интерпретации *URL* перед обработкой необходимым модулем, а самое большое его преимущество – это поддержка

работы с регулярными выражениями, благодаря которым возможно описать практически любые ситуации для замены *URL*.

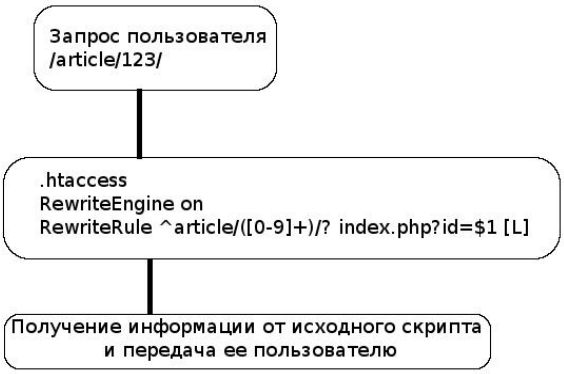

Рис. 1. Алгоритм роботи *.htaccess* (mod_rewrite)

Для того чтобы пользователь мог смотреть и читать файл «онлайн» нужна специальная программа (скрипт) для отображения графических данных. Почти все варианты скриптов для отображения реализованые для текстовых данных не подходят, поскольку для решения задачи необходим скрипт, способный расформировать PDF файл на страницы в качестве изображений, а не текста. Все скрипты, написаные на языке программирования *php* не имели такой возможности, поэтому выбор пал на язык программирования JavaScript с использованием библиотеки Jquery и плагином «*pdf*».

В код плагина были внесены следующие изменения: удалена возможность скачивания файла и выполнена блокировка клавиши «*Printscreen*». Алгоритм работы этого кода такой, что при нажатии клавиши *PrintScreen* фрагмент кода (*body*) скрывается и нельзя сделать снимок экрана. Пример отображения файла «*pdf*» в браузере показан на рис. 2.

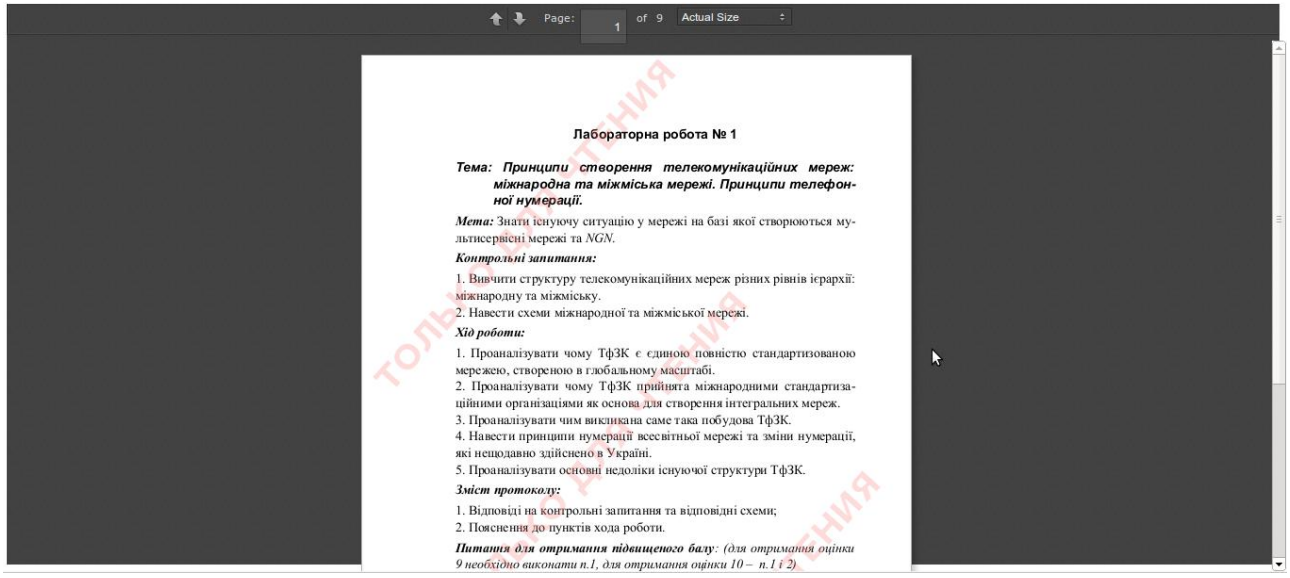

Рис. 3. Пример отображения файла «*pdf*» в браузере

Для того чтобы пользователь не видел ссылки на файл, его имя передается в зашифрованном виде. Для этого был выбран следующий алгоритм: кодируем имя файла шифром *MD5,* и когда в запросе *GET* передается его скрипт, он проверяет существует ли такой файл в каталоге на сайте, если есть, тогда отдает его содержание в наш скрипт для отображения самого файла *pdf*. Для того чтобы помешать исследованию кода *javascript* и выяснению принципов его работы использован обфускатор кода, при котором код

усложняется за счет разложения операторов на меньшие части и использования более сложных математических преобразований

Теперь, когда скрипт для отображения и скрипт для кодирования работают, остается разместить их на сайте, при этом в ходе установки и интеграции скрипта в систему *Wordpress* было обнаружено, что права на файлы и директории имеют очень большое значение. Для файлов необходимо иметь права, позволяющие открывать и читать файл такие же и на всю директорию. Это относится ко всем файлам, а не только к файлам, имеющим расширение *pdf*. В случае если этого не сделать, то у нас не будет возможности пользоваться скриптом. Следует заметить, что правильное выставление прав доступа действительно для всех *Linux* систем.

Заключение

Таким образом, рассмотрен простой пример сокрытия путей на популярном сервере *Apache* с использованием файла *.htaccess.* Предложенный подход, позволяет получить скрипт, который может быть интегрирован в систему управления сайтом и функционировать в ней без ущерба или внесения изменений в основной системе, любой сбой или неполадка никак не повлияет на главную систему, потому что сама система автономна и может использоваться и без систем управления сайтом. Благодаря кодированию имени файла получена анонимизация и защита от прямого копирования, а благодаря использованию специального скрипта отображения получена возможность комфортного чтения без возможности копирования. Скорость работы скрипта зависит только от количества файлов в директории. Возможности применения такого подхода гораздо шире, чем рассмотренный пример. Например, можно отдельно загружать *css* файлы и *js* скрипты. Так же можно показывать пользователю поддельные пути, в случае со злоумышленником это усложнит подбор пути. Важно понимать, что сокрытие путей не только защищает от *PHP Injection,* но и от загрузки файла на сервер путем взломанной базы данных *MySQL* (*«into outfile»*), так как для этого нужно знать пути для записи в разрешенные директории.

К достоинствам описанного подхода можно отнести:

– сокрытие путей к файлам;

– создание легко запоминаемых *URL web*-ресурсов;

– уменьшение ошибок при использовании *URL* (например, при передачи голосом по телефону);

– уменьшение времени на понимание структуры *web*-ресурса;

– легкость в перемещении по иерархии *web*-ресурса путем текстового изменения части *URL*-пути.

К недостаткам относится следующее:

– увеличение затрат ресурсов сервера для большинства реализаций;

– усложнение настройки *web*-ресурса.

Тем не менее, при нынешнем развитии технологий данные недостатки несущественны и ими можно пренебречь.

ЛИТЕРАТУРА

1. Шлосснейгл Д. Профессиональное программирование на PHP. – М.: Вильямс, 2006. – 624 с.
2. Фленов М. PHP глазами хакера. – М.: БХВ-Петербург, 2010. – 336 с.
3. Хефлин Д., Ней Т. Разработка *Web*-скриптов. Библиотека программиста. – СПб.: Питер, 2001. – 496 с.
4. Основы информационной безопасности: курс лекций, учебное пособие. Изд. третье /Галатенко В.А. Под ред. Акад. РАН В.Б. Бетелина// М.: ИНТУИТ.РУ «Интернет-университет Информационных Технологий», 2006. – 208 с.

Надійшла: 12.11.2013р.

Рецензент: д.т.н., професор Бурячок В.Л.