Дылдин И.А.
Некоторые аспекты безопасности интернет-сайтов. Россия, Екатеринбург, УрГУПС
dyldinilyas@gmail.com

**Введение.**

Для систем, рассчитанных на постоянную работу (например, для серверов), безопасность играет весьма значительную роль. Веб-серверы — это основа Интернета. Они отвечают за базовые услуги и функционирование миллиардов веб-сайтов по всему миру, в результате превращаясь в хранилище персональных данных посетителей. Обеспечение защиты серверов от атаки извне — это одна из важнейших задач любой полагающейся на них организации.

В последние годы число атак, нацеленных на веб-сервера, значительно возросло. Географическое местоположение сервера роли при этом не играет: вредоносный код не обращает внимания на границы. Угроза, принимающая международный характер, теперь проистекает от организованных преступных сообществ, занимающихся массовым сбором паролей, финансовых сведений и другой информации; это уже не просто малолетние хакеры-хулиганы. В большинстве случаев атака происходит с минимальным вмешательством, а вредоносное ПО, размещаемое на серверах и сайтах, рассчитано на заражение максимального количества пользователей.

Веб-серверы особенно уязвимы в силу своей открытости — по своей природе они рассчитаны на обмен информацией с пользователями. Злоумышленник может внести модификации в код HTTP сервера или сервера базы данных, или самих страниц веб-сайта, поменяв его изначальную функциональность.

Не следует, однако, считать, что веб-сервер защитить невозможно. Это вполне реально, но для этого требуются совместные действия администраторов веб-сайтов, программистов и проектировщиков; такие вещи, как антивирусное ПО, операционные системы и права доступа, требуют постоянного внимания.

В данной научной работе рассматриваются некоторые распространенные способы компрометации веб-серверов и методы противодействия этому.

**Основы безопасности интернет-сайтов.**

Первый этап проектирования, создания или использования безопасного веб-сайта — это обеспечение максимального уровня безопасности сервера, на котором он размещается.

Веб-сервер формируется несколькими слоями ПО, каждый из которых подвержен разнообразным способам атаки, как показано на диаграмме. Целью атаки может стать любой из блоков.

Основа любого сервера — это операционная система. Обеспечить ее безопасность сравнительно просто: достаточно вовремя устанавливать последние обновления системы безопасности. Это не самое трудоемкое занятие, поскольку системы Microsoft и многие представители семейства Linux позволяют организациям устанавливать исправления автоматически или запускать их одним щелчком мыши.

Следует помнить, что хакеры также склонны автоматизировать свои атаки, используя вредоносное ПО, перебирающее один сервер за другим в поисках сервера, где обновление не было установлено. В связи с этим важно следить за тем, чтобы обновления устанавливались своевременно и надлежащим образом; любой сервер, на котором установлены устаревшие версии обновлений, может подвергнуться атаке.

*Блок-схема 1. Строение веб-сервера.*

Следующий важный момент — это антивирусное ПО. Его использование является обязательным требованием для любого веб-сервера вне зависимости от того, используется в качестве платформы Windows или Unix. В сочетании с гибким межсетевым экраном антивирусное ПО становится одним из самых эффективных способов защиты от угроз безопасности. Когда веб-сервер становится целью атаки, злоумышленник без промедления пытается загрузить инструменты взлома или вредоносное ПО, чтобы успеть использовать уязвимость системы безопасности до того, как она будет закрыта. При отсутствии качественного антивирусного пакета уязвимость системы безопасности может долгое время оставаться незамеченной.

В вопросах защиты оптимальным является многоуровневый подход. На переднем крае — межсетевой экран и операционная система; стоящий за ними антивирус готов заполнить любые возникающие бреши.

**Защита веб-сервера.**

Обычно антивирус выступает последней линией обороны в случае атаки. В связи с этим на веб-серверах (в особенности там, где используется динамически формируемый контент) должна быть включена проверка файлов при обращении. От вредоносного ПО не застрахован ни один сервер. Вне зависимости от вашей уверенности в безопасности веб-сервера всегда есть шанс успешной атаки. Проверка файлов при обращении позволяет существенно снизить вероятность запуска вредоносного кода в системе, поскольку она может вестись как в режиме «при чтении», так и в режиме «при записи», что позволяет своевременно уведомить администратора о попытке размещения вредоносного кода на сервере.

Несмотря на то, что проверка файлов при обращении может сказаться на производительности сервера, ее преимущества для безопасности значительно превосходят любое снижение пропускной способности сервера.

Атаки веб-серверов можно разделить на две категории: локальные и глобальные.

* Локальные атаки обычно направлены на кражу информации или перехват управления на отдельном веб-сервере.
* Глобальные атаки обычно направлены на несколько веб-сайтов и ставят своей целью заражение всех их посетителей.

Несмотря на то, что в некоторых кругах системы Linux и BSD считаются безопаснее Windows, использование этих операционных систем также не является гарантией защиты от организованной преступности. В них можно (и нужно) установить антивирусное ПО. Даже в случае, когда вредоносная программа не может быть запущена на самом сервере, если он защищен антивирусным ПО, она все же может передаваться под видом обычного контента посетителям веб-сайтов, поскольку хакеры зачастую загружают такие программы с помощью PHP или ASP, что исключает необходимость заражения операционной системы веб-сервера.

Серверы также могут заражаться через локальную сеть. К примеру, семейство червей Fujacks способно заражать HTML-, PHP- и ASP-файлы, расположенные на сетевых дисках.

**Безопасное проектирование.**

Планирование является неотъемлемой частью системы безопасности, поскольку оно позволяет сократить ущерб от вирусов, шпионского ПО и других вредоносных программ. Безопасное проектирование состоит из следующих аспектов:

**Файлы cookie.**

Одна из основных проблем при проектировании веб-приложения заключается в том, что запрос новой страницы всегда обрабатывается вне контекста предшествующих запросов. Попросить веб-приложение «запомнить пользователя» сложнее, чем обычно.

Большинство браузеров поддерживают два метода, которые веб-приложения могут использовать для «запоминания» посетителей: обычные файлы cookie и файлы cookie сеанса.

* Файл cookie — это небольшой файл, создаваемый браузером и хранящийся на компьютере пользователя. Его содержимое не регламентируется, но обычно в таких файлах хранится название, дата окончания срока действия и некий объем данных, например:«Count = 100» или «Member = false».
* Файл cookie сеанса похож на обычный, но при этом он позволяет веб-приложениям хранить данные в памяти.

Различие заключается в том, что обычный файл cookie сохраняется на компьютере пользователя и остается на нем до момента удаления пользователем. Файл cookie сеанса, напротив, хранится только на протяжении времени работы компьютера и автоматически теряется при закрытии приложения-браузера. Тем не менее, у них есть общая черта: они подвержены манипуляциям извне.

Разработчики часто склонны считать данные из файлов cookie надежными, поскольку считают, что все должно быть в порядке, поскольку они сами разрабатывают соответствующий код. Они ошибаются. Хакер может с легкостью изменить файл cookie (а в некоторых случаях — и данные активного сеанса), чтобы обманом заставить веб-сайт предоставить доступ к закрытой странице.

При проектировании системы никогда нельзя полагаться на надежность пользовательских данных, вводимых посетителями напрямую или поступающих через файлы cookie. Стремитесь ограничить объем данных, сохраняемых в файлах cookie, в особенности тогда, когда эти данные не следует хранить в открытом доступе. Оптимальный подход — считать все данные, хранящиеся на компьютере пользователя, ненадежными.

В 2007 году сайт MySpace.com подвергся атаке с помощью троянаJS/SpaceStalk-A, который крал информацию, хранящуюся в файлах cookie, и передавал ее на удаленный сервер. Такая информация может содержать конфиденциальные сведения — имена пользователей, адреса предпочитаемых сайтов и пароли.

**Аутентификация.**

Если какие-то области веб-сайта должны быть доступны только некоторым клиентам или зарегистрированным пользователям, для подобного разграничения доступа потребуется метод проверки подлинности пользователей<http://help.yandex.ru/webmaster/?id=1071331>.

Существует несколько способов аутентификации пользователей: базовая аутентификация, дайджест-аутентификация и HTTPS.

* При использовании базовой аутентификации имя пользователя и пароль включаются в состав веб-запроса. Даже если контент с ограниченным доступом не слишком важен, этот метод лучше не использовать, так как пользователь может применять один и тот же пароль на нескольких веб-сайтах. Опрос Sophos показал, что 41% пользователей применяют для всей своей деятельности в Интернете всего один пароль, будь то сайт банка или районный форум. Старайтесь защищать пользователей от подобных ошибок, используя более безопасные методы аутентификации.
* Дайджест-аутентификация, поддерживаемая всеми популярными серверами и браузерами, позволяет надежно шифровать имя пользователя и пароль в запросе. Она помогает обеспечить безопасность имен и паролей, что производит соответствующее впечатление на пользователей и снижает вероятность успешной атаки на сервер.
* Протокол HTTPS позволяет шифровать все данные, передаваемые между браузером и сервером, а не только имена пользователей и пароли. Протокол HTTPS (основанный на системе безопасности SSL) следует использовать в случае, если пользователи должны вводить важные личные данные — адрес, номер кредитной карты или банковские сведения.

При выборе системы аутентификации рекомендуется использовать самый безопасный вариант из имеющихся в наличии. Другие варианты отпугнут клиентов, заботящихся о защите своих данных, и могут привести к возникновению излишнего риска для пользователей.

 **Лог-файлы.**

Серверные лог-файлы — весьма важный инструмент управления веб-сайтом. Обычно HTTP-серверы способны вести раздельные журналы доступа и ошибок; при наличии подобной функциональности ее следует использовать в обязательном порядке, поскольку это может пригодиться для последующего анализа.

Лог-файлы также следует регулярно просматривать, поскольку они могут помочь в более глубоком знакомстве с угрозами, которым подвергаются веб-сайты. Лог-файлы позволяют отслеживать потенциальные угрозы безопасности, так как в них детально фиксируются все успешные и предпринятые попытки обращения к сайту.

**Обеспечение безопасности кода.**

Разрабатывать безопасный код не всегда так просто, как кажется. Для этого нужны не просто опытные программисты — также требуются знания о конкретных проблемах безопасности. Основными атаками через код сайта являются SQL-инъекция и XSS (межсайтовый скриптинг).

**SQL-инъекция**

SQL-инъекция используется для атаки веб-сайтов, работающих с базами данных. Возможность внедрения SQL-кода возникает, если в SQL-запросах используются неотфильтрованные данные, вводимые пользователями.

SQL-запросы используются для извлечения информации из базы данных, добавления информации в базу данных, а также для изменения и удаления данных в базе. Многие современные веб-сайты используют скрипты и SQL для динамического формирования содержимого страницы. В SQL-запросах часто используются данные, вводимые пользователями; это может привести к угрозе безопасности, поскольку хакеры могут попытаться внедрить во входные данные вредоносный SQL-код. Без надлежащих мер защиты такой код может быть успешно выполнен на сервере.

К счастью, этот вид угрозы легко устраним благодаря проверке вводимых пользователем данных. В PHP имеется специальная функция **mysql\_real\_escape\_string**, удаляющая из строки потенциальный код SQL-инъекции. Ее следует использовать для фильтрации всех данных, внедряемых в SQL-инструкции.

**XSS**

Данный вид атак направлен на веб-сайты, отображающие вводимые пользователями данные. Вместо попытки получения контроля над базой данных путем ввода вредоносного кода злоумышленник пытается атаковать код самого веб-сайта, внедряя в него вредоносные сегменты.

Многие сайты хранят имена всех посетителей в базе данных, чтобы иметь возможность отображать их при вводе соответствующих пользователей. Злоумышленник может создать подложную учетную запись, разместив при этом в поле имени вредоносный код. Подобные атаки обычно реализуются с помощью вредоносных скриптов на языке Javascript, которые затем загружают контент с другого веб-сайта. Предполагается, что в базе данных хранится имя пользователя, но на самом деле в данном случае это будет вредоносный код. Соответственно, если веб-сайт отображает имя пользователя в верхней части страницы, то этот код будет выполнен. Поскольку при наличии определенных условий такой код может делать практически все, что угодно, угроза становится вполне реальной; тем не менее, разработчики зачастую про нее забывают. За последнее время жертвами XSS-атак стали многие популярные веб-сайты, в том числе MySpace, Facebook, Google Mail, ВКонтакте.

К сожалению, XSS-атакам зачастую трудно что-либо противопоставить, поскольку для этого необходимо должным образом фильтровать вводимые и выводимые данные, а также все поля, которые могут меняться пользователями. Сюда относятся данные, получаемые из запросов GET и POST, а также запросы, возвращаемые из базы данных.

В PHP имеется целый ряд пакетов, которые помогают фильтровать выводимые данные, например, CodeIgniter. <http://help.yandex.ru/webmaster/?id=1071331>Также в PHP имеется встроенная функция **htmlspecialchars**, которую можно использовать для фильтрации выводимых данных.

**Заключение.**

На данный момент на просторах интернета не существует не решенных проблем в проектировании интернет-сайтов. В случае возникновения таких проблем, они быстро решаются пользователями всемирной паутины.