УДК 004.056

М.А. Стюгин

Защита интернет-ресурсов по технологии движущейся цели

Представлен новый метод построения защищенных интернет-приложений, основанный на технологии движущейся цели (moving target defense). Сделано расширение данной технологии к общей проблеме защиты систем от исследования и разработан алгоритм построения защищенных от исследования систем с неограниченным количеством дополнительных параметров. На основе данного алгоритма можно строить системы защиты от атак SQL-инъекций, CSS-атак и пр. Взаимодействие с интернет-приложением построено таким образом, что когда злоумышленник пытается исследовать ресурс, он получает информацию, интерпретируя которую, он получает все более и более сложную структуру системы. Данные технологии позволяют защитить веб-сайты от несанкционированного доступа и исследовать поведение злоумышленника. Исследование поведения злоумышленника дает возможность поиска уязвимостей, не известных на этапе проектирования системы. В данной работе представлен пример реализации подобной технологии.

Ключевые слова: информационная безопасность, технология движущейся цели, SQL-инъекции, защита от исследования.

Предоставление надежной защиты для веб-сайтов является наиболее сложной задачей разработки систем информационной безопасности. Веб-приложения, как правило, должны быть доступны для всех пользователей в Интернете. И это является причиной того, что существует соответствующая плоскость атаки, которую невозможно устранить. Поскольку веб-сайты получают данные из внешней среды, это создает уязвимости, которые могут быть реализованы злоумышленниками. Риски безопасности подобных ресурсов не могут быть определены заранее, поскольку существует множество уязвимостей, которые будут открыты в будущем. Данные уязвимости могут быть критическими для системы. Кроме того, злоумышленник может осуществить рекогносцировку системы перед атакой на нее, что может дать ему дополнительное преимущество над защитником. Существует общая технология устранения таких преимуществ злоумышленника. Данный метод назван защитой на основе движущейся цели (moving target defense – MTD) и основан на непрерывном изменении системы. Поэтому когда атакующий осуществляет рекогносцировку, он не может получить актуальную информацию, которую можно использовать в следующий момент времени.

Практически любой атаке предшествует процесс рекогносцировки. На основании полученной информации злоумышленник пытается реализовать те или иные эксплойты. Введя в информационную систему динамические параметры, мы можем сделать процесс рекогносцировки бесполезным или малоэффективным. Основанная на этом технология MTD уже стала неким трендом построения систем информационной безопасности. Наиболее полное изложение технологии и ее применения можно найти в публикациях [1] и [2]. Существуют практические реализации таких методов в области виртуализации [3], программно-определяемых сетей [4], рандомизации виртуальной памяти (address space layout randomization – ASLR) и пр. Существуют подобные реализации и в области защиты веб-серверов [1]. Однако все они, как правило, сводятся к динамическим именам переменных, таблиц в базах данных и пр.

Недостатком всех предложенных методов в области MTD является то, что схемы защиты на их основе также остаются стереотипными. Другими словами, если злоумышленник изучил конкретный метод MTD на одной системе, то он может использовать полученную информацию для исследования других систем, где применяется тот же метод MTD. Проблема заключается в том, что сама схема MTD также должна быть защищена от исследования. Это требует более общей постановки задачи защиты от исследования и получения метода MTD, который позволял бы строить нестереотипные системы за счет введения дополнительных параметров.

Целью данной работы является разработка алгоритма MTD, защищенного от исследования злоумышленником, и построения на его основе метода защиты интернет-ресурсов от преднамеренных атак.

Проблема защиты от исследования систем. Получение новой информации – это исследование системы, при котором мы интерпретируем информацию в соответствии с гипотезами относительно ее структуры. То есть даже в исследовании некая информация всегда априорно предшествует действиям, иначе мы не сможем интерпретировать получаемую от системы обратную связь. На этом основывается принцип защиты от исследования систем: чтобы сохранить функциональную структуру системы от несанкционированных воздействий, надо ее разнообразить до такой степени, чтобы для непосвященного исследователя она представляла хаос, относительно структуры которого трудно сформулировать однозначную гипотезу. Достаточно подробно этот вопрос был раскрыт в [7]. В данной статье мы не будем подробно рассматривать методологию защиты от исследования систем, а сосредоточимся на выводе алгоритма модификации структуры информационной системы, сходной по своей сути с технологией MTD.

Таким образом, защита от исследования не накладывают каких-либо ограничений на действия злоумышленника, а управляет информацией, на основе которой он принимает решения. Это дает снижение риска сразу по всему множеству (даже не открытых) уязвимостей, так как блокирует информативную обратную связь при попытке исследования системы. Технология, о которой идет речь в данной работе, основана на искусственном усложнении системы путем введения в нее «бессмысленных» параметров. Однако результат, который мы достигаем в результате ее реализации, заключается как раз в отклонении структуры системы в область нестереотипных шаблонов вплоть до полного хаоса для стороннего наблюдателя.

Предлагаемая технология защиты от исследования заключается в функциональном изменении процессов в системе в соответствии с некой концепцией уникальности. Полученный алгоритм показан на рис. 1.

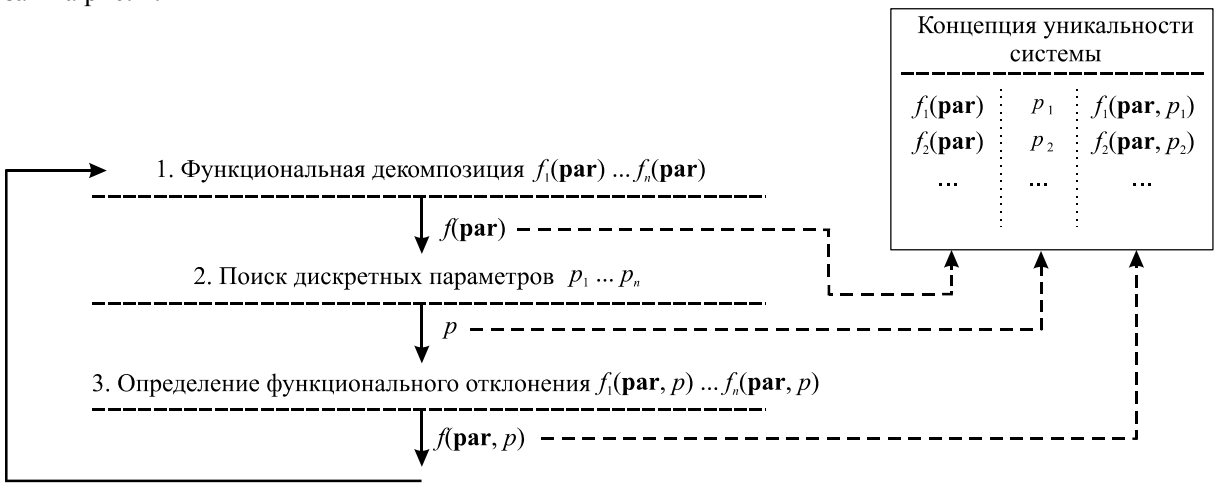

Рис. 1. Алгоритм построения «уникальной» системы

На первом шаге мы производим функциональную декомпозицию системы, целью которой является выделение подпроцессов для конкретной цели ($f_1(\mathbf{par})...f_n(\mathbf{par})$), где **par** – любые параметры процесса (множество). Например, доступ к базе данных – это последовательная аутентификация, соединение с базой данных и формирование запроса. Аутентификацию в свою очередь можно разбить на ввод логина и пароля и т.д.

Второй шаг – для конкретных процессов, полученных в ходе функциональной декомпозиции, мы ищем возможные дополнительные параметры, имеющие дискретный характер ($p_1...p_n$). Например, время в минутах, позиция символов, номер сессии и т.д.

После определения дискретных параметров мы вводим бессмысленное отклонение исходного процесса по его значению ($f_1(\mathbf{par},p)...f_n(\mathbf{par},p)$). Например, для ввода пароля – это смещение символов на клавиатуре в зависимости от их позиции в строке, для коммутации – это перераспределение портов и адресов хостов в сети от номера сессии и пр. Функция отклонения должна быть отражена в концепции уникальности системы. Зная ее, мы можем восстановить исходную функциональную структуру.

Что нам это дает? Исходное значение функции $f(\mathbf{par})$ является стереотипной схемой, относительно которой злоумышленник пытается произвести атаку. Если мы ее изменяем, то не даем со-

вершить стандартные действия и склоняем нарушителя к исследованию. Но исследование возможно только в том случае, если злоумышленник правильно определит гипотезу (дополнительные параметры) новой функции. В противном случае он не сможет получить информативной обратной связи от «черного ящика». Для случая одного дополнительного параметра $f(\mathbf{par}, p_1)$ зависимость получить достаточно просто, а следовательно, и сформулировать правильную гипотезу. Но параметров в процесс можно ввести бесконечно много – $f(\mathbf{par}, p_1, p_2, ..., p_n)$, тем самым мы делаем структуру системы все более сложной для ее исследования потенциальнм злоумышленником.

Защита от исследования веб-сервера. Усложняя процессы в информационных системах, необходимо точно осознавать возможную область для таких модификаций. Многие процессы, такие как стек протокола в операционной системе, принципы коммутации и пр., не поддаются усложнению (или по-другому можно сказать, что их усложнение чрезмерно затратное), но в целом в любые процессы можно вводить дополнительные параметры и формировать для них «концепцию уникальности». Эти идеи были заложены в разработку целого комплекса программных модулей для защиты от исследования систем (проект Reflexion Web, в настоящий момент резидент Инновационного центра Сколково). В области безопасности рациональнее закрывать от исследования самые популярные уязвимости, совершая атаку по которым, нарушитель не получал бы информативной обратной связи. Сценарий такого подхода достаточно прост – пытаясь реализовать простые уязвимости, злоумышленник не наблюдает «сопротивления» системы, а следовательно, тратит много времени на «распутывание» логики ее работы. Можно сказать, что он «вязнет» в системе, так как будучи не в состоянии правильно интерпретировать обратную связь, он не совершает действий в рамках поставленной цели. В это время приложение легко протоколирует несанкционированную активность, т.к. обнаруживает действия по стереотипным схемам.

Разработанные в настоящее время приложения для веб-сервера относятся к защите сервера от SQL-инъекций. Это наиболее популярный вид уязвимостей, а поэтому и наиболее оправданный с точки зрения защиты от исследования. Далее мы рассмотрим принципы работы этих двух модулей.

Первый пример достаточно прост в реализации, так как является функцией всего одного фиксированного параметра и не требует настройки «концепции уникальности». Принцип работы модуля схож с сетевым устройством HoneyPot. Как известно, данное устройство эмулирует несуществующую сеть как часть реальной и используется в качестве приманки для нарушителя. Оно сдерживает несанкционированную активность и позволяет изучить нарушителя на «безопасной территории». В рамках предложенного алгоритма устройство HoneyPot можно интерпретировать как защиту от исследования системы с отклонением по единственному параметру. В нашем случае мы аналогично создаем копию структуры базы данных, которая выдается за реальную при обнаружении несанкционированной активности.

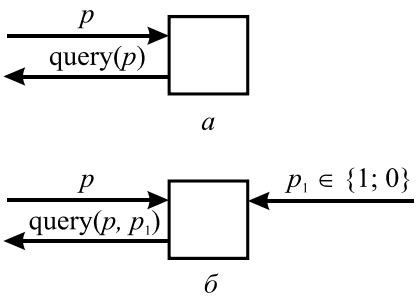

Рис. 2. Обращение к базе данных

Стандартный запрос к базе данных показан на рис. 2, *а*. Параметр p представляет собой запрос к базе данных, формируемый системой на основании информации, передаваемой через массивы GET и POST. Система управления базой данных обрабатывает запрос и выдает ответную реакцию – query(p).

Мы вводим в этот процесс дополнительный параметр – p_1 (рис. 2, *б*). Он может принимать всего два значения – 0 и 1. Значение единицы он принимает в том случае если регулярные выражения, проверяющие массивы GET и POST, обнаружили характерные символы для атак ISS и SQL-инъекций.

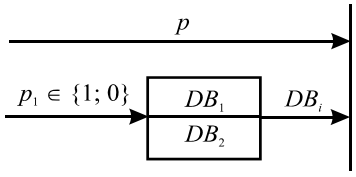

Рис. 3. Формирование запроса к базе данных

Формирование запроса к базе данных показано на рис. 3.

DB_1 – исходная (оригинальная) база данных сайта. На ее основе создается копия – DB_2, из которой можно убрать (подменить) нужную информацию. Кроме того, не представляет риска удаление и модификация информации в данной БД. Если по регулярным выражениям модуль опознает атаки ISS и SQL-инъекций, то параметр p_1 принимает значение 1 и система переподключается к базе данных DB_2, а также протоколирует данные массива

SERVER, содержащего IP-адрес злоумышленника, версию браузера, тип операционной системы и т.д.

Пытаясь реализовать атаку, злоумышленник не встречает сопротивления системы, а поэтому попадается на ту же самую приманку, что и в системе HoneyPot. Для исследования системы ему необходимо получить информативную обратную связь, а это в свою очередь возможно, если будет найдена функция $f(p_1)$ – функция работы регулярных выражений.

Данный модуль очень посредственно иллюстрирует метод защиты от исследования, так как не содержит концепции уникальности (точнее, она всегда постоянна и состоит из одного параметра), однако он очень прост в установке и не требует никаких дополнительных настроек. Далее мы рассмотрим расширенный вариант модуля работы с базой данных, позволяющего настраивать концепцию уникальности системы.

Модуль работы с базой данных. Поскольку подключаемые модули не могут влиять на структуру запроса к базе данных, то все, что мы можем сделать, – это увеличить многообразие того, на чем непосредственно строится запрос, т.е. структуру и содержание базы данных. Алгоритм защиты от исследования требует введения множества дискретных параметров $p_1 ... p_n$, относительно которых можно ввести функцию отклонения системы $f(p_1, \ ... \ p_n)$. Поскольку результатом должен быть запрос

$$\text{query}(p, f(p_1, \ \dots \ , p_n)) = \text{query}(p, DB_i),$$

то функция f есть отображение на множество баз данных:

$$f(p_1,...,p_n) \in \left\{DB_1, DB_2, DB_2^{\ 1}, DB_2^{\ 2},...,DB_2^{\ m}\right\}.$$

Принцип формирования запроса для такой функции показан на рис. 4.

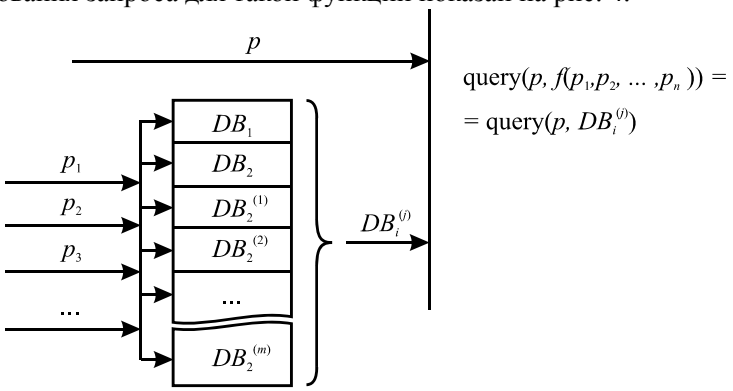

Рис. 4. Формирование запроса к базе данных

Запись баз данных с двумя индексами отражает принцип их формирования. DB_1 – исходная (оригинальная) база данных сайта. DB_2 – как и в предыдущем примере, копия ее структуры с пустыми таблицами. Если в концепции уникальности прописаны правила модификации структуры в зависимости от найденных параметров, то структура DB_2 меняется и формируется новое состояние $DB_2^{(j)}$.

Концепция уникальности позволяет администратору настраивать индивидуальную функциональную структуру системы. Она может быть выполнена в виде xml-файла или любого другого файла конфигураций. Структура концепции показана в таблице.

Концепция уникальности системы

Тип параметра	Параметр	Тип функции	Функция
1	DELETE	1	DB_2
2	UNION + LOAD_FILE	3	+1
3	USERS	2	*pass* = md5(*pass*)
…	…	…	…

Тип параметра: 1 – оператор языка SQL; 2 – пара операторов SQL; 3 – имена таблиц.

Тип функции: 1 – выбор базы данных; 2 – количество столбцов; 3 – значение столбцов.

Для примера из таблицы рассмотрим принцип работы модуля. Если в SQL-запросе встречается оператор DELETE, то происходит переадресация к базе данных DB_2. Вторая строка: если одновре-

менно встречаются операторы UNION и LOAD_FILE, то в таблицу вставляется дополнительный столбец. Это может быть необходимо для запрета загрузки файлов. Если в исходной таблице и в объединяемой количество столбцов не совпадает, то база данных выдает ошибку. И последняя строчка: если в тексте SQL-инъекции запрашивается таблица USERS, то над каждой ее строкой в базе данных DB_2 выполняется оператор UPDATE с указанной функцией. В данном случае для каждого пароля еще раз высчитывается хэш-свертка.

Такой подход позволяет администратору добавлять в работу сайта (запрос к базе данных) любые дополнительные параметры и строить, таким образом, уникальную с точки зрения защиты от исследования систему.

Работа системы. Работу модулей можно проиллюстрировать следующим образом. В поисках уязвимостей первое, что сделает злоумышленник, – попытается определить, есть ли на сайте проверка параметров, передаваемых через массивы GET и POST. Для этого он может ввести в запросе одинарную кавычку или конструкции “=1 and 1=1”, “=1 and 1=0” и т.п. Любые простые запросы система переадресует к исходной базе данных, и нарушитель может сделать вывод о возможности успешной атаки. Однако дальнейшие его исследования будут располагаться на множестве $\left\{DB_1, DB_2, DB_2{}^1, DB_2{}^2, ..., DB_2{}^m\right\}$ и вряд ли дадут ему сколько-нибудь значительную информативную обратную связь. Кроме того, модуль сразу запротоколирует источник несанкционированной активности (IP-адрес, версию браузера, тип операционной системы и т.д.).

Выводы. В данной работе был предложен новый метод построения защиты на основе технологии MTD с возможностью защиты от исследования не только информационной системы, но и самой схемы преобразования системы на основе MTD. Получен алгоритм проектирования таких информационных систем. Данный метод был реализован на примере защиты интернет-ресурсов от преднамеренных атак (программная система Reflexion Web). Был проведен эксперимент, включающий в себя попытки исследования защищенных веб-ресурсов с применением модулей Reflexion Web и без них, в том числе с применением ORM-модуля, описанного в [1], для защиты от исследования базы данных. Информация об используемой технологии защиты являлась открытой. ORM-модуль не обеспечил должной защиты от исследования, поскольку менял наименования сущностей базы данных только в определенные интервалы времени, в течение которых структура оставалась неизменной. Кроме того, он оказался неэффективен для исследования, где вместо наименований сущностей использовался их порядок в базе данных. Структура данных с использованием ORM-модуля была раскрыта в 2 случаях из 10. Структура базы данных с использованием модуля Reflexion Web не была получена ни в одном эксперименте.

В целом тестирование данных систем и анализ их работы дали положительные результаты. При использовании трех и более параметров в «концепции уникальности» злоумышленник тратил достаточно много времени на попытку рекогносцировки в системе и в конечном счете отказывался от дальнейшего исследования. При этом полученные сигнатуры поведения злоумышленника позволяли с 80%-ной точностью идентифицировать его при попытке атак на другие веб-серверы.

Литература

1. Jajodia S. Moving Target Defense. Creating Asymmetric Uncertainty for Cyber Threats / S. Jajodia, A.K. Ghosh, V. Swarup et al. – London: Springer, 2011. – 184 p.

2. Moving Target Defense II. Application of Game Theory and Adversarial Modeling / S. Jajodia, A.K. Ghosh, V. Swarup et al. – London: Springer, 2013. – 203 p.

3. Moving target defense (MTD) in an adaptive execution environment. / A. Paulos, P. Pal, R. Schantz, B. Benyo // ACM International Conference Proceeding Series. 8th Annual Cyber Security and Information Intelligence Research Workshop: Federal Cyber Security R and D Program Thrusts. – Oak Ridge: ACM Series, 2013. – P. 125–143.

4. Carvalho M. Moving-target defenses for computer networks. / M. Carvalho, R. Ford // IEEE Security and Privacy. – 2012. – Vol. 12, Iss. 2. – P. 73–76.

5. Li J.E. Address-space randomization for windows systems / J.E. Li, R. Sekar // Proceedings of 2006 Annual Computer Security Applications Conference (ACSAC). – Miami Beach: AC SAC, 2006. – P. 329–338.

6. Jafar Q.D. Openflow random host mutation: Transparent moving target defense using software-defined neyworking / Q.D. Jafar, E. Al-Shaer // Proceedings of the 1st Workshop on Hot Topics in Software Defined Networking (HotSDN). – Helsinki: ACM Series, 2012. – P. 127–132.

7. Styugin M. Protection against system research / M. Styugin // Cybernetics and Systems. – 2014. – Vol. 45, Iss. 4. – P. 362–372.

Стюгин Михаил Андреевич
Доцент каф. прикладной математики и компьютерной безопасности
Института космических и информационных технологий Сибирского федерального университета.
Тел.: 8 (391) 294-95-34
Эл. почта: styugin@gmail.com

Styugin M.A.

A new method of security development for web services based on moving target defense (MTD) technologies

This article describes a new method of security development for web services based on moving target defense (MTD) technologies. This method addresses the security of websites from attackers that employ SQL-injection, cross-site scripting, etc. The system interaction is built so that when the adversary tries to investigate the system structure, he obtains ever-increasing complexity of information from the system.
These technologies allow us to defend websites from users with malicious intent and to research the behavior of those attackers. Researching the behavior of such intruders affords us the opportunity to find new vulnerabilities. This paper provides the examples of these technologies.
Keywords: information security, moving target defense, SQL-injection, protection from research.
