МЕТОДЫ ЗАЩИТЫ ВЕБ-ПРИЛОЖЕНИЙ ОТ ЗЛОУМЫШЛЕННИКОВ

Боровков В.Е.[1], Ключарев П.Г.[2]

Цель статьи: *аналитический обзор методов защиты веб-приложений.*

Метод исследования: *анализ научных публикаций по теме статьи.*

Полученные результаты: *в обзорной статье проанализирована литература, посвященная защите веб-приложений от уязвимостей, а также такого программного обеспечения, как веб-бэкдоры, которые встраиваются злоумышленником для выполнения нелегитимных операций. Высокая угроза последних обусловлена тем, что они могут загружаться на веб-сервер через уязвимости, а также через другие доступные для злоумышленника пути. К тому же исходный код веб-бэкдора может быть различным. Все это усложняет процесс эффективного обнаружения. В статье приведена классификация методов защиты и дана их сравнительная характеристика. Особое внимание уделяется интеллектуальным методам защиты и проблемам, которые возникают при обучении моделей. Основными проблемами являются некачественные, неполные наборы данных, а также отсутствие проверки многими исследователями своих результатов в реальных условиях.*

Научная новизна *заключается в систематизации и достаточно обширном обзоре работ в области защиты веб-приложений от уязвимостей и бэкдоров, которые могут быть использованы злоумышленниками. Работа выявляет проблемы, связанные с этой областью, что подчеркивает важность и актуальность данной темы.*

Ключевые слова: *веб-уязвимости, веб-бэкдоры, веб-шеллы, машинное обучение.*

DOI:10.21681/2311-3456-2023-5-89-99

Введение

В настоящее время наблюдается глобальный процесс информатизации общества, который сопровождается ростом числа онлайн-сервисов и веб-приложений. В свою очередь, этот процесс также увеличивает количество уязвимостей, которые часто используют злоумышленники. Под злоумышленником мы будем понимать нарушителя, преднамеренно совершающего попытки выполнения запрещенных операций с данными, следствием чего может являться нарушение информационной безопасности (ИБ). По данным компании Positive Technologies[3] 17% от общего числа атак связаны с уязвимостями и недостатками защиты веб-приложений, которые могут быть использованы для проникновения в локальный сетевой периметр организации или распространения вредоносного программного обеспечения.

Для защиты веб-приложений используют различные методы и средства, начиная от ручного анализа логов специалистами и применения антивирусов до использования облачных файрволлов. Системы защиты, такие как «Security Information and Event Management» (SIEM), как правило, генерируют большие объемы данных, что затрудняет их анализ. Использование математических расчетов и научных достижений во многом ускорило процесс анализа данных и принятия необходимого решения. Одним из актуальных направлений развития стало использование в системах кибербезопасности искусственного интеллекта, а в частности такого подкласса, как глубокое машинное обучение. Его преимущества по сравнению с традиционными методами машинного обучения, такие как низкий коэффициент ошибок, непрерывное усовершенствование и оптимизация различных алгоритмов обучения и упрощения сетей позволяют автоматизировать процесс анализа данных, прогнозирования и классификации.

3 Уязвимости и угрозы веб-приложений в 2020-2021 г.г. // Positive Technologies [Электронный ресурс]. 2022. – URL: https://www.ptsecurity.com/ru-ru/research/ analytics/web-vulnerabilities-2020-2021/ (дата обращения: 20.09.2022)

1 Боровков Владислав Евгеньевич, аспирант кафедры «Информационная безопасность» МГТУ им Н.Э. Баумана, Москва, Россия. E-mail: vbscience@yandex.ru

2 Ключарёв Петр Георгиевич, доктор технических наук, доцент кафедры «Информационная безопасность», МГТУ им. Н.Э. Баумана, Москва, Россия. E-mail: pk.iu8@yandex.ru

В данной работе будут рассмотрены теоретические основы нелегитимных действий злоумышленника в сфере веб-приложений, а также представлен обзор современных исследований в области методов их защиты.

1. Веб-приложения

Необходимо отметить, что существует множество определений таких понятий, как «веб-приложение», «веб-сайт», «веб-ресурс». Зачастую некоторые нативные приложения (приложения, которые разработаны для использования на определённой платформе) определяются как веб-приложения и наоборот. Некоторые приложения могут использовать веб-технологии для связи, обработки данных, хранения, и тогда такие системы можно рассматривать как веб-приложения. В данном исследовании мы будем использовать определение, предложенное в работе [1]: «веб-приложение» — это система с компонентами на стороне клиента (компоненты клиента), которые взаимодействуют с компонентами на веб-сервере (компоненты сервера) для обработки данных. Они используют веб-службу, основанную на клиент-серверной архитектуре, модели «запрос-ответ», стандартном HTTP и других связанных с ним методах и технологиях.

В качестве основы для классификации веб-приложений используем тип клиентского компонента. Мы можем разделить веб-приложения на основе этого на две группы – приложения, использующие браузер, и приложения, которые его не используют.

Для работы браузерных приложений необходимо специальное программное обеспечение (ПО), называемое браузером. В нем происходит выполнение всех клиентских компонентов. В качестве примера могут служить различные сайты, которые загружаются посредством браузеров Yandex, Google и т.д.

Приложения, не использующие браузер, во многом похожи на обычные настольные приложения, но взаимодействуют с веб-серверами на основе HTTP-запросов. Этот метод не является широко используемым для построения полноценной архитектуры приложения из-за недостатков с управлением и обслуживанием [1]. В свою очередь, эти приложения являются составной частью некоторых программных решений, таких как гибридные веб-приложения, которые сочетают в себе функции нативных и веб-приложений. Благодаря такой комбинации они могут быть легко адаптированы и развернуты на различных программных платформах, таких как IOS, Android, Windows и т.д. [2].

Некоторым промежуточным вариантом выступают прогрессивные веб-приложения (Progressive Web Application – PWA). Это технология позволяет клиентам устанавливать сайт как мобильное приложение. Она также сочетает в себе нативные функции операционной системы и стратегии веб-разработки, является альтернативой другим подходам из-за дополнительных преимуществ, таких как автономность и фоновая синхронизация [3].

Как можно увидеть, главным составляющим каждого из подходов к веб-разработке является использование HTTP-запросов для взаимодействия между клиентом и сервером. Наличие недостатков в логике этого взаимодействия зачастую приводит к атакам на веб-серверы и использованию их в нелегитимных целях.

2. Использование злоумышленниками уязвимостей веб-приложений

Открытый проект обеспечения безопасности веб-приложений (Open Web Application Security Project – OWASP) публикует известный список угроз веб-приложений, ранжированных в порядке от одного до десяти. На момент написания данной работы актуальным является список OWASP Top 10 – 2021[4]:

A01:2021 – Нарушение контроля доступа (Broken Access Control).

A02:2021 – Ошибки в криптографии (Cryptographic Failures).

A03:2021 – Внедрение кода (Injection).

A04:2021 – Небезопасный дизайн (Insecure Design).

A05:2021 – Неправильная конфигурация (Security Misconfiguration).

A06:2021 – Уязвимые и устаревшие компоненты (Vulnerable and Outdated Components).

A07:2021 – Ошибки идентификации и аутентификации (Identification and Authentication Failures).

A08:2021 – Нарушение целостности данных и программного обеспечения (Software and Data Integrity Failures).

A09:2021 – Журнал безопасности и сбои мониторинга (Security Logging and Monitoring Failures).

A10:2021 – Подделка запросов со стороны сервера или же SSRF (Server-Side Request Forgery).

В отчете от компании Positive Technologies[5] представлен подробный анализ веб-уязвимостей, основан-

4 OWASP Top 10:2021 // OWASP [Электронный ресурс]. 2021. – URL: https://owasp.org/Top10/ (дата обращения 01.10.2022).

5 Web application vulnerabilities and threats: statistics for 2019 // Positive Technologies [Электронный ресурс]. 2020. – URL: https://www.ptsecurity.com/ww-en/analytics/web-vulnerabilities-2020/ (дата обращения 05.10.2022).

ный на предыдущем списке OWASP Top 10 – 2017. Каждая из представленных угроз приводит к той или иной степени компрометации веб-приложения, что может привести к утечке чувствительных данных или получению контроля над управлением веб-сервером. Также используя некоторые уязвимости (например, межсайтовый скриптинг XSS, который теперь входит в категорию A03:2021), злоумышленники могут проводить атаки на клиентов, организовывать фишинговые атаки, например, для получения учетных данных, и выполнять действия, выдавая себя за другого пользователя.

Сама категория «Внедрение кода» всегда являлась критически опасной. Это метод, используемый злоумышленниками для внедрения вредоносного кода в уязвимые веб-приложения. Атака происходит чаще всего, когда администратор не добавляет правила, ограничивающие использование определенных символов. Некоторые из наиболее распространенных инъекций — SQL, NoSQL, команда ОС, реляционное сопоставление объектов (ORM), LDAP и внедрение языка выражений (EL) или библиотеки навигации по графу объектов (OGNL). Концепция одинакова среди всех интерпретаторов[6].

В крупных хакерских атаках на организации веб-сервер часто не является конечной целью. В качестве злоумышленников в таком случае могут выступать APT-группировки (Advanced Persistent Threat). В таких атаках используются различные изощренные методы и приемы с целью кражи конфиденциальной информации. Одной из приоритетных задач таких группировок является получение быстрого нелегитимного доступа к атакуемой системе, и веб-сервера часто выступают в качестве первой входной точки во внутреннюю сеть организации.

3. Использование злоумышленниками веб-бэкдоров

Получив доступ к веб-серверу, злоумышленники могут встроить в него *бэкдор*. Бэкдор – это средство доступа к компьютерной системе или зашифрованным данным в обход обычных механизмов безопасности системы. В качестве компьютерной системы в контексте исследования выступает веб-сервер, поэтому мы будем использовать понятие *веб-бэкдор*. В качестве бэкдора зачастую на таких серверах могут использоваться *веб-шеллы*.

Веб-шеллом называют скрипт, целью которого является обеспечение постоянного доступа к зараженным серверам для выполнения злонамеренных действий. Злоумышленники часто загружают хорошо продуманные сценарии веб-шеллов с помощью SQL-инъекций, уязвимостей неограниченной загрузки файлов и атак с использованием межсайтовых сценариев [4]. Также необходимо отметить, что веб-шеллы могут быть установлены и другими способами, например, злоумышленники, подобрав пароли от SSH или FTP, встраивают веб-шеллы для создания резервных каналов доступа к атакуемой системе.

Как отмечают исследователи в работе [5] в зависимости от функции и размера языка сценариев веб-шеллы можно условно разделить на три категории:

1) Большой троян (Big Trojan). Он имеет большой размер и обладает комплексными функциями для выполнения команд, работы с сетями, проведения операций с базами данных и осуществления других вредоносных намерений. Кроме того, к таким веб-шеллам применен дружественный графический интерфейс.

2) Однословный троян (One Word Trojan). Это веб-шелл с одной строкой кода, его часто встраивают в обычные файлы или картинки из-за своей компактности. Он может выполнять множество функций, подобно большому веб-шеллу, при этом код, который нужно выполнить, вместе с командами зачастую передается через HTTP запрос, что увеличивает объём полезной нагрузки.

3) Маленький троян *(Small Trojan).* Он имеет небольшой размер и его легко скрыть, но обычно он имеет только функцию загрузки файлов, поэтому его часто называют загрузчиком. Поскольку большинство веб-сайтов имеют ограничения по размеру при загрузке файлов, злоумышленники обычно сначала устанавливают загрузчик, а затем загружают на веб-сайт полноценный веб-шелл для выполнения ключевых функций.

Один из примеров пользовательского интерфейса веб-шелла представлен на рис. 1.

Сложнее всего обнаружить веб-шеллы второго и третьего типа. При этом для однословных троянов зачастую могут разрабатываться целые приложения, которые генерируют полезную нагрузку, и они по своей функциональности ничем не уступают большим троянам. Таким является китайский инструмент для управления веб-шеллами AntSword[7]. Этот инструмент может работать с директориями и файлами системы, подключаться к базам данных, создавать терминал и

6 A03:2021 – Injection // OWASP [Электронный ресурс]. 2021. – URL: https://owasp.org/Top10/A03_2021-Injection/ (дата обращения 25.10.2022).

7 Github. AntSword // Github [Электронный ресурс]. 2022. – URL: https://github.com/AntSwordProject/antSword (дата обращения 20.11.2022).

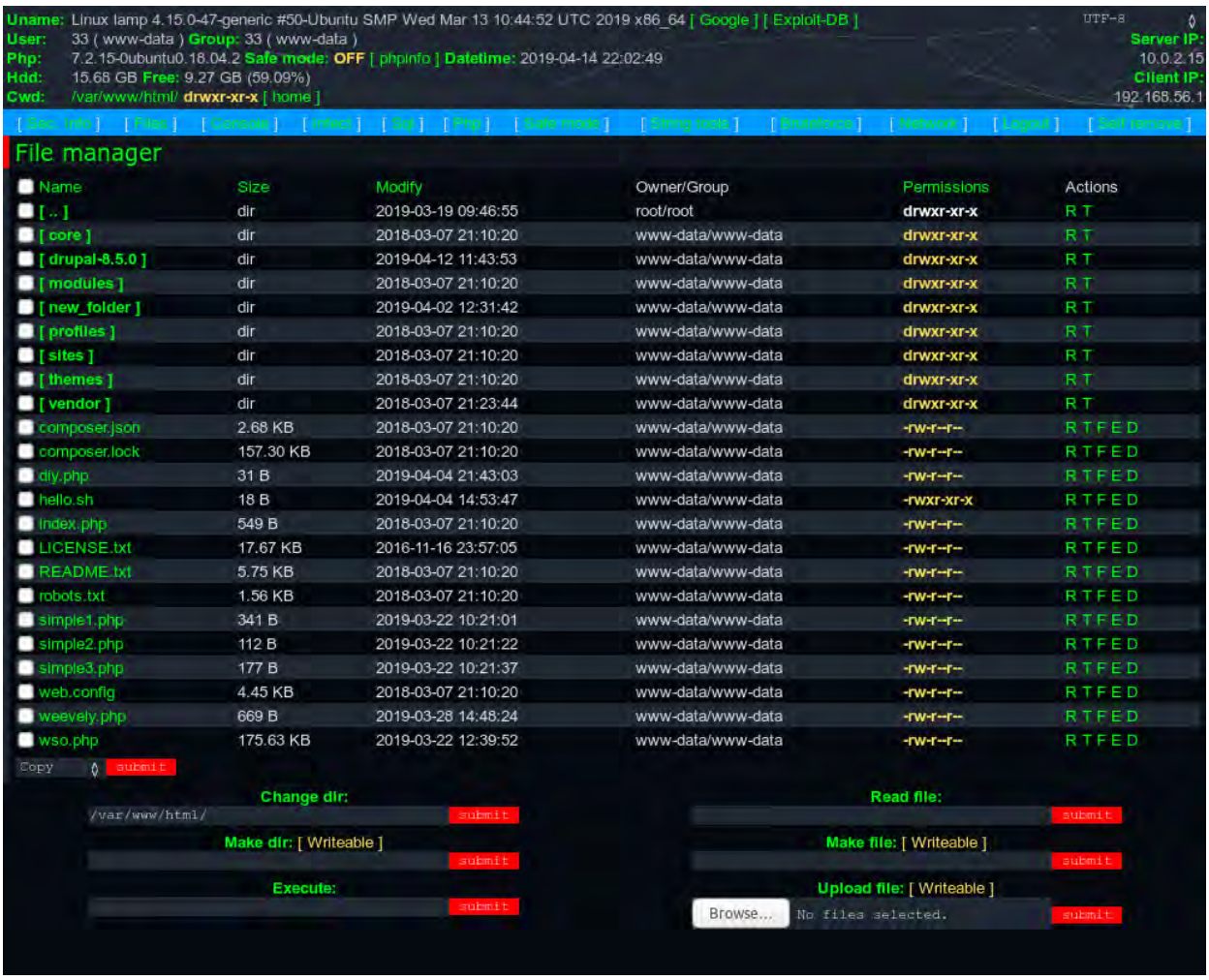

Рис.1. Пример пользовательского интерфейса большого трояна

использовать различные плагины, которые помогают обходить ограничения безопасности. Для его использования достаточно всего лишь одной строки кода на языке PHP – *eval(@$_POST['ant'])*.

Часто злоумышленники могут использовать веб-приложение для компрометации паролей пользователей, если оно включает авторизацию. Для этого они могут внедрить вредоносный код, называемый *стиллером* (от англ. *steal* – красть), который позволяет сохранять пароли пользователей в открытом виде. После получения паролей злоумышленники могут использовать их для авторизации в данном веб-приложении и других сервисах, что может привести к повышению привилегий злоумышленника.

Обнаружение уязвимостей веб-приложений и вредоносных средств, таких как веб-бэкдоры, может быть очень затруднено. Во ввиду этого методы защиты веб-приложений являются предметом изучения многих исследователей.

4. Методы защиты веб-приложений от веб-уязвимостей и веб-бэкдоров

Многие категории угроз могут быть обнаружены при использовании систем логирования и проверкой их администраторами. Однако проверка всех журналов безопасности может быть физически невозможна, особенно в случае, если администраторы должны обслуживать десятки или сотни серверов. В связи с этим для защиты веб-приложений используются различные средства, такие как межсетевые экраны веб-приложений, сканеры уязвимостей, системы обнаружения и предотвращения вторжений и т.д. В следующих разделах мы рассмотрим классификацию методов защиты веб-приложений и приведем обзор исследований в данной области.

4.1. Классификация методов защиты веб-приложений

Методы защиты веб-приложений могут быть классифицированы по нескольким факторам. Такие фак-

торы могут включать в себя тип защищаемого веб-приложения, класс атак, которые метод защиты обнаруживает или предотвращает, используемые техники и т.д. В статье [6] была предложена классификация методов защиты, достаточно полно охватывающая все современные классы атак, направленные на веб-ресурсы.

Для классификации методов защиты веб-приложений мы условно используем два независимых набора свойств: «*Анализируемые данные*» и «*Основание обнаружения*».

Методы защиты для анализа могут использовать:

1) Входные данные веб-приложения. Сюда входят значения полей HTTP-запросов.

2) Данные веб-приложения. Для анализа может использоваться исходный код веб-приложения, а также данные, сгенерированные веб-сервером и хранящиеся на нем.

3) Выходные данные веб-приложения. К ним относятся сгенерированные HTTP-ответы, их заголовки и поля.

По основаниям обнаружения методы можно разделить на следующие категории:

1) Метод на основе политик и правил. Предполагает анализ данных, выявление сигнатур и установление определенных правил.

2) Метод на основе намерений. Предполагает анализ и сравнение работы приложения с установленной заранее спецификацией веб-приложения.

3) Статистический метод. Предполагает использование вероятностного подхода к выявлению веб-атаки.

Данные классификации являются независимыми, и каждая из них может использоваться для описания метода защиты. Например, защита может контролировать входные данные веб-приложения и принимать решения на основе политик и правил. К таким средствам защиты можно отнести межсетевые экраны веб-приложений (Web application firewall – WAF), которые проверяют поступающий на приложение трафик HTTP/HTTPS, после чего принимают решения на основании заданных правил (блокировать, разрешить, отправить уведомление).

В следующем разделе приведем более подробное техническое описание классификаций и рассмотрим работы исследователей в области защиты веб-приложений от уязвимостей.

4.2. Методы защиты от веб-уязвимостей

Средства защиты веб-приложений зависят от данных, которые они использует для анализа, и могут быть размещены на веб-сервере или за его пределами. Объем доступных данных может отличаться в зависимости от выбранного варианта. При использовании веб-приложения как «черного ящика» у средства защиты есть возможность анализировать только входные и/или выходные данные из него, а само средство находится за пределами веб-сервера. Так работают сканеры веб-приложений такие как Acunetix, AppScan, BurpSuite, Arachni, W3af т.д. Также к таким средствам относятся облачные файрволлы, такие как Cloudflare.

Если используется метод «белого ящика», то защитное средство имеет возможность анализировать программный код и/или процессы, происходящие внутри сервера. Хотя данный метод позволяет выявлять более широкий круг проблем, так как обладает гораздо большим количеством информации, он менее универсален, чем метод «черного ящика».

Согласно вышеописанной классификации на основе «Анализируемых данных», можно установить, что методы, основанные на «черном ящике», используют только входные и выходные данные веб-приложений. Методы «белого ящика», кроме этого, могут также использовать данные самого веб-приложения.

В работе [7] исследователи анализируют различные методы тестирования программного обеспечения на каждом этапе жизненного цикла, которые связаны как с тестированием «черного», так и «белого ящика».

В недавней работе [8] исследователи провели сравнение генерации тестовых сценариев для REST API (Representational State Transfer Application Programming Interface) на основе «черного» и «белого ящиков» и показали, что комбинация обоих подходов дает наилучшие результаты в большинстве исследований с точки зрения поиска ошибок.

Выбор набора данных, необходимого и достаточного тому или иному методу защиты является одной из основных задач. Другой, не менее важной задачей, является правильная манипуляция этими данными, выбор техники и основания для обнаружения уязвимостей.

4.2.1 Методы на основе политик и правил

Метод на основе политик и правил является часто используемым методом защиты веб-приложений. Брандмауэры веб-приложений, или WAF, просматривают каждый запрос и/или ответ на различных уровнях обслуживания, таких как HTTP и HTTPS. Большинство существующих WAF основаны на правилах, ис-

пользующие регулярные выражения и сигнатуры, для обнаружения ключевых синтаксических конструкций, которые могут являться признаком проведения атаки.

Анализом сигнатурных методов защиты занимались многие исследователи. В работе [9] представлено исследование производительности трех систем обнаружения вторжений на основе сигнатур (Signature-based Intrusion Detection Systems – SIDS) – Snort, ModSecurity и Nemesida в контексте веб-атак. Результаты показали, что заранее заданные конфигурации этих систем не обеспечивают достаточно высокой производительности и способности обнаруживать известные атаки даже в самых чувствительных конфигурациях. Менее чувствительные конфигурации обеспечивали очень низкую скорость обнаружения, а наиболее чувствительные – неприемлемую точность и частоту срабатывания. Тем самым, исследователи поставили под сомнение вопрос о роли ненастроенных SIDS с открытым исходным кодом в качестве основных элементов защиты в контексте веб-служб.

Основным недостатком WAF на основе сигнатур является постоянная необходимость обновления их баз. Тем не менее, невозможно включить все сигнатуры в набор политик безопасности. Кроме того, наличие избыточного количества ненужных сигнатур также увеличивает вероятность блокировки законного трафика, повышая количество ложных срабатываний. При этом необходимо понимать, что они не защищают от атак «*нулевого дня*» (0-day). Обычно рекомендуемой контрмерой против таких кибератак является своевременное применение исправлений ПО, распространяемых поставщиком. Однако существует период от выявления уязвимости до выхода исправления.

В работе [10] авторы предлагают уменьшить воздействие атак «нулевого дня» на веб-приложения путем создания системы, которая собирает информацию об уязвимостях, связанных с веб-приложениями, в режиме реального времени в Интернете и генерирует сигнатуры брандмауэра веб-приложений. Для получения актуальной информации об уязвимостях система использует потоки данных, такие как социальные сети и веб-сайты для обсуждения технологий безопасности. В статье авторы использовали Twitter и базу данных уязвимостей NVD. После очистки собранных данных система проверяет наличие связанных уязвимых веб-приложений и создает для них сигнатуры WAF в виде виртуального исправления.

Более модернизированным методом является генерация обобщенного правила на основе шаблонов атак, что позволяет определять такое же количество атак, что и сотни сигнатур. В зависимости от среды приложения политика безопасности на основе сигнатур работает с более 2000–8000 сигнатур, в то время как политика безопасности на основе правил требует всего несколько десятков правил для обнаружения эквивалентного количества атак. Примером такого файрволла является Wapples[8].

В работе [11] был представлен гибридный метод, который использует обнаружение на основе сигнатур (signature-based detection – SBD) и аномалий (anomaly-based detection – ABD). Для обнаружения аномалий использовался байесовский классификатор. Запросы, классифицированные как ненормальные, помещаются в базу сигнатур. При повторном получении таких запросов они блокируется на этапе проверки сигнатур. Исследователями был сделан вывод, что сочетание обоих подходов будет более эффективным в процессе обнаружения и предотвращения веб-атак.

4.2.2 Методы на основе намерений

В отличие от предыдущего данный метод учитывает первоначальные намерения разработчика веб-приложения. Под «*намерением*» подразумевается функциональность, которая должна быть заложена в приложении с учетом целей и решаемых задач. Намерения определяют требования к приложению.

Данный метод использует информацию о том, какие процессы и операции заложены и разрешены разработчиком веб-приложения. Как правило, такая информация находится в технической документации на программный продукт. Также ее можно получить из анализа исходного кода приложения.

На первом этапе происходит обработка полученной информации и формирование «намерений» разработчика. Этот этап может существенно усложниться, если связи с разработчиком нет. В итоге должен получиться список «намерений», который указывает, как должно работать веб-приложение при штатных условиях.

На следующем этапе происходит мониторинг веб-приложения и выявление непредвиденных «намерений». К примеру, в HTTP-запросе было передано неожиданное количество аргументов, или запрос оказался неожиданно большим.

Данный метод является достаточно трудным в реализации, так как зачастую очень сложно учитывать все намерения разработчика. Как правило, данный метод используют для защиты некоторых аспектов

8 Wapples – The logical web application // PentaSecurity [Электронный ресурс]. 2022. – URL: https://www.pentasecurity.com/product/wapples (дата обращения 17.11.2022).

веб-приложения (например, потоков SQL-запросов) в сочетании с другими методами защиты.

4.2.3 Статистические методы

Ввиду сложности создания эффективной системы, основанной на сигнатурах, системы защиты, использующие статистические методы стали предметом множества научных исследований. Данные методы предполагают использование вероятностного подхода к обнаружению атак и направлены в первую очередь на обнаружение уязвимостей «нулевого дня».

Статистические подходы основаны на моделях. Это означает, что для данных создается модель или прототип, и объекты оцениваются с точки зрения того, насколько хорошо они вписываются в модель. Обнаружение атак строится на поиске «аномальных» объектов, которые не соответствуют модели и связаны с недостатками или редкими событиями.

В последние годы алгоритмы обнаружения аномалий на основе машинного обучения применяются для разнообразных задач. Особую популярность получили алгоритмы, использующие глубокое машинное обучение. В статье [12] авторами представлен широкий обзор методов глубокого обучения для задач по кибербезопасности. Они охватили широкий спектр типов атак, включая вредоносные программы, спам, инсайдерские угрозы, сетевые вторжения, ввод ложных данных и др.

В статье [13] исследователи проводят структурированный обзор методов исследования в области обнаружения аномалий на основе глубокого обучения, а также оценивают эффективность внедрения методов глубокого обучения в различные области применения. В частности они указывают, что варианты моделей глубокого обучения без учителя, основанные на глубокой нейронной сети и рекуррентной нейронной сети с памятью LSTM (Long Short-Term Memory), превосходят традиционные методы, такие как метод главных компонент (PCA), метод опорных векторов (SVM) и Isolation Forest в таких областях приложений, как здравоохранение и кибербезопасность.

В статье [14] авторами исследована эффективность WAF на основе машинного обучения. В некоторых приложениях оцениваемая система достигла точности 98,8% (во избежание недопониманий, под точностью (accuracy) будем понимать долю объектов, для которых был правильно предсказан класс. Она определяется формулой (1), где y и y^{pred} – настоящие и предсказанные метки классов соответственно, N – общее количество объектов, $\mathbb{I}\left[y_i = y_i^{pred}\right]$ возвращает единицу, если метки классов совпадают). Они показали, что методы, основанные на машинном обучении, имеют преимущества перед сигнатурными методами, так как могут предотвратить атаки нулевого дня, проще настраиваются и поддерживаются в актуальном состоянии.

$$Accuracy\left(y, y^{pred}\right) = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left[y_i = y_i^{pred}\right] \quad (1)$$

Все чаще исследователи показывают эффективность гибридных методов защиты. А. Текерек и др. в исследовании [15] продолжили усовершенствовать гибридную модель брандмауэра веб-приложений, используя SBD и ABD. Всего алгоритм включает три стадии обнаружения:

— 1 стадия – обнаружение известных типов атак по сигнатурам;
— 2 стадия – проверка через список HTTP-запросов, которые раньше были идентифицированы как аномальные;
— 3 стадия – проверка ABD и обновление списка аномальных HTTP-запросов.

Обнаружитель ABD реализуется с использованием искусственных нейронных сетей (Artificial Neural Networks – ANN). Предлагаемая модель была протестирована с использованием наборов данных WAF2015, CSIC2010 и ECML-PKDD. Согласно результатам теста, средний процент точности составлял 96,59 %.

Развитие машинного обучения также может отражаться на создании новых подходов к проведению атак. Авторами статьи [16] был представлен инструмент WAF-A-MoLE, который предназначен для обхода различных WAF, основанных на машинном обучении, путем поэтапного изменения полезной нагрузки – мутации кода. Сам инструмент использует состязательное машинное обучение (Adversarial Machine Learning – AML). Сутью AML является генерация таких входных данных, которые вводят в заблуждение модели машинного обучения, что приводит к неправильной классификации. Несмотря на то, что брандмауэры включали в себя различные алгоритмы машинного обучения, такие как рекуррентные нейронные сети, SVM, случайные леса и т.д., инструмент WAF-A-MoLE показал возможность их обхода.

Методы машинного обучения показывают свою эффективность, но при этом также имеют ряд проблем [17]:

1) *Недостаточный размер обучающих данных*. При обучении даже для решения простых задач необходимо большое количество данных, а для таких задач как

распознавание изображений или речи, понадобится миллионы образцов.

2) *Нерепрезентативные обучающие данные*. Для эффективного обучения необходимо использовать репрезентативные обучающие данные, которые могут быть обобщены на новые примеры.

3) *Данные плохого качества*. Чем больше в данных будет ошибок и шума, тем сложнее алгоритмам будет выявить закономерности в данных.

4) *Несущественные признаки*. Важно, чтобы обучающие данные имели как можно больше существенных признаков на входе.

5) *Переобучение обучающими данными*. При чрезмерном обучении может возникнуть случай, когда модель переобучается – хорошо выполняется на обучающих данных, но не обобщается при тестировании и использовании модели. Для ограничения модели с целью ее упрощения и снижения риска переобучения используют регуляризацию.

6) *Недообучение обучающими данными*. В данном случае для обучения используется слишком простая модель, чтобы узнать лежащую в основе структуру данных.

Кроме того, авторами статьи [18] была выявлена проблема, что модели машинного обучения, показывающие высокую эффективность в тестовой среде, при исследовании на реальном сетевом трафике демонстрируют низкую эффективность. Это говорит о плохих исходных данных, которые не вполне соответствуют реальности. Поэтому при разработке средства защиты успешные эксперименты должны проверяться в реальных условиях.

4.3. Методы защиты от веб-бэкдоров

Веб-бэкдоры могут быть загружены через уязвимости в веб-приложениях или слабую конфигурацию веб-сервера. Злоумышленник может встраивать веб-бэкдор в различные места, использовать обфускацию и шифрование, создавая уникальный код. Эти факторы затрудняют обнаружение веб-бэкдоров.

Подходы к обнаружению веб-бэкдоров могут существенно различаться и включать в себя анализ исходного кода, журналов безопасности и логирования, а также сетевого трафика.

С анализом исходного кода связано много исследований. В работе [4] авторами предложен метод обнаружен веб-шеллов на языке PHP, основанный на выделении признаков на различных уровнях представления (лексические, синтаксические и абстрактные). Затем данные признаки использовались в модели машинного обучения, основанной на методе опорных векторов, для обнаружения веб-шеллов. В работе [19] модель обнаружения основана на внутристрочной ассоциации слов и использовании Word2vec для векторного представления слов, которое затем используются в управляемом рекуррентном блоке (Gated Recurrent Units – GRU) для выполнения процесса обучения. Авторами работы [20] был предложен метод обнаружения с использованием модели сверточной нейронной сети при анализе кода на языках PHP, ASP, JSP.

Вместо того, чтобы производить анализ исходных кодов и содержимого пакетов HTTP-трафика, авторами работы [5] был предложен метод, который предполагает выделение и анализ сеансов, полученных из веб-журналов. Функции были извлечены из необработанных данных в веб-журналах, и для точного определения сеансов был применен статистический метод, основанный на временном интервале. Эксперимент показал, что модель на основе рекуррентной сети LSTM может достигать точности 95.97% при показатели полноты в 96,15%.

Значимых результатов достигли исследователи в работе [21]. Основная идея заключалась в том, что они производили анализ исходного кода PHP и выделяли 3 группы характеристик. Первая группа включала в себя статические признаки, такие как информационная энтропия, длина самого длинного слова и др. Вторая группа характеристик связана с выделением кода операций с использованием отладчика PHP PHPDBG. Третья группа связана с построением абстрактного синтаксического дерева [22] кода PHP для извлечения характеристик исполняемых данных с помощью PHP-Parser[9]. Эксперименты проводились на наборе данных, состоящих из 2917 образцов веб-шеллов. Результаты показали эффективность модели, достигнув точности обнаружения 99,66% без изучения оптимизации алгоритма машинного обучения. Идеи с выделением кода операций также использовали исследователи в работе [23].

Обобщенные результаты рассмотренных исследований представлены в табл. 1.

Главной трудностью, с которой сталкиваются исследователи, является наборы данных низкого качества. Веб-шеллы могут быть как отдельными файлами, так и встраиваться в легитимный программный код. Кроме того, они могут состоять из нескольких файлов, использовать различные поля и заголовки в пакетах

9 PHP Parser // Github [Электронный ресурс]. 2022. – URL: https://github.com/nikic/PHP-Parser (дата обращения 05.12.2022).

HTTP. Все это усложняет поиск и составление хорошего набора данных. Другой проблемой является то, что эксперименты не проводились на реальных системах, а результаты предоставлялись только на тех наборах, которые были извлечены исследователями. Так результаты, представленные в табл. 1, не могут являться объективными, так как эксперименты не были протестированы в реальных условиях. Наконец, многие исследования не являются универсальными, т.е. алгоритм для обнаружения веб-шелла, который создавался под конкретный язык программирования, неприменим или сложен в адаптации к другим языкам.

Таблица 1

Точность обнаружения веб-шеллов в различных исследованиях

Исследование	Язык программирования	Точность (accuracy)
[4]	PHP	92,18%
[19]	PHP, ASP, ASPX, JSP	99,19 %
[20]	PHP	99,5%
	JSP	97,5%
	ASP	98,3%
[5]	-	95.97%
[21]	PHP	99,66%
[23]	PHP	97,1%

5. Заключение

Современные веб-приложения – важная цель для злоумышленников. Через них они могут украсть конфиденциальную информацию и проникнуть в локальную сеть организации. «Закрепление» на веб-сервере является одним из методов сохранения доступа.

В статье рассматриваются основные угрозы, связанные с веб-приложениями, такие как уязвимости и использование злоумышленниками веб-бэкдоров. Уязвимости могут возникнуть из-за системных недостатков или ошибок при разработке приложения, и их эксплуатация может дать злоумышленникам первоначальный доступ к веб-серверу. Веб-бэкдор является следствием получения доступа, и встраивается злоумышленником для удобного выполнения несанкционированных действий. Чаще всего для таких целей злоумышленники использует веб-шеллы, а также веб-стиллеры для кражи учетных данных пользователей. Стоит отметить, что веб-бэкдоры могут быть загружены не только через уязвимости, но и через другие доступы к системе.

Методы защиты от угроз веб-приложений в последние время стали больше основываться на интеллектуальном анализе данных. Это связано с тем, что сигнатурные методы, несмотря на свою эффективность, не позволяют выявлять новые, заранее неизвестные, угрозы. К тому же злоумышленники часто производят обфускацию и шифрование для обхода таких методов защиты. Методы, основанные на машинном обучении, в свою очередь имеют свои проблемы, такие как низкое качество данных и их небольшой объем. К тому же многие исследователи в своих работах не проверяют эффективность методов на реальных системах, а ориентируются на результаты экспериментов, проводимых на выбранных наборах данных. Также отягчающими факторами являются трудная настройка системы и неуниверсальность методов. Ввиду этого исследования в данной области продолжают быть актуальными.

Литература

1. Dissanayake N., Dias. K. Web-based Applications: Extending the General Perspective of the Service of Web // 10th International Research Conference of KDU (KDU-IRC 2017) on Changing Dynamics in the Global Environment: Challenges and Opportunities, 2017.
2. Navyashree S., Rashmi R. Hybrid Web Application using Content Management System App used by all platforms // Advances in Computational Sciences and Technology. 2019. Vol. 12. P. 23-36.
3. Adetunji O., Ajaegbu C., Nzechukwu O. Dawning of Progressive Web Applications (PWA): Edging Out the Pitfalls of Traditional Mobile Development // American Scientific Research Journal for Engineering, Technology, and Sciences. 2020. Vol. 68(1). P. 85–99.
4. Zhu T., Weng Z., Fu L., Ruan L. A Web Shell Detection Method Based on Multiview Feature Fusion // Applied Sciences. 2020. Vol. 10(18). P. 1-16.
5. Wu Y., Sun Y., Huang C., Jia P., Liu L. Session-Based Webshell Detection Using Machine Learning in Web Logs // Security and Communication Networks. 2019. Vol. 2019. P. 1-11.
6. Лесько С. А. Модели и методы защиты веб–ресурсов: систематический обзор // CLOUD OF SCIENCE. 2020. № 3. С. 577-610.
7. Nidhra S. Black Box and White Box Testing Techniques - A Literature Review // International Journal of Embedded Systems and Applications. 2021. Vol. 2. P. 29-50.
8. Martin-Lopez A., Arcuri A., Segura S., Ruiz-Cortes A. Black-Box and White-Box Test Case Generation for RESTful APIs: Enemies or Allies? // Conference: International Symposium on Software Reliability Engineering. 2021. P.1-11.
9. Diaz-Verdejo J., Munoz-Calle J., Estepa A., Estepa R., Madinabeitia G. On the Detection Capabilities of Signature-Based Intrusion Detection Systems in the Context of Web Attacks // Applied Sciences. 2022. Vol. 12. P.1-16.
10. Kumazaki M., Yamaguchi Y., Shimada H., Hasegawa H. WAF Signature Generation with Real-Time Information on the Web // The

Fourteenth International Conference on Emerging Security Information, Systems and Technologies. 2020. P. 40-45.
11. Adem T., Cemal G., Omer F. Web tabanlı saldırı önleme sistemi tasarımı ve gerçekleştirilmesi: yeni bir hibrit model // Journal of the Faculty of Engineering and Architecture of Gazi University. 2016. Vol. 31(3). P. 645-653.
12. Berman D., Buczak A., Chavis J., Corbett C. A Survey of Deep Learning Methods for Cyber Security // Information (Switzerland). 2019. Vol. 10. P. 1-35.
13. Chawla S., Chalapathy R. Deep Learning for Anomaly Detection: A Survey // Computer Science. 2019. P.1-47.
14. Applebaum S., Gaber T., Ahmed A. Signature-based and Machine-Learning-based Web Application Firewalls: A Short Survey // Procedia Computer Science. 2021. Vol. 189. P. 359-367.
15. Tekerek A., Bay O. Design and Implementation of an Artificial Intelligence-Based Web Application Firewall Model // Neural Network World. 2019. Vol. 29. P. 189-206.
16. Demetrio L., Valenza A., Costa G., Lagorio G. WAF-A-MoLE: Evading Web Application Firewalls through Adversarial Machine Learning // Conference: 35th Annual ACM Symposium on Applied Computing. 2020. P. 1745-1752.
17. Жерон О., Прикладное машинное обучение с помощью Scikit-Learn, Keras и Tensorflow: концепции, инструменты и техники для создания интелектуальных систем, 2-е изд.: Пер. с англ. / О. Жерон. – СПб.: ООО «Диалектика», 2020. – 1040 с.
18. Горюнов, М.Н. Синтез модели машинного обучения для обнаружения компьютерных атак на основе набора данных CICIDS2017 / М.Н. Горюнов, А.Г. Мацкевич, Д.А. Рыболовлев // Труды Института системного программирования РАН. – 2020. – № 32(5). – С. 81-94.
19. Tingting L., Chunhui R., Yusheng F., Jie X., Jinhong G., Xinyu C. Webshell Detection Based on the Word Attention Mechanism // IEEE Access. Deep Learning: Security and Forensics Research Advances and Challenges. 2019. P. 185140 – 185147.
20. Zhuo-Hang L., Han-Bing Y., Rui M. Automatic and Accurate Detection of Webshell Based on Convolutional Neural Network // 15th International Annual Conference, CNCERT 2018. 2018. P. 73-85.
21. Pan Z., Chen Y., Chen Y., Shen Y., Guo X. Webshell Detection Based on Executable Data Characteristics of PHP Code // Wireless Communications and Mobile Computing. 2021. P.1-12.
22. Neamtiu I., Foster J., Hicks M. Understanding source code evolution using abstract syntax tree matching // in Proceedings of the 2005 international workshop on Mining software repositories - MSR '05. 2005. P. 1–5.
23. Fu J., Li L., Wang Y. Webshell Detection Based on Convolutional Neural Network // Journal of Zhengzhou University (Natural Science Edition). 2019. Vol 51(2). P. 1-8.

METHODS OF PROTECTING WEB APPLICATIONS FROM ATTACKER

Borovkov V.E.[10], Klyucharev P.G.[11]

The purpose of the article *is an analytical review of web application protection methods.*

Research method: *an analysis of scientific publications on the topic of the article.*

Results: *the review article analyzes the literature devoted to the protection of web applications from vulnerabilities, as well as software such as web backdoors that are embedded by an attacker to perform illegitimate operations. The high threat of the latter is due to the fact that they can be uploaded to the web server through vulnerabilities, as well as through other paths available to an attacker. In addition, the source code of the web backdoor may be different. All this complicates the process of effective detection. The article provides a classification of protection methods and their comparative characteristics. Special attention is paid to intellectual methods of protection and problems that arise when training models. The main problems are poor-quality, incomplete datasets, as well as the lack of verification by many researchers of their results in real conditions.*

The scientific novelty *lies in the systematization and a fairly extensive review of works in the field of protecting web applications from vulnerabilities and backdoors that can be used by attackers. The work identifies problems related to this area, which emphasizes the importance and relevance of this topic.*

Keywords: *web vulnerabilities, web backdoors, web shells, machine learning.*

10 Vladislav Borovkov, postgraduate student of Information Security department, Bauman Moscow State Technical University, Moscow, Russia. E-mail: vbscience@yandex.ru

11 Petr G. Klyucharev, Grand PhD, associate professor of Information Security department, Bauman Moscow State Technical University, Moscow, Russia. E-mail: pk.iu8@yandex.ru

References

1. Dissanayake N., Dias. K. Web-based Applications: Extending the General Perspective of the Service of Web // 10th International Research Conference of KDU (KDU-IRC 2017) on Changing Dynamics in the Global Environment: Challenges and Opportunities, 2017.
2. Navyashree S., Rashmi R. Hybrid Web Application using Content Management System App used by all platforms // Advances in Computational Sciences and Technology. 2019. Vol. 12. P. 23-36.
3. Adetunji O., Ajaegbu C., Nzechukwu O. Dawning of Progressive Web Applications (PWA): Edging Out the Pitfalls of Traditional Mobile Development // American Scientific Research Journal for Engineering, Technology, and Sciences. 2020. Vol. 68(1). P. 85–99.
4. Zhu T., Weng Z., Fu L., Ruan L. A Web Shell Detection Method Based on Multiview Feature Fusion // Applied Sciences. 2020. Vol. 10(18). P. 1-16.
5. Wu Y., Sun Y., Huang C., Jia P., Liu L. Session-Based Webshell Detection Using Machine Learning in Web Logs // Security and Communication Networks. 2019. Vol. 2019. P. 1-11.
6. Lesko S. A. Modeli i metody zashchity web–resursov: sistematicheskiy obzor // CLOUD OF SCIENCE. 2020. № 3. P. 577-610.
7. Nidhra S. Black Box and White Box Testing Techniques - A Literature Review // International Journal of Embedded Systems and Applications. 2021. Vol. 2. P. 29-50.
8. Martin-Lopez A., Arcuri A., Segura S., Ruiz-Cortes A. Black-Box and White-Box Test Case Generation for RESTful APIs: Enemies or Allies? // Conference: International Symposium on Software Reliability Engineering. 2021. P.1-11.
9. Diaz-Verdejo J., Munoz-Calle J., Estepa A., Estepa R., Madinabeitia G. On the Detection Capabilities of Signature-Based Intrusion Detection Systems in the Context of Web Attacks // Applied Sciences. 2022. Vol. 12. P.1-16.
10. Kumazaki M., Yamaguchi Y., Shimada H., Hasegawa H. WAF Signature Generation with Real-Time Information on the Web // The Fourteenth International Conference on Emerging Security Information, Systems and Technologies. 2020. P. 40-45.
11. Adem T., Cemal G., Omer F. Web tabanlı saldırı önleme sistemi tasarımı ve gerçekleştirilmesi: yeni bir hibrit model // Journal of the Faculty of Engineering and Architecture of Gazi University. 2016. Vol. 31(3). P. 645-653.
12. Berman D., Buczak A., Chavis J., Corbett C. A Survey of Deep Learning Methods for Cyber Security // Information (Switzerland). 2019. Vol. 10. P. 1-35.
13. Chawla S., Chalapathy R. Deep Learning for Anomaly Detection: A Survey // Computer Science. 2019. P.1-47.
14. Applebaum S., Gaber T., Ahmed A. Signature-based and Machine-Learning-based Web Application Firewalls: A Short Survey // Procedia Computer Science. 2021. Vol. 189. P. 359-367.
15. Tekerek A., Bay O. Design and Implementation of an Artificial Intelligence-Based Web Application Firewall Model // Neural Network World. 2019. Vol. 29. P. 189-206.
16. Demetrio L., Valenza A., Costa G., Lagorio G. WAF-A-MoLE: Evading Web Application Firewalls through Adversarial Machine Learning // Conference: 35th Annual ACM Symposium on Applied Computing. 2020. P. 1745-1752.
17. Zheron O. Prikladnoye mashinnoye obucheniye s pomoshchyu Scikit-Learn. Keras i Tensorflow: kontseptsii. instrumenty i tekhniki dlya sozdaniya intelektualnykh sistem. 2 izd.: Per. s angl. / O. Zheron. – SPb.: OOO «Dialektika». 2020. – P. 1040.
18. Goryunov. M.N. Sintez modeli mashinnogo obucheniya dlya obnaruzheniya kompyuternykh atak na osnove nabora dannykh CICIDS2017 / M.N. Goryunov. A.G. Matskevich. D.A. Rybolovlev // Trudy Instituta sistemnogo programmirovaniya RAN. – 2020. – № 32(5). – P. 81-94.
19. Tingting L., Chunhui R., Yusheng F., Jie X., Jinhong G., Xinyu C. Webshell Detection Based on the Word Attention Mechanism // IEEE Access. Deep Learning: Security and Forensics Research Advances and Challenges. 2019. P. 185140 – 185147.
20. Zhuo-Hang L., Han-Bing Y., Rui M. Automatic and Accurate Detection of Webshell Based on Convolutional Neural Network // 15th International Annual Conference, CNCERT 2018. 2018. P. 73-85.
21. Pan Z., Chen Y., Chen Y., Shen Y., Guo X. Webshell Detection Based on Executable Data Characteristics of PHP Code // Wireless Communications and Mobile Computing. 2021. P.1-12.
22. Neamtiu I., Foster J., Hicks M. Understanding source code evolution using abstract syntax tree matching // in Proceedings of the 2005 international workshop on Mining software repositories - MSR '05. 2005. P. 1–5.
23. Fu J., Li L., Wang Y. Webshell Detection Based on Convolutional Neural Network // Journal of Zhengzhou University (Natural Science Edition). 2019. Vol 51(2). P. 1-8.