Описание основных видов проверок безопасности в профиле защиты веб-приложения

Средства защиты WAF представляют собой набор фильтров, предназначенных для обнаружения атак на веб-сайты. Проверки безопасности используют эвристику, положительную модель безопасности и другие методы для обнаружения атак, которые невозможно обнаружить только с помощью сигнатур. Настраиваются проверки безопасности, путем создания и настройки профиля веб-приложения, который представляет собой набор определяемых пользователем параметров, сообщающих WAF, какие проверки безопасности использовать и как обрабатывать запрос или ответ, не прошедший проверку безопасности. Профиль связан с политикой безопасности для конкретного веб-сайта.

Start URL check

Проверяет URL-адреса во входящих запросах и блокирует попытку подключения, если URL-адрес не соответствует указанным критериям. Чтобы соответствовать критериям, URL-адрес должен совпадать с записью в списке начальных URL-адресов. Если включить этот параметр, пользователь, который нажимает ссылку на вашем веб-сайте, подключается к цели этой ссылки.

Deny URL check

Проверяет и блокирует подключения к URL-адресам, к которым обычно обращаются хакеры и вредоносный код. Эта проверка содержит список URL-адресов, которые являются обычными целями хакеров или вредоносного кода и которые редко, если вообще когда-либо появляются в легитимных запросах. Можно добавить URL-адреса или шаблоны URL-адресов в список. Проверка URL-адреса предотвращает атаки на различные уязвимости в системе безопасности, которые, как известно, существуют в программном обеспечении веб-сервера.

Cookie consistency check

Проверяет файлы cookie, возвращенные пользователями, чтобы убедиться, что они соответствуют файлам cookie, которые веб-сайт установил для этого пользователя. Если обнаружен измененный файл cookie, он удаляется из запроса до того, как запрос будет перенаправлен на веб-сервер. Также можно настроить проверку согласованности файлов cookie для преобразования всех файлов cookie сервера, которые он обрабатывает, путем шифрования файлов cookie, передачи файлов cookie через прокси или добавления флагов к файлам cookie. Эта проверка применяется к запросам и ответам.

Cookie hijacking protection

Предотвращает атаки хакеров, направленные на кражу файлов cookie. При атаке на систему безопасности злоумышленник захватывает сеанс пользователя, чтобы получить несанкционированный доступ к веб-приложению. Когда пользователь просматривает веб-сайт, например, банковское приложение, веб-сайт устанавливает сеанс с браузером. Во время сеанса приложение сохраняет данные пользователя, такие как учетные данные для входа, посещения страниц, в файле cookie. Затем файл cookie отправляется в клиентский браузер в ответ. Браузер сохраняет файлы cookie для поддержания активных сеансов. Злоумышленник может украсть эти файлы cookie либо вручную из хранилища файлов cookie браузера, либо с помощью какого-либо расширения браузера. Затем злоумышленник использует эти файлы cookie, чтобы получить доступ к сеансам веб-приложения пользователя.

Credit card check

Не позволяет злоумышленникам использовать недостатки предотвращения утечки данных для получения номеров кредитных карт. Блокируются все запросы, в которых содержится информация о номерах кредитных карт.

Buffer overflow check

Обнаруживает попытки вызвать переполнение буфера на веб-сервере. Если WAF обнаруживает, что URL-адрес, файлы cookie или заголовок длиннее настроенной длины, он блокирует запрос, поскольку это может привести к переполнению буфера.

File Upload Types

Многие злоумышленники пытаются загрузить вредоносный код, вирус или вредоносное ПО в виде вложенных файлов во время отправки нескольких форм. Чтобы предотвратить загрузку таких вредоносных файлов, администратор WAF может настроить набор допустимых форматов загрузки файлов в профиле WAF. Это ограничивает загрузку файлов с определенными форматами и защищает веб-сайт от загрузки вредоносных файлов и скриптов.

Form field consistency check

Проверяет веб-формы, возвращенные пользователями веб-сайта, и проверяет, не были ли веб-формы изменены клиентом ненадлежащим образом. Эта проверка применяется только к запросам HTML, которые содержат веб-форму с данными или без них. Проверка не относится к XML-запросам.

Проверка предотвращает несанкционированное внесение клиентами изменений в структуру веб-форм на веб-сайте при заполнении и отправке форм, гарантирует, что данные, отправляемые пользователем, соответствуют ограничениям HTML по длине и типу, а данные в скрытых полях не изменяются. Предотвращает подделку веб-формы злоумышленником и использование измененной формы для получения несанкционированного доступа к веб-сайту, перенаправления вывода контактной формы, использующей небезопасный сценарий, и, таким образом, отправки нежелательной массовой электронной почты или использования уязвимости в программном обеспечении веб-сервера. чтобы получить контроль над веб-сервером или базовой операционной системой.

Field formats check

Проверяет данные, которые пользователи отправляют на сайты в веб-формах, в том числе длину и тип данных, чтобы убедиться, что они подходят для поля формы, в котором они появляются. Если WAF обнаруживает недопустимые данные веб-формы в запросе пользователя, он блокирует запрос, не позволяя злоумышленнику отправлять недопустимые данные веб-формы на веб-сайт, проверка форматов полей предотвращает определенные типы атак на веб-сайт и серверы баз данных. Например, если в определенном поле предполагается, что пользователь введет номер телефона, проверка формата поля проверяет введенные пользователем данные, чтобы убедиться, что данные соответствуют формату номера телефона. Если для определенного поля требуется имя, проверка поля гарантирует, что тип и длина данных в этом поле соответствуют имени. Проверка выполняется для каждого поля формы, которое было настроено в профиле защиты.

CSRF form tagging check

Проверка помечает каждую веб-форму, отправляемую защищенным веб-сайтом пользователям, уникальным и непредсказуемым идентификатором формы, а затем

проверяет веб-формы, возвращаемые пользователями, чтобы убедиться, что предоставленный идентификатор формы верен. Эта проверка защищает от атак с подделкой межсайтовых запросов и применяется только к HTML-запросам, которые содержат веб-форму с данными или без них. Проверка не применяется к XML-запросам.

Проверка тегов форм CSRF не позволяет злоумышленникам использовать свои собственные веб-формы для отправки большого объема ответов с данными на защищаемые веб-сайты. Проверка требует относительно небольшой вычислительной мощности процессора по сравнению с некоторыми другими проверками безопасности, которые глубоко анализируют веб-формы.

HTML cross-site scripting check

Проверяет как заголовки, так и тела пользовательских POST-запросов на возможные атаки межсайтового скриптинга. Если проверка находит межсайтовый скрипт, она либо модифицирует (трансформирует) запрос, чтобы обезвредить атаку, либо блокирует запрос.

HTML SQL injection check

Многие веб-приложения имеют веб-формы, которые используют SQL для связи с серверами реляционных баз данных. Вредоносный код или хакер могут использовать небезопасную веб-форму для отправки команд SQL на веб-сервер. Проверка обеспечивает специальную защиту от внедрения неавторизованного кода SQL, который может нарушить безопасность. Если WAF обнаруживает неавторизованный код SQL в запросе пользователя, он либо преобразует запрос, чтобы сделать код SQL неактивным, либо блокирует запрос. WAF проверяет полезные данные запроса на внедренный код SQL в трех местах: 1) тело POST, 2) заголовки и 3) файлы cookie.

HTML command injection protection check

Проверяет, содержит ли входящий трафик несанкционированные команды, которые нарушают безопасность веб-сайта или изменяют ее. Если в трафике обнаруживаются какие-либо вредоносные команды, WAF блокирует запрос или выполняет преднастроенное действие.

XML format check

Проверяет формат XML входящих запросов и блокирует те запросы, которые неправильно сформированы или не соответствуют критериям спецификации XML для правильно сформированных документов XML. Пример некоторых критериев:

– XML-документ должен содержать только правильно закодированные символы Unicode, соответствующие спецификации Unicode;

– никакие специальные символы синтаксиса XML, такие как , и &, не могут быть включены в документ, за исключением случаев, когда они используются в XML-разметке;

– все начальные, конечные и пустые теги элементов должны быть правильно вложены друг в друга, ни один из них не должен отсутствовать или перекрываться;

– теги элементов XML чувствительны к регистру. Все начальные и конечные теги должны точно совпадать;

– один корневой элемент должен содержать все остальные элементы XML-документа.

XML Denial-of-Service check

Проверяет входящие XML-запросы, чтобы определить, соответствуют ли они характеристикам атаки типа «отказ в обслуживании» (DoS). Если есть совпадение,

блокирует эти запросы. Целью проверки XML DoS является предотвращение использования злоумышленником XML-запросов для запуска атаки типа «отказ в обслуживании» на веб-сервер или веб-сайт.

XML cross-site scripting check

Проверяет пользовательские запросы на возможные атаки с использованием межсайтовых сценариев в полезной нагрузке XML. Если обнаруживается возможная атака с использованием межсайтовых сценариев, такой запрос блокируется.

Чтобы предотвратить неправомерное использование сценариев в защищенных веб-службах для нарушения безопасности веб-служб, проверка межсайтовых сценариев XML блокирует сценарии, которые нарушают одно и то же правило происхождения, которое гласит, что сценарии не должны получать доступ или изменять содержимое на каком-либо сервере, кроме сервера, на котором они расположены. Любой сценарий, нарушающий одно и то же правило происхождения, называется межсайтовым сценарием, а практика использования сценариев для доступа или изменения содержимого на другом сервере называется межсайтовым сценарием. Причина, по которой межсайтовые сценарии представляют собой проблему безопасности, заключается в том, что веб-сервер, допускающий межсайтовые сценарии, может быть атакован сценарием, который находится не на этом веб-сервере, а на другом веб-сервере, например, на том, который принадлежит и контролируется злоумышленник.

XML SQL injection check

Исследует пользовательские запросы на предмет возможных атак XML SQL Injection. Если в запросе обнаруживается внедренный SQL-код в полезной нагрузке XML, такой запрос блокируется.

Атака XML SQL Injection может внедрить исходный код в веб-приложение таким образом, чтобы его можно было интерпретировать и запустить как допустимый SQL-запрос для выполнения операции с базой данных со злым умыслом. Например, атаки XML SQL Injection могут быть запущены для получения несанкционированного доступа к содержимому базы данных или для манипулирования сохраненными данными.

XML attachment check

Проверяет входящие запросы на наличие вредоносных вложений и блокирует те запросы, которые содержат вложения, которые могут нарушить безопасность приложений. Цель проверки XML-вложения – предотвратить использование злоумышленником XML-вложения для нарушения безопасности на веб-сервере или веб-сайте.

Web services interoperability check

Проверяет запросы и ответы на соответствие стандарту WS-I и блокирует те запросы и ответы, которые не соответствуют этому стандарту. Цель проверки WS-I – заблокировать запросы, которые могут не взаимодействовать с другими XML должным образом. Злоумышленник может использовать несоответствия в совместимости для запуска атаки на XML-приложение.

XML message validation check

Проверяет запросы, содержащие XML-сообщения, чтобы убедиться, что они допустимы. Если запрос содержит недопустимое XML-сообщение, WAF блокирует запрос. Целью проверки является предотвращение использования злоумышленником специально созданных недопустимых сообщений XML для нарушения безопасности веб-сайта.

XML SOAP fault filtering check

Проверяет ответы от защищенных веб-служб и отфильтровывает ошибки XML SOAP. Это предотвращает возможную утечку конфиденциальной информации.

JSON Denial-of-Service protection check

Проверяет входящий запрос JSON и проверяет наличие каких-либо данных, соответствующих характеристикам DoS-атаки. Если в запросе были нарушения JSON, WAF блокирует запрос, регистрирует данные, а также отображает страницу ошибки JSON. Цель проверки JSON DoS – предотвратить отправку злоумышленником запроса JSON для запуска DoS-атак на JSON-приложения или веб-сайт.

JSON Cross-Site Scripting protection check

Если входящие полезные данные JSON содержат вредоносные данные межсайтового скриптинга, WAF блокирует запрос.

JSON SQL Injection protection check

Входящий запрос JSON может иметь SQL-инъекцию в виде частичных строк SQL-запроса или неавторизованных команд в коде. Выполнение такого запроса может привести к краже данных из базы данных веб-сервера. При получении такого запроса WAF его блокирует.

JSON command injection protection check

Проверяет входящий трафик JSON на наличие неавторизованных команд, которые нарушают безопасность веб-сайта или изменяют его. При проверке трафика, если обнаруживаются какие-либо вредоносные команды, WAF блокирует запрос или выполняет настроенное действие.

Описание методов проверок в профиле ботнет защиты веб-приложения

Система управления ботами на WAF использует различные методы для обнаружения входящего трафика ботов. Для обнаружения ботов и определения из типов используются следующие методы (правила):

- **Bot white list.** Настраиваемый список IP-адресов (IPv4 и IPv6), подсетей (IPv4 и IPv6) и выражений политик, которые требуется обрабатывать как разрешенный список.
- **Bot black list**. Настраиваемый список IP-адресов (IPv4 и IPv6), подсетей (IPv4 и IPv6) и выражений политик, которым необходимо заблокировать доступ к веб-сайту и веб-приложениям.
- **IP reputation**. Репутационный список ботов. Определяет исходит ли входящий трафик бота с вредоносного IP-адреса или нет.
- **Device fingerprint**. Определяет, имеет ли входящий трафик бота идентификатор отпечатка устройства в заголовке входящего запроса и атрибуты браузера входящего трафика клиентского бота.
- **Bot log expression**. Метод обнаружения позволяет собирать дополнительную информацию в виде сообщений журнала. Данные могут быть именем пользователя, запросившего URL-адрес, исходным IP-адресом и исходным портом, с которого пользователь отправил запрос, или данными, сгенерированными из выражения.
- **Rate limit**. Это правило ограничивает количество запросов, поступающих от одного и того же клиента (IP-адреса бота).

– **Bot trap.** Обнаруживает и блокирует автоматических ботов, объявляя URL-ловушку в ответе клиента. URL-адрес отображается невидимым и недоступным, если клиент является пользователем-человеком. Метод обнаружения эффективен при блокировании атак автоматических ботов.

– **TPS**. Обнаруживает входящий трафик как ботов, если максимальное количество запросов и процентное увеличение запросов превышает настроенный временной интервал.

– **CAPTCHA**. Это правило использует CAPTCHA для предотвращения атак ботов. CAPTCHA – это проверка запроса-ответа, чтобы определить, исходит ли входящий трафик от пользователя-человека или от автоматизированного бота. Проверка помогает блокировать автоматических ботов, которые нарушают безопасность веб-приложений.