Литература

1. The Ten Most Critical Web Application Security Risks [Электронный ресурс]. – URL: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_en.pdf (дата обращения: 11.05.2018).

2. Евсеев Д. Введение в тему безопасности веб-приложений [Электронный ресурс]. – URL: https://www.ptsecurity.com/upload/corporate/ru-ru/webinars/ics/Д.Евтеев_Введение_в_тему_безоп_веб_прилож.pdf. (дата обращения: 11.05.2018).

КОНТРОЛЬ БЕЗОПАСНОСТИ В КЛИЕНТ-СЕРВЕРНЫХ ПРИЛОЖЕНИЯХ ПРИ ИСПОЛЬЗОВАНИИ ФРЕЙМВОРКА MICROSOFT SIGNALR НА ОСНОВЕ КЛИЕНТСКИХ ГРУПП

В.В. Кузнецов

При разработке клиент-серверных приложений в системах информационной безопасности острой ставится задача запросов клиента к серверу, особенно учитывая проблемы связанные с несанкционированным доступом в базу данных, отправку специфических аргументов серверу и других.

Целью настоящей работы явилась разработка подхода контроля безопасности запросов клиентов к серверу.

Фреймворк Microsoft SignalR [1], обеспечивающий двустороннее взаимодействие в клиент-серверных web-приложениях позволяет на стороне сервера (backend-side) применять атрибуты клиентских групп для классов, обеспечивающих API (Application programming interface) клиентов к серверу, в результате чего исключается необходимость проверять права клиента при обработке запросов. Клиентские группы, такие как например «Пользователи» и «Администраторы» хранятся на машине (ПК) выполняющего функции сервера в разделе «Управление компьютером/Локальные пользователи и группы/Группы». Добавив соответствующие группы, предоставляется возможность их использования в добавлении аттрибутов, например [Authorize(“Custom Administrators group”)], после чего доступ к соответствующему классу или методу получат только те пользователи, имена (домены) которых включены в соответствующие группы.

Таким образом, разработан подход по контролю безопасности клиент-серверных веб-приложений при использовании фреймворка Microsoft SignalR с технологией применений атрибутов для определенных клиентских групп, указанных на серверной машине с операционной системой Windows.

Литература

1. SignalR [Электронный ресурс]. – URL: https://docs.microsoft.com/en-us/aspnet/signalr (дата обращения: 16.05.2018).

УЯЗВИМОСТИ ПОВРЕЖДЕНИЯ ПАМЯТИ В УСТРОЙСТВАХ «INTERNET OF THINGS»

В.Ф. Кулиш

Устройства «интернета вещей» используют широкий диапазон архитекур центрального процессора, но наибольшее распространение получили архитектуры ARM и MIPS. Использование таких архитектур обусловлено их низким энергопотреблением и, соответсвенно, низким количеством выделяемого тепла при работе. Однако использование таких архитектур не защищает от уязвимостей повреждения памяти. Также как и устройства с архитектурой x86, такие устройства также подвержены уязвимостям переполнения буфера и уязвимостям форматных строк.

Уязвимости переполнения буфера обнаружились еще в начале компьютерной эпохи и продолжают существовать по сей день. Уязвимостям такого типа подвержены языки программирования, в которых управление процессом выделения памяти отдано на откуп программисту (пример: C, C++). При создании программ разработчику необходимо контролировать размер помещаемых в переменную данных, память под которую была

выделена ранее специальной командой языка программирования. В случае, если размер данных будет больше, чем количество выделенной памяти произойдет переполнение буфера. В результате переполнения могут быть повреждены важные данные программы, что может вызвать падение программы. Однако перезапись данных может привести не только к краху, но и к перехвату управления над программой и выполнению произвольного кода. В зависимисти от места выделения памяти переполнение может произойти на стеке вызовов функций или в динамически выделяемой области памяти (куча).

Уязвимости форматной строки основаны на ошибках программирования, влияние которых на безопасности не всегда очевидно. Форматные строки используются для вывода определенного вида информации. В ней предварительно указывается какой тип данных предполагается выводить. Отсуствие формата позволяет злоумышленнику внедрить свой формат предварительно передав на вход программы исполняемый код. При выводе форматной строки код будет выполнен и управление будет передано злоумышленнику.

Методы противодействия атакам на повреждение памяти направлены на уменьшение последствий от атаки. Методы никак не защищают программу от падения, но позволяют предотвратить перехват управления программой с помощью специальных данных. Существуют два основных способа предотвращения: NX-бит; ALSR.

NX-бит, отвечающий за No-eXecute, – это технология, используемая в CPU для выделения пространства памяти для хранения кода процессора или данных. Однако, NX bit все больше и больше используется в архитектуре процессора фон Неймана, из-за соображений безопасности. Операционная система с помощью NX бит может точно размечать пространства памяти как не исполнимые. Процессор будет отвергать выполнение любого кода, расположенного в этих зона памяти. Общая название техники известна как executable space protection, и используется для предотвращения исполнения вредоносного программного обеспечения, которые могут «захватить» компьютер вставляя свой код в код других программ, и запуская свой код в этом разделе; один из классов таких атак известный как переполнение буфера. Однако злоумышленник может обойти защиту с помощью NX бит путем переиспользования кода программы или библиотек, используемых ей. Технологии получили название «return-to-libc» и возвратно-ориентированное программирование. Они позволяют избавиться исполняемого кода передаваемого на вход программе. Вместо этого атакующий использует для атаки функции уже имеющиеся в программе.

ASLR (англ. address space layout randomization – «рандомизация размещения адресного пространства») – технология, применяемая в операционных системах, при использовании которой случайным образом изменяется расположение в адресном пространстве процесса важных структур данных, а именно образов исполняемого файла, подгружаемых библиотек, кучи и стека. Технология ASLR позволяет размещать системные компоненты в разных областях памяти при каждом запуске операционной системы. Задача этой технологии – внесение случайности в распределение адресов оперативной памяти, используемых приложением. Использование ASLR изменяет расположение в адресном пространстве таких важных структур как образ исполняемого файла, подгружаемых библиотек, кучи и прочего. Использование ASLR приводит к тому, что целый класс атак (к примеру, переполнение буфера, возврат в библиотеку (return-to-libc)) заканчивается неудачей. Помимо этого технология позволяет обнаруживать подобные атаки, так как в результате неудачных попыток исполнение атакуемого приложения прекращается.

Уязвимостям повреждения памяти подвержены процессоры на разных архитектурах. Различаются только методы эксплуатации на разных архитектурах. Методы противодействия атакам, использующим уязвимости повреждения памяти, также используются одни и те же на разных архитектурах. Однако методы противодействия не всегда достаточны для обеспечения защищенности программы.