ЗАЩИТА ИНФОРМАЦИИ В ПРОЕКТАХ СРЕДЫ РАЗРАБОТКИ ПРИЛОЖЕНИЙ Unity

И. В. Непран

Одесский национальный политехнический университет
Украина, г. Одесса
Ivangopro@ya.ru

Предлагаются способы защиты информации в проектах среды разработки приложений Unity, которые позволяют существенно снизить вероятность несанкционированного доступа к защищенным областям проекта, а именно к временным данным, хранящимся в оперативном запоминающем устройстве, и к данным, хранящимся в постоянном запоминающем устройстве.

Ключевые слова: защита данных, шифрование данных, среда разработки приложений Unity.

В среде разработки приложений Unity присутствует множество встроенных способов защиты и шифрования подобной информации, однако из-за большой популярности платформы стандартные элементы защиты быстро взламываются, и мошенники в кратчайшие сроки получают доступ к закрытым данным. Во избежание несанкционированного доступа к закрытой информации приложений необходима разработка новых способов ее защиты, что и будет рассмотрено в рамках данной работы.

Рассмотрим шифрование встроенного хранилища данных PlayerPrefs с использованием хеш-фунций [1]. Во избежание несанкционированного доступа мошенников к данным раздела PlayerPrefs эти данные могут быть зашифрованы и проверены на предмет, проводилось ли их несанкционированное редактирование. Можно предложить достаточно простой и эффективный способ шифрования, заключающийся в том, что после сохранения данных в раздел PlayerPrefs вычисляется хеш-сумма раздела с добавлением секретного ключа и полученное значение суммы записывается в отдельную строку PlayerPrefs. Таким образом, при получении данных из PlayerPrefs и секретного ключа приложение сможет сравнить записанную хеш-сумму с реальной, вычислив ее заново, и если эти суммы отличаются, раздел PlayerPrefs может быть обнулен. Использование предложенного способа шифрования делает редактирование раздела PlayerPrefs бесполезным.

Теперь рассмотрим защиту динамических данных приложений, которые хранятся в оперативной памяти. Самая большая проблема большинства приложений заключается в том, что эти данные хранятся в оперативной памяти в незашифрованном виде, что и позволяет мошенникам без особых затруднений находить эти значения в оперативной памяти и редактировать их прямо там. Для исключения несанкционированного доступа необходимо изменить подход хранения данных в оперативной памяти. В результате проведенных исследований для работы с оперативной памятью было решено разработать специальный класс. Его суть заключается в том, что перед записью в оперативную память данные необходимо исказить. В предлагаемом способе к числовым данным перед записью в оперативную память добавляется случайно генерируемое число, которое хранится в другой ячейке памяти. При необходимости доступа к этим данным это число просто отнимается от исходных числовых данных, давая в результате реальное значение, используемое в приложении. Такой достаточно простой способ значительно усложняет поиск ячеек памяти, которые хранят важные данные приложения в оперативной памяти, а следовательно усложняет и их редактирование, обеспечивая защиту данных. Даже если в приложении данные выводятся на экран, мошенники не смогут получить доступ к действительным значениям этих переменных и, соответственно, не смогут повлиять на игровой процесс.

Подобный способ искажения данных может быть применен и для текстовых данных. К примеру, можно создать таблицу символов, в которой каждому символу будет соответствовать какое-то число. После генерирования случайного числа для добавления к реальным данным приложения с помощью таблицы возможно получение некого набора символов, соответствующего случайному числу, а реальные текстовые данные, наоборот, можно перевести в числовое значение по этой же таблице. В процессе доступа к этим символьным данным простая обратная расшифровка с помощью таблицы символов позволяет получить действительные текстовые значения, которые изначально были зашифрованы.

Кроме этого также существует необходимость шифрования игровых ресурсов проектов Unity, особенно при их передаче через сеть интернет. Для этой цели существует встроенный способ шифрования, однако даже сами разработчики платформы рекомендуют использовать собственные способы шифрования. Одним из них является использование типа TextAsset для хранения данных в байтах [2]. Данный подход подразумевает сохранение нужных вам файлов с расширением .bytes, такие файлы будут восприниматься Unity как файл типа TextAsset и будут включены в ваши AssetBundles, а после этого размещены на сервере. Таким образом, шифруется не весь AssetBundles, а только отдельные файлы TextAsset.

Второй подход, заключается в шифровании целых AssetBundles, также сохраняя их с расширением .bytes, однако такое шифрование приводит к тому, что такие AssetBundles не будут кешироваться, поэтому есть смысл сохранять зашифрованные AssetBundles в один обычный AssetBundle. Файлы, зашифрованные таким способом, после скачивания в редактор Unity можно декодировать с помощью метода AssetBundle.CreateFromMemory.

В результате проведенного анализа были разработаны практические способы шифрования данных приложений и проектов Unity, которые являются более эффективными, чем общеизвестные или встроенные в среду разработки способы. В ходе проведенных тестов была подтверждена работоспособность и эффективность описанных подходов. При использовании предложенных способов шифрования данных осуществляется простая и эффективная защита «цифровой собственности», включая внутриигровой контент игровых приложений, что позволяет избежать значительных финансовых потерь при использовании финансовой модели покупок внутри приложений. Дальнейшие исследования планируется направить на разработку новых способов защиты файлов приложений среды Unity, в частности способов защиты двухмерных объектов и шифрования электронных адресов серверов обновлений приложений.

ИСПОЛЬЗОВАННЫЕ ИСТОЧНИКИ

1. Джозеф Хокинг. Unity в действии. Мультиплатформенная разработка на C#.— Санкт-Петербург: Питер, 2016.

2. Хорхе Паласиос. Unity 5.x. Программирования искусственного интеллекта в играх.— Москва: ДМК Пресс, 2016.

———

I. V. Nepran

Information protection in application development framework Unity projects

The proposed methods of information protection in application development environment Unity projects can significantly reduce the possibility of unauthorized access to the protected project data, such as the temporary data stored in random access memory and the data stored in read only memory.

Keywords: data protection, data encryption, application development environment Unity.

———