УДК004.056.5
А.А.Ящук
Луцький національний технічний університет

МЕТОДИ ЗАХИСТУ КОДУ .NET-ПРОГРАМ

Ящук А.А Методи захисту коду Net-програм. .У статті розглядаються актуальні методи захисту коду програм, що виконуються в середовищі .NetFramework.
Ключові слова: NetFramework.

Ящук А.А. Методы защиты кода Net-программ. В статье рассматриваются актуальные методы защиты кода программ, выполняемых в среде .Net Framework.
Ключевые слова: NetFramework.

Yaschuk A.A.The methods of Net code protection. The actual methods of protection of code of .Net Framework programs areconsidered in the article.
Keywords: NetFramework.

Постановка проблеми. Код, написаний будь-якою .NET мовою (C#, VisualBasic, F#та ін.) компілюється в IntermediateLanguage, який зазвичай називають IL або MSIL. Крім того, в програмі на .NET активно використовуються метадані, тобто вся інформація про класи, методи, властивості, атрибути і все інше, зберігається у виконуваному файлі[1].Зі скомпільованих збірок (exe і dll) для платформи .NET код може бути легко відновлений на мовах високого рівня (C#, VB.net) [2]. Якщо в програмі є система ліцензування, то вона може бути легко знята, крім того вихідний код практично незахищений від копіювання і використання сторонніми особами. У випадку розробки комерційних програмних продуктів, призначених для роботи в середовищі .NetFramework актуальною є проблема захисту.

Мета.Дослідити сучасні методи і засоби захисту коду програм .NET Framework від несанкціонованого використання і модифікації.

Виклад основного матеріалу.Середнайбільш поширених на даний момент методами захисту програмних продуктів можна виділити [3]:

- виконання на стороні сервера (напр. веб-додатки ASP.NET).Даний метод захисту ґрунтується на технології клієнт-сервер, він дозволяє запобігти відсиланню коду програми користувачам, які будуть з нею працювати, так як сама програма зберігатися, і виконується на сервері, а користувачі, використовуючи клієнтську частину цієї програми, отримують результати її виконання (рис. 1).

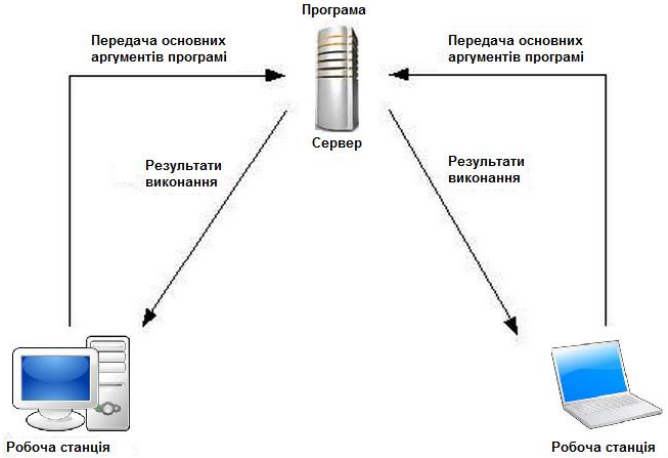


Рис.1. Виконання програми на стороні сервера

.NET-додатки, що призначені для виконання на ПКможна захищати двома основними методами: пакування і обфускація [2].

Пакування – модифікація збірки таким чином, що на виході одержується WinAPI додаток, що містить в собі модуль для розпаковування і запакований IL-код.

Захист пакуванням має наступні недоліки:

- втрата кросплатформенності;

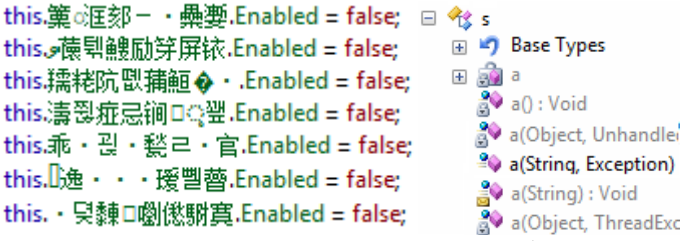
- зниження продуктивності. Затрачаються додаткові ресурси системи на розпаковування коду під час виконання;
- ненадійність методу. Для нормального виконання в середовищі .Net в будь-якому випадку потрібен доступ до оригінального IL-байткоду, тому отримати цей байткод теж реально.

Зважаючи на недоліки попереднього методу особливий інтерес представляє обфускація.

Суть процесу обфускації полягає в тому, щоб заплутати програмний код і усунути більшість логічних зв'язків у ньому, тобто трансформувати його так, щоб він був дуже важкий для вивчення і модифікації сторонніми особами.

Для реалізації обфускації .NET програм використовують ряд принципів:

перейменування класів, методів та інших елементів. Видаляються всі «підказки», які може використовувати зловмисник для швидкого пошуку класів, що відповідають, наприклад, за ліцензування. При «крадіжці» коду буде дуже важко розібратися в логіці програми, для чого і як створюються класи, і викликаються методи. Популярним варіантом є перейменування в недруковані символи, китайські ієрогліфи (рис.2, а).Аналогічний варіант – використання коротких, але друкованих ідентифікаторів (a, b, c, ... aa, ab, ac ...), (рис.2, б).



а б

Рис.2. Обфускація, шляхом перейменування

Ще одним варіантом перейменування є – використання ключових слів мови високого рівня (C# або VB.net), або невалідність ідентифікаторів для цієї мови (наприклад ?123?) – цей варіант нічим не кращий двох попередніх, але вважається, що при «крадіжці» коду не скористаються деобфускатором, і на виході вийде некомпільований текст.

- об'єднання збірок і просторів імен (Namespace Flatten, Assembly Merge). Зі збільшенням кількості класів у збірці ускладнюється розуміння коду програми. Чим більше класів міститиме результуюча збірка, тим складніше без детального аналізу буде в ній знайти потрібну інформацію.Простори імен (namespase)зазвичай об'єднуються під час обфускації імен класів (щоб не було колізій з класами з однаковими іменами з різних просторів імен).
- створення великої кількості overload-методів з одним ім'ям, які мали до обфускації різні імена, і ніяк не були пов'язані. Також .net дозволяє створювати override-методи, імена яких відрізняються від імен методів, які вони перекрили. Це збиває з пантелику не лише зловмисників, а й додає зайві вимоги до деобфускатора.
- зміна вмісту класів Деякі обфускатори можуть об'єднувати декілька класів в один, або робити зі звичайного класу вкладений. Така обфускація часто призводить до помилок в результуючій програмі, і використовується рідко.
- обфускація control flow. На цьому етапі змінюється порядок інструкцій в коді і змінюються самі інструкції. Дана методика дозволяє ввести в оману більшість декомпіляторів мов високого рівня, що дуже добре протидіє «крадіжкам» коду, заплутує кракерів і авторів генераторів ключів. Недоліком є іноді знижена продуктивність, оскільки чим більше ми заплутуємо хід виконання програми, тим довше вона виконується.У більшості випадків код методу розбивається на блоки, ці блоки перемішуються у випадковому порядку і «склеюються» за допомогою безумовних переходів (інструкції br і br.s).

Деякі обфускатори замінюють інструкції переходу (як оригінальні, так і вставлені) на завантаження константи і перехід на switch. Також можливий варіант:

```
if (5 <(3 - 6)) { // IL-сміття, або неправильний код }
```

- Invalid IL.В ділянки коду, які ніколи не будуть виконані, вставляються не описані в стандарті коди операцій (тобто невалідні інструкції). В рефлекторі буде відображено приблизно наступне:

```
[STAThread]
Private static void Main()
{
        // Invalidmethodbody.
}
```

Дана методика не є складною для обходу.

- приховування рядків. Зазвичай це робиться якимось простим алгоритмом шифрування типу XOR на константу:

```
public static string decode(string str, int num)
{
   int length = str.Length;
   char[] chArray = str.ToCharArray();
   while (--length >= 0)
      chArray[length] = (char)(chArray[length] ^ num);
   return new string(chArray);
}
```

Можливе об'єднання рядків в один з викликом методу Substring(), приховування рядків в ресурси. У кожному випадку «шифрування» представлено у вигляді статичного методу з кількома аргументами, зазвичай це рядок і / або число. Криптографічі алгоритми не застосовується, оскільки справжнє шифрування, суттєво знизить продуктивність програми.

- використання специфічних атрибутів і багів декомпіляторів. Найбільш часто зустрічається атрибут [SuppressIldasm], який не дозволяє працювати на даній збірці з офіційним декомпілятором Microsoft - ildasm.В якості багів можна зустріти чисто технічні недоробки декомпілятора.

- інші методи Іноді можна зустріти дуже схожий на приховання рядків підхід, але для ресурсів.Також використовується конвертування manged в unmanaged .net-коду. Тобто все перезбирають з позначками unmanaged. Практично вся функціональність в межах домена зберігається, але рефлектором код вже не подивитися.

На сьогоднішній день існує велика кількість як безкоштовних так і комерційних обфускаторів, що надають різний ступінь захисту програмних продуктів.

Dotfuscator (рис.3)–аналізує програми та робить їх меншими, швидшими і складнішими для декомпіляції [4]. До методів обфускації, що використовуються в Dotfuscator включають перейменування (заміна значущих ідентифікаторів з короткими безглуздими назвами), створення overload-методів з одним ім'ям; зміна потоку керування, шифрування строкових літералів.Dotfuscator включено до пакету VisualStudio.

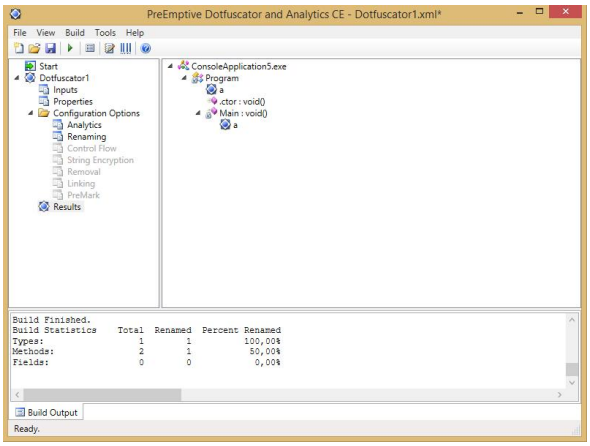


Рис.3. Прогама Dotfuscator

.Net Reactor (рис.4)– обфускатор, який надає можливості не тільки захисту коду за допомогою обфускації, але і вбудовування в додаток механізмів перевірки ліцензій і реєстраційних ключів [5]. З його допомогою можна обмежити час або кількість запусків пробної версії, додати для продукту нескінченні або обмежені за часом ліцензії та багато іншого.

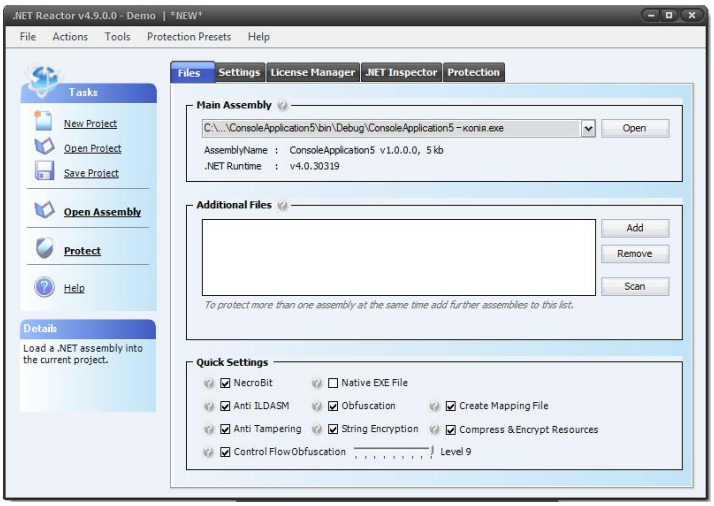


Рис.4. Прогама .Net Reactor

DotNetProtector(рис.5) – система захисту коду .NET, яка заважає декомпіляції збірок [6]. Може застосовуватися для захисту в додатках Windows, бібліотеках DLL ASP і для захисту SQL збірок.

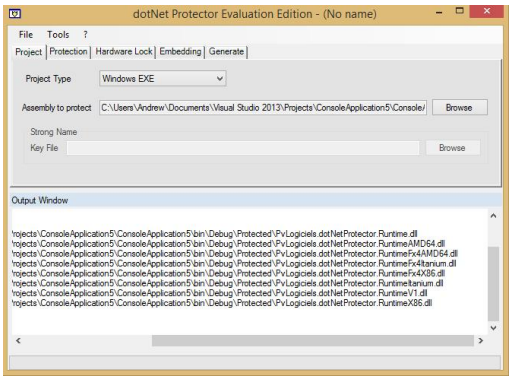


Рис.5. Програма DotNet Protector.

Для тестування захисту коду використовували декомпілятор JetBrains dotPeek (рис.6) [7]. Цей безкоштовний засіб дозволяє декомпілювати .NET 1.0-4.5 збірки в C#, експортувати код у VisualStudio, підтримує підсвічування синтаксису та ряд інших функцій.

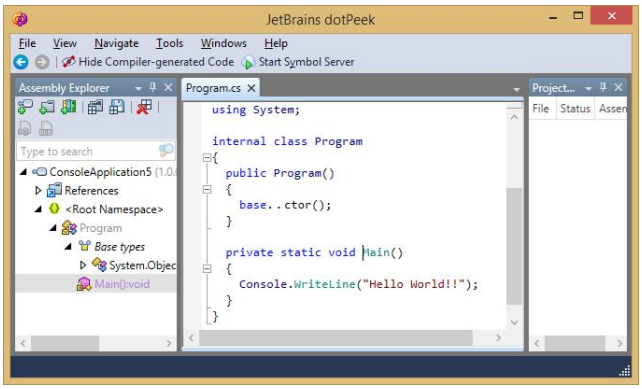


Рис. 6. C#-код програми HelloWorld, одержаний з незахищеної збірки за допомогою декомпілятораJetBrains dotPeek

Рис.7. Декомпіляція за допомогою JetBrains dotPeekмодифікованої за допомогою засобів Dotfuscator (а), .Net Reactor (б) і DotNet Protector (в) збірки програми HelloWorld

На рис. 7. Показані результати декомпіляції програми «HelloWorld»захищеної за допомогою вищенаведених засобів.

Висновок:аналіз показує, що сучасні засоби захисту забезпечують прийнятний ступінь захисту програмного коду .Net-програм від аналізу і небажаної модифікації. Програми для обфускації часто комбінують у собі додаткові механізми захисту, зокрема протидії декомпіляції, вбудовування у готові продукти систем реєстрації і перевірки ліцензій. Поєднання обфускації і пакування збірки здатне суттєво підвищити ступінь захисту, однак можливі негативні наслідки, що полягають у зниженні швидкодії програми, зменшення її стабільності. Комерційні засоби захисту гарантують вищий ступінь захисту коду в порівнянні з безкоштовними. В цілому, питання підвищення захищеностікоду для платформи .NetFramework, розробки нових механізмів і засобів захистузалишається актуальним.

Список використаних джерел

1. Шилдт Г. C# 4.0: полное руководство / Герберт Шилдт. — М.: «Вильямс», 2011. — 1056 с. — ISBN 978-5-8459-1684-6.
2. Обфускаторы (и деобфускаторы) для .NET §1 [Электронный ресурс]. – Режим доступа: URL : http://habrahabr.ru/post/74463/– заголовок с экрана.
3. Обфускация и защита программных продуктов.[Электронныйресурс]. – Режимдоступа: URL :http://citforum.ru/security/articles/obfus/– заголовоксэкрана.
4. Dotfuscator [Electronic Resource]. – Mode of access :URL : http://www.preemptive.com/. – Title from the screen.
5. .NET Reactor[Electronic Resource]. – Mode of access :URL : http://www.eziriz.com/ . – Title from the screen.
6. dotNet Protector [Electronic Resource]. – Mode of access :URL : http://dotnetprotector.pvlog.com/ . – Title from the screen.
7. JetBrains dotPeek [Electronic Resource]. – Mode of access :URL : https://www.jetbrains.com/decompiler/– Title from the screen.