«Обфускация кода»

**Цель работы:** Изучить методы и средства обфускации кода. Попробовать одно из средств обфускации на практике.

* Используемые системы

Linux

|  |
| --- |
| System**: Linux Mint 18** Sarah x86\_64  Kernel: 4.4.0-34-generic DE: X-Cinnamon Session: cinnamon  ----------------------------------  Processor: Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz  Memory (Gb): 3.7  Video: 00:02.0 VGA compatible controller: Intel Corporation 3rd Gen Core processor Graphics Controller (rev 09) Subsystem: Acer Incorporated [ALI] 3rd Gen Core processor Graphics Controller Kernel driver in use: i915 -- 01:00.0 VGA compatible controller: NVIDIA Corporation GF108M [GeForce GT 630M] (rev a1) Subsystem: Acer Incorporated [ALI] GF108M [GeForce GT 630M] Kernel driver in use: nvidia  ---------------------------------- |

Windows

|  |
| --- |
| C:\Users\Saboteur>systeminfo  Имя узла: SABOTEUR-PC  Название ОС: **Microsoft Windows 7** Максимальная  Версия ОС: 6.1.7601 Service Pack 1 сборка 7601  Изготовитель ОС: Microsoft Corporation  Параметры ОС: Изолированная рабочая станция  Сборка ОС: Multiprocessor Free  Зарегистрированный владелец: Saboteur  Зарегистрированная организация:  Код продукта: 00426-OEM-8992662-00006  Дата установки: 20.09.2015, 2:18:48  Время загрузки системы: 19.12.2015, 11:38:47  Изготовитель системы: Acer  Модель системы: Aspire V3-571G  Тип системы: x64-based PC  Процессор(ы): Число процессоров - 1.  [01]: Intel64 Family 6 Model 58 Stepping 9 Gen  uineIntel ~2501 МГц  Версия BIOS: Acer V1.13, 09.10.2012  Папка Windows: C:\Windows  Системная папка: C:\Windows\system32  Устройство загрузки: \Device\HarddiskVolume1  Язык системы: ru;Русский  Язык ввода: ru;Русский  Часовой пояс: (UTC+04:00) Волгоград, Москва, Санкт-Петербург  Полный объем физической памяти: 3 934 МБ  Доступная физическая память: 1 027 МБ  Виртуальная память: Макс. размер: 7 867 МБ  Виртуальная память: Доступна: 4 672 МБ  Виртуальная память: Используется: 3 195 МБ  Расположение файла подкачки: C:\pagefile.sys |

Программа работы

* Обфускаторы исходного кода языка С++
* Обфускаторы машинного кода

Обфускация или запутывание кода — приведение исходного текста или исполняемого кода программы к виду, сохраняющему её функциональность, но затрудняющему анализ, понимание алгоритмов работы и модификацию при декомпиляции. Цели обфускации:

* Затруднение декомпиляции/отладки и изучения программ с целью обнаружения функциональности.
* Затруднение декомпиляции проприетарных программ с целью предотвращения обратной разработки или обхода DRM и систем проверки лицензий.
* Оптимизация программы с целью уменьшения размера работающего кода и (если используется некомпилируемый язык) ускорения работы.
* Демонстрация неочевидных возможностей языка и квалификации программиста (если производится вручную, а не инструментальными средствами).

Обфускация кода может проводиться на нескольких уровнях:

* На уровне исходного кода
* На уровне машинного кода
* На уровне промежуточного (байт) кода

Обфускаторы исходного кода языка С++

**Stunnix**

Stunnix CXX-Obfus – это продвинутый кросс-платформенный (Windows, MacOS X и любой Unix поддерживаем!) профессиональный кодировщик и шифратор – решение для защиты интеллектуальной собственности, которая делает невозможным прочитать и использовать C/C++ коды, с продвинутым GUI – менеджером проектов – поддержка проектов с файлами, которыми сложно управлять, с использованием директив и макросов, смешанным C и C++ языков, которые используют любой диалект C/C++. GUI имеет сервисную программу для извлечения символов ActiveX. Все функции доступны через гибкую и мощную командную строку.

Для начала работы необходимо указать папку с исходными файлами, папку для хранения результатов обфускации и папку для промежуточных данных. В настройках обфускатора можно выбрать из нескольких стилей обфускации. Стили представлены в виде примеров(рис.1). Всего предлагается 9 разных варантов.

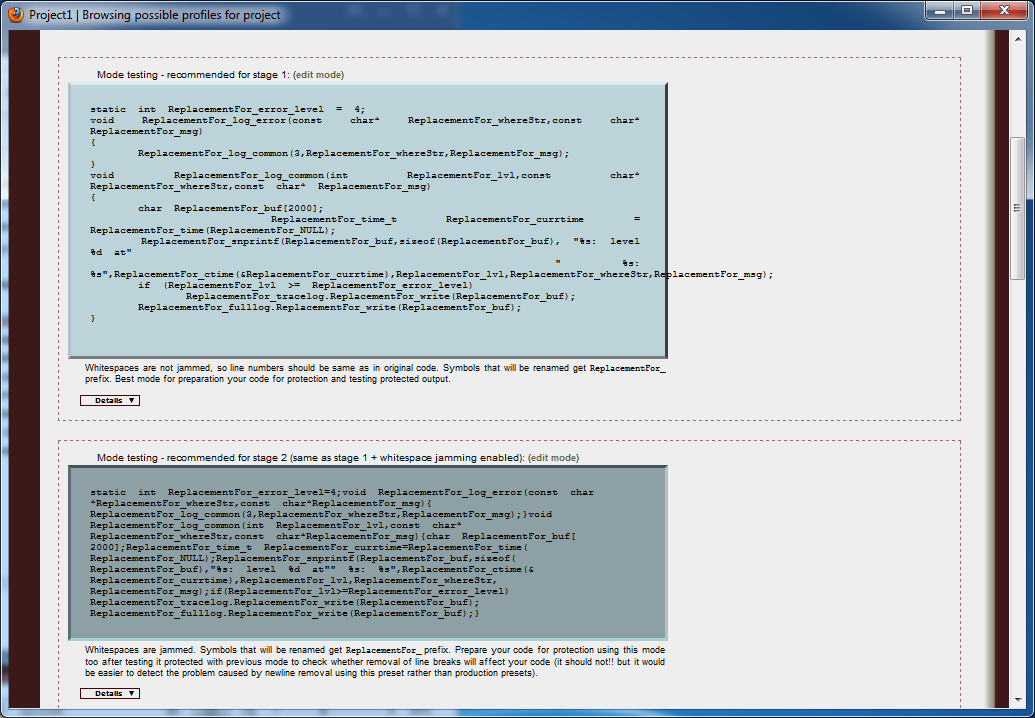


Рис.1. Стили обфускации.

После выбора стиля можно преступить к обфускации. Процесс длится достаточно долго. Как видно из рис.2

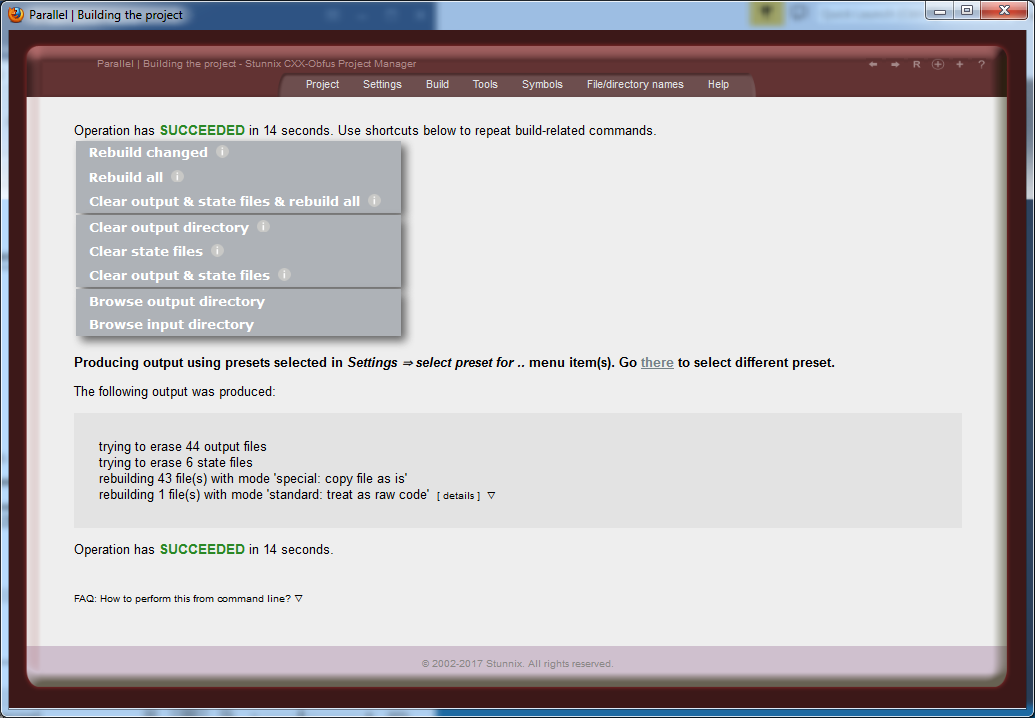


Рис.2 Завершенный процесс обфускации.

Для отображения результатов обфускации сравним файл main до и после.

|  |
| --- |
| #include <iostream>  #include <vector>  #include <string>  #include <fstream>  #include <sstream>  #include <cctype>  #include <ctime>  #include <unordered\_map>  using namespace std;  …  int main(int argc, char \*argv[])  {  setlocale(LC\_ALL, "rus");  cout << "Program was started" << endl;  string fileName("D:\\University\\Master2\\Parallel\\C\\ParallelSimple\\ParallelSimple\\Пикник на обочине.txt");//fileName(argv[0]);//("text.txt"); ("текст.txt");//    if (argc >= 2)  {  fileName = argv[1];//("text.txt"); ("текст.txt");//  }  vector<string> \*vecText = readFileByWords(fileName);  if (vecText == nullptr)  {  cout << "File not found\n" << endl;  delete vecText;  return 1;  }  vecText = formatFile(vecText);  unsigned int timeAfterReading = clock();  unordered\_map<string, int>\* resMap = computeFrequencyNew(3, vecText);  unordered\_map<string, int>::iterator itMax = findeMaxValue(resMap);  int n = itMax->second; //Числитель - число появлений конкретной nGramm-ы  int L = resMap->size(); //Знаменатель - число всех возможных nGramm  float frequency = ((float)n / (float)L) \* 100;  cout.setf(std::ios::fixed);  cout << "nGramm: " << itMax->first << endl;  cout << "Number of occurrences:" << n << endl;  cout << "Number of all nGramm:" << L << endl;  cout << "Frequency:" << frequency << "%" << endl;  cout.precision(2); //2 - число символов после точки  delete vecText;  delete resMap;  unsigned int timeEnd = clock();  unsigned int workTime = (timeEnd - timeAfterReading);// / 1000; //в секундах  cout << "runtime = " << timeEnd << endl; // время работы программы  cout << "runtime without reading = " << workTime << endl; // время работы программы  system("pause");  ofstream fout("workTime.txt", ios\_base::app); // открываем файл для добавления информации к концу файла  if (!fout.is\_open()) // если файл не открыт  cout << "Файл не может быть открыт!\n"; // сообщить об этом  fout << workTime << endl;  fout.close();  return 0;  } |

Файл main.cpp после обфускации

|  |
| --- |
| #include <iostream>  #include <vector>  #include <string>  #include <fstream>  #include <sstream>  #include <cctype>  #include <ctime>  #include <unordered\_map>  using namespace std;  …  int main(int ReplacementFor\_argc, char \*ReplacementFor\_argv[])  {  setlocale(LC\_ALL, "rus");  cout << "Program was started" << endl;  string ReplacementFor\_fileName("D:\\University\\Master2\\Parallel\\C\\ParallelSimple\\ParallelSimple\\Пикник на обочине.txt");    if (ReplacementFor\_argc >= 2)  {  ReplacementFor\_fileName = ReplacementFor\_argv[1];  }  vector<string> \*ReplacementFor\_vecText = ReplacementFor\_readFileByWords(ReplacementFor\_fileName);  if (ReplacementFor\_vecText == nullptr)  {  cout << "File not found\n" << endl;  delete ReplacementFor\_vecText;  return 1;  }  ReplacementFor\_vecText = ReplacementFor\_formatFile(ReplacementFor\_vecText);  unsigned int ReplacementFor\_timeAfterReading = clock();  unordered\_map<string, int>\* ReplacementFor\_resMap = ReplacementFor\_computeFrequencyNew(3, ReplacementFor\_vecText);  unordered\_map<string, int>::iterator ReplacementFor\_itMax = ReplacementFor\_findeMaxValue(ReplacementFor\_resMap);  int ReplacementFor\_n = ReplacementFor\_itMax->second;  int ReplacementFor\_L = ReplacementFor\_resMap->size();  float ReplacementFor\_frequency = ((float)ReplacementFor\_n / (float)ReplacementFor\_L) \* 100;  cout.setf(std::ios::fixed);  cout << "nGramm: " << ReplacementFor\_itMax->first << endl;  cout << "Number of occurrences:" << ReplacementFor\_n << endl;  cout << "Number of all nGramm:" << ReplacementFor\_L << endl;  cout << "Frequency:" << ReplacementFor\_frequency << "%" << endl;  cout.precision(2);  delete ReplacementFor\_vecText;  delete ReplacementFor\_resMap;  unsigned int ReplacementFor\_timeEnd = clock();  unsigned int ReplacementFor\_workTime = (ReplacementFor\_timeEnd - ReplacementFor\_timeAfterReading);  cout << "runtime = " << ReplacementFor\_timeEnd << endl;  cout << "runtime without reading = " << ReplacementFor\_workTime << endl;  system("pause");  ofstream ReplacementFor\_fout("workTime.txt", ios\_base::app);  if (!ReplacementFor\_fout.is\_open())  cout << "Файл не может быть открыт!\n";  ReplacementFor\_fout << ReplacementFor\_workTime << endl;  ReplacementFor\_fout.close();  return 0;  } |

В результате обфускатор удалил все комментарии, переименовал переменные. В ходе обфускации была переименованная одна библиотечная функция, что приводило к ошибке компиляции. При исправлении этой ошибки, обфусцированный код работает как и исходный.

Попробуем более сложный шаблон обфускации

|  |
| --- |
| #include <iostream>  #include <vector>  #include <string>  #include <fstream>  #include <sstream>  #include <cctype>  #include <ctime>  #include <unordered\_map>  using namespace std;string ReplacementFor\_formatStr(string&str){string  ReplacementFor\_newStr="";for(char&ReplacementFor\_ch:str){if(!iswalpha((unsigned  char)ReplacementFor\_ch))continue;ReplacementFor\_ch=tolower(ReplacementFor\_ch);  ReplacementFor\_newStr.push\_back(ReplacementFor\_ch);}return ReplacementFor\_newStr  ;}vector<string>\*ReplacementFor\_formatFile(vector<string>\*ReplacementFor\_text){  vector<string>\*ReplacementFor\_newVec=new vector<string>;for(string&str:\*  ReplacementFor\_text){str=ReplacementFor\_formatStr(str);if(strcmp(str.c\_str(),"")  !=0)ReplacementFor\_newVec->push\_back(str);}delete ReplacementFor\_text;return  ReplacementFor\_newVec;}vector<string>\*ReplacementFor\_readFileByWords(string  ReplacementFor\_fileName){vector<string>\*ReplacementFor\_vec=new vector<string>;  ifstream ReplacementFor\_file(ReplacementFor\_fileName);if(!ReplacementFor\_file.  is\_open())return nullptr;string ReplacementFor\_word;while(ReplacementFor\_file>>  ReplacementFor\_word){ReplacementFor\_vec->push\_back(ReplacementFor\_word);}return  ReplacementFor\_vec;}unordered\_map<string,int>\*ReplacementFor\_computeFrequencyNew  (int ReplacementFor\_n,vector<string>\*ReplacementFor\_text){if(!ReplacementFor\_n||  !ReplacementFor\_text->size())return nullptr;string ReplacementFor\_word;if(  ReplacementFor\_text->size()<ReplacementFor\_n)return 0;unordered\_map<string,int>\*  ReplacementFor\_resMap=new unordered\_map<string,int>;for(int ReplacementFor\_i=0;  ReplacementFor\_i<ReplacementFor\_text->size()-ReplacementFor\_n;ReplacementFor\_i++  ){string str="";for(int ReplacementFor\_j=0;ReplacementFor\_j<ReplacementFor\_n;  ReplacementFor\_j++){str=str+" "+ReplacementFor\_text->at(ReplacementFor\_i+  ReplacementFor\_j);}unordered\_map<string,int>::iterator ReplacementFor\_it;  ReplacementFor\_it=ReplacementFor\_resMap->find(str);if(ReplacementFor\_it==  ReplacementFor\_resMap->end())ReplacementFor\_resMap->insert(pair<string,int>(str,  1));else ReplacementFor\_it->second++;}return ReplacementFor\_resMap;}  unordered\_map<string,int>::iterator ReplacementFor\_findeMaxValue(unordered\_map<  string,int>\*ReplacementFor\_resMap){int ReplacementFor\_currentMax=0;unordered\_map  <string,int>::iterator ReplacementFor\_iter;for(auto ReplacementFor\_it=  ReplacementFor\_resMap->begin();ReplacementFor\_it!=ReplacementFor\_resMap->end();  ++ReplacementFor\_it){if(ReplacementFor\_it->second>ReplacementFor\_currentMax){  ReplacementFor\_currentMax=ReplacementFor\_it->second;ReplacementFor\_iter=  ReplacementFor\_it;}}return ReplacementFor\_iter;}int main(int ReplacementFor\_argc  ,char\*ReplacementFor\_argv[]){setlocale(LC\_ALL,"rus");cout<<"Program was started"  <<endl;string ReplacementFor\_fileName(  "D:\\University\\Master2\\Parallel\\C\\ParallelSimple\\ParallelSimple\\Пикник на обочине.txt"  );if(ReplacementFor\_argc>=2){ReplacementFor\_fileName=ReplacementFor\_argv[1];}  vector<string>\*ReplacementFor\_vecText=ReplacementFor\_readFileByWords(  ReplacementFor\_fileName);if(ReplacementFor\_vecText==nullptr){cout<<  "File not found\n"<<endl;delete ReplacementFor\_vecText;return 1;}  ReplacementFor\_vecText=ReplacementFor\_formatFile(ReplacementFor\_vecText);  unsigned int ReplacementFor\_timeAfterReading=clock();unordered\_map<string,int>\*  ReplacementFor\_resMap=ReplacementFor\_computeFrequencyNew(3,  ReplacementFor\_vecText);unordered\_map<string,int>::iterator ReplacementFor\_itMax  =ReplacementFor\_findeMaxValue(ReplacementFor\_resMap);int ReplacementFor\_n=  ReplacementFor\_itMax->second;int ReplacementFor\_L=ReplacementFor\_resMap->size();  float ReplacementFor\_frequency=((float)ReplacementFor\_n/(float)ReplacementFor\_L)  \*100;cout.setf(std::ios::fixed);cout<<"nGramm: "<<ReplacementFor\_itMax->first<<  endl;cout<<"Number of occurrences:"<<ReplacementFor\_n<<endl;cout<<  "Number of all nGramm:"<<ReplacementFor\_L<<endl;cout<<"Frequency:"<<  ReplacementFor\_frequency<<"%"<<endl;cout.precision(2);delete  ReplacementFor\_vecText;delete ReplacementFor\_resMap;unsigned int  ReplacementFor\_timeEnd=clock();unsigned int ReplacementFor\_workTime=(  ReplacementFor\_timeEnd-ReplacementFor\_timeAfterReading);cout<<"runtime = "<<  ReplacementFor\_timeEnd<<endl;cout<<"runtime without reading = "<<  ReplacementFor\_workTime<<endl;system("pause");ofstream ReplacementFor\_fout(  "workTime.txt",ios\_base::app);if(!ReplacementFor\_fout.is\_open())  cout<<"Файл не может быть открыт!\n";ReplacementFor\_fout<<  ReplacementFor\_workTime<<endl;ReplacementFor\_fout.close();return 0;} |

После обфускации так же возникла ошибка в библиотесной функции, однако ее исправление позволило скомпилировать прогамму.

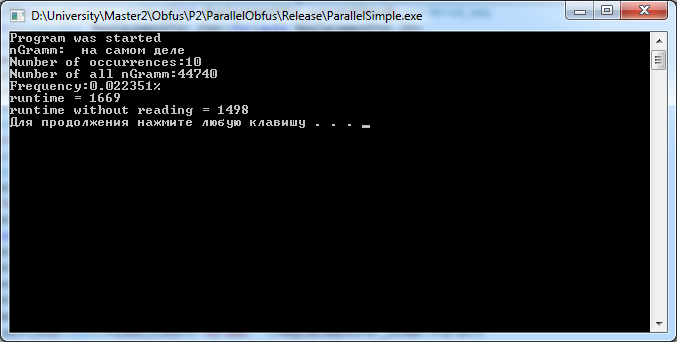


Рис. 3. Запуск обфусцированной программы

Обфусцируемая программа подсчитывает время своего выполнения. На рис. 4 можно увидеть время выполнения не обфусцированной программы

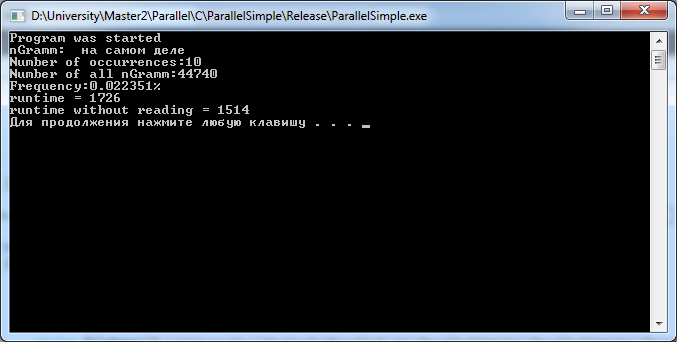


Рис.4. Выполнение не обфусцированной программы.

Как видно из рисунков 3 и 4 время выполнения программ мало отличается. Это является следствием того что обфускация происходит на уровне исходного кода, а не машинного.

Обфускаторы машинного кода

**VMProtect**

Программа VMProtect относится к новому поколению средств защиты программного обеспечения. VMProtect поддерживает компиляторы Delphi, Borland C Builder, Visual C/C++, Visual Basic (native), Virtual Pascal, при этом VMProtect содержит встроенный дизассемблер, позволяющий работать с файлами форматов EXE, DLL, BPL, OCX, SYS и подключать MAP-файл, создаваемый компилятором, для быстрого выбора участков кода, которые следует защитить. Для автоматизации операций по защите приложения в VMProtect реализован встроенный скриптовый язык. Программа VMProtect обладает полной поддержкой всех 32/64-разрядных операционных систем семейства Windows: Windows 95/98/ME, Windows NT, Windows 2000, Windows XP, Windows 2003, Windows Vista.

Для тестирования обфускатора была написана простейшая программа.

|  |
| --- |
| #include "iostream"  using namespace std;  int main(void)  {  for (int i = 0; i < 5; i++)  cout << "hello world" << endl;  system("pause");  return 0;  } |

Результат работы программы представлен на рис.5

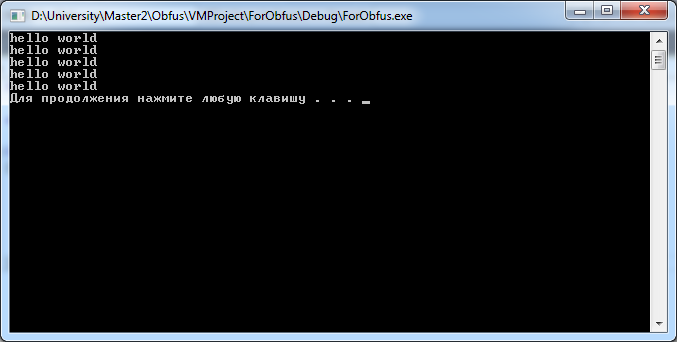


Рис.5. Работа программы.

Далее выбираем скомпилированную программу (exe файл) в VMProtect. Компилируем новую программу. Результат компиляции представлен на рисунке 6 и 7.

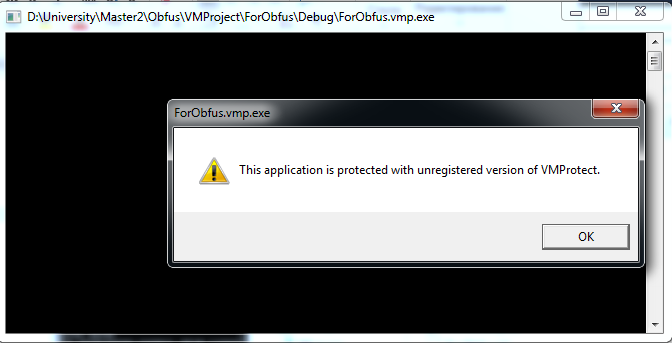


Рис.6. Всплывающее окно при запуске новой программы.

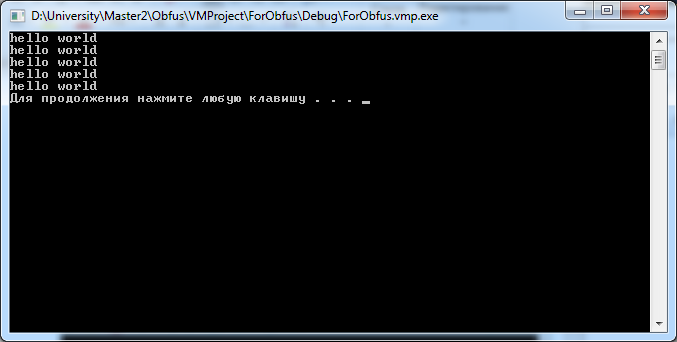


Рис.7. Прежняя функциональность программы сохранена.

Так как программа VMProtect служит для защиты машинного кода, нам необходимо проанализировать дизасемблированный код программы. Для этого будем использовать программу IDA Pro.

Поиск необходимого куска кода облегчен списком видимых функций (рис.8). Названия функций сохранены, т.к. эта информация осталась при компиляции в мышинном коде программы.

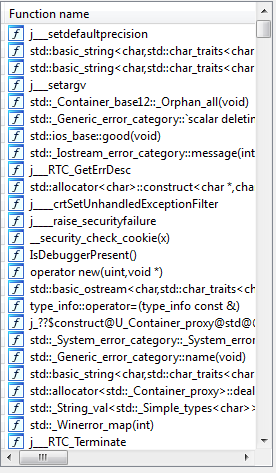


Рис. 8. Список функций исходной программы.

Далее представлен найденный код функции main в исходной программе.

|  |
| --- |
| .text:00415E70 ; =============== S U B R O U T I N E =======================================  .text:00415E70  .text:00415E70 ; Attributes: bp-based frame  .text:00415E70  .text:00415E70 ; int \_\_cdecl main()  .text:00415E70 \_main proc near ; CODE XREF: j\_\_mainj  .text:00415E70  .text:00415E70 var\_CC = byte ptr -0CCh  .text:00415E70 i = dword ptr -8  .text:00415E70  .text:00415E70 push ebp  .text:00415E71 mov ebp, esp  .text:00415E73 sub esp, 0CCh  .text:00415E79 push ebx  .text:00415E7A push esi  .text:00415E7B push edi  .text:00415E7C lea edi, [ebp+var\_CC]  .text:00415E82 mov ecx, 33h  .text:00415E87 mov eax, 0CCCCCCCCh  .text:00415E8C rep stosd  .text:00415E8E mov [ebp+i], 0  .text:00415E95 jmp short loc\_415EA0  .text:00415E97 ; ---------------------------------------------------------------------------  .text:00415E97  .text:00415E97 loc\_415E97: ; CODE XREF: \_main+5Fj  .text:00415E97 mov eax, [ebp+i]  .text:00415E9A add eax, 1  .text:00415E9D mov [ebp+i], eax  .text:00415EA0  .text:00415EA0 loc\_415EA0: ; CODE XREF: \_main+25j  .text:00415EA0 cmp [ebp+i], 5  .text:00415EA4 jge short loc\_415ED1  .text:00415EA6 mov esi, esp  .text:00415EA8 push offset j\_??$endl@DU?$char\_traits@D@std@@@std@@YAAAV?$basic\_ostream@DU?$char\_traits@D@std@@@0@AAV10@@Z ; std::endl<char,std::char\_traits<char>>(std::basic\_ostream<char,std::char\_traits<char>> &)  .text:00415EAD push offset \_Val ; "hello world"  .text:00415EB2 mov eax, ds:\_\_imp\_?cout@std@@3V?$basic\_ostream@DU?$char\_traits@D@std@@@1@A ; std::basic\_ostream<char,std::char\_traits<char>> std::cout  .text:00415EB7 push eax ; \_Ostr  .text:00415EB8 call j\_??$?6U?$char\_traits@D@std@@@std@@YAAAV?$basic\_ostream@DU?$char\_traits@D@std@@@0@AAV10@PBD@Z ; std::operator<<<std::char\_traits<char>>(std::basic\_ostream<char,std::char\_traits<char>> &,char const \*)  .text:00415EBD add esp, 8  .text:00415EC0 mov ecx, eax  .text:00415EC2 call ds:\_\_imp\_??6?$basic\_ostream@DU?$char\_traits@D@std@@@std@@QAEAAV01@P6AAAV01@AAV01@@Z@Z ; std::basic\_ostream<char,std::char\_traits<char>>::operator<<(std::basic\_ostream<char,std::char\_traits<char>> & (\*)(std::basic\_ostream<char,std::char\_traits<char>> &))  .text:00415EC8 cmp esi, esp  .text:00415ECA call j\_\_\_RTC\_CheckEsp  .text:00415ECF jmp short loc\_415E97  .text:00415ED1 ; ---------------------------------------------------------------------------  .text:00415ED1  .text:00415ED1 loc\_415ED1: ; CODE XREF: \_main+34j  .text:00415ED1 mov esi, esp  .text:00415ED3 push offset Command ; "pause"  .text:00415ED8 call ds:\_\_imp\_\_system  .text:00415EDE add esp, 4  .text:00415EE1 cmp esi, esp  .text:00415EE3 call j\_\_\_RTC\_CheckEsp  .text:00415EE8 xor eax, eax  .text:00415EEA pop edi  .text:00415EEB pop esi  .text:00415EEC pop ebx  .text:00415EED add esp, 0CCh  .text:00415EF3 cmp ebp, esp  .text:00415EF5 call j\_\_\_RTC\_CheckEsp  .text:00415EFA mov esp, ebp  .text:00415EFC pop ebp  .text:00415EFD retn  .text:00415EFD \_main endp  .text:00415EFD  .text:00415EFD ; ---------------------------------------------------------------------------  .text:00415EFE db 24h dup(0CCh) |

На рисунке 9 представлен цикл for, в котором выводится фраза “Hello world”

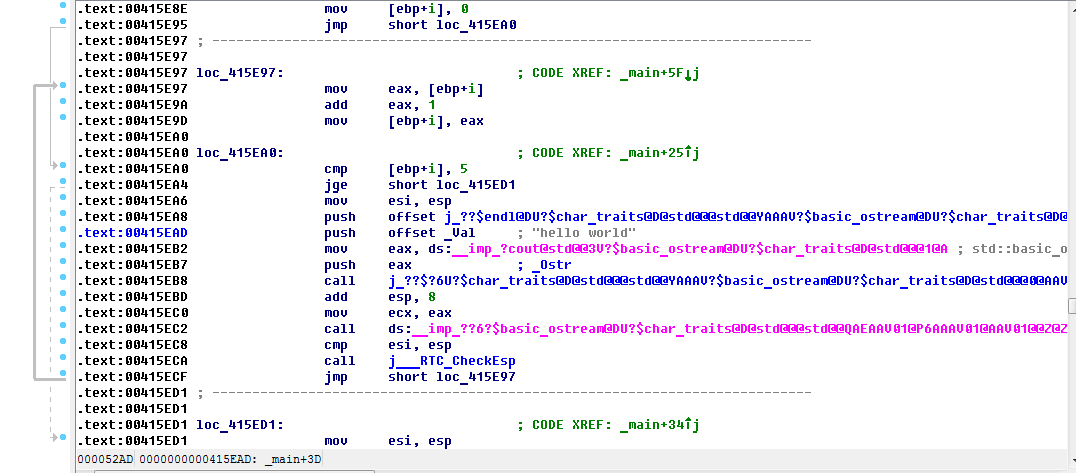


Рис. 9. Код цикла for.

По адресу 00415EA0 можно видеть команду cmp, которая проверяет, не достиг ли счетчик значения 5.

Попробуем найти что-то подобное в программе, обработанной с помощью VMProtect. На рисунке 10 представлен вид модифицированной программы.

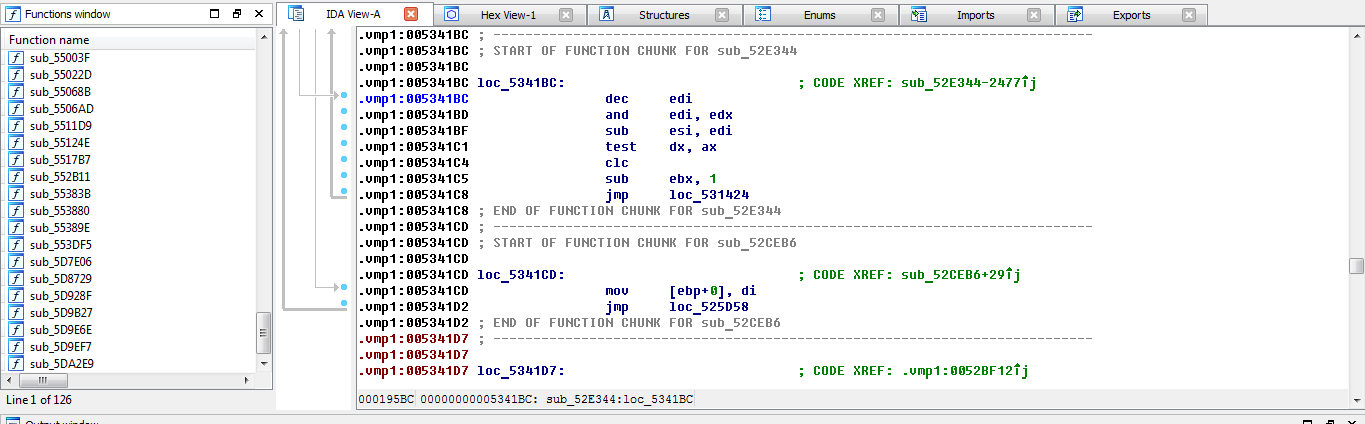


Рис. 10. Машинный код модифицированной программы.

На рис.10 видно, что вся информация о именях функций удалена.

Поиск выражения «Hello world» ни к чему не привел.

IDA Pro позволяет запускать программу в режиме отладки. Для исходной программы это делается легко (рис.11)

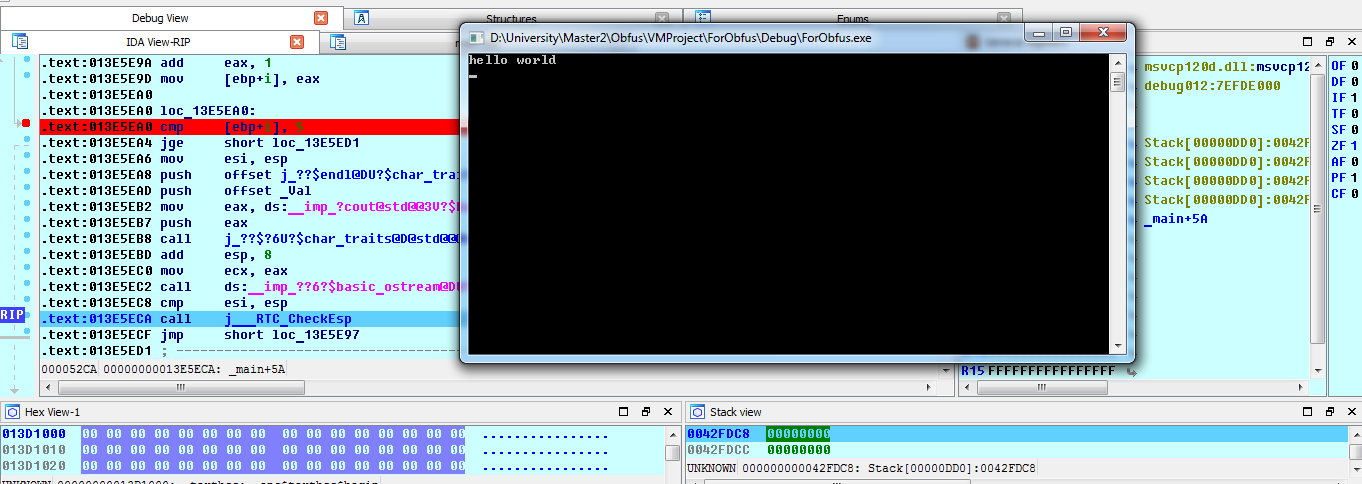


Рис. 11. Запуск исходной программы в режиме отладки.

При попытке запустить модифицированную программу до точки останова всплывает ошибка (рис. 12).

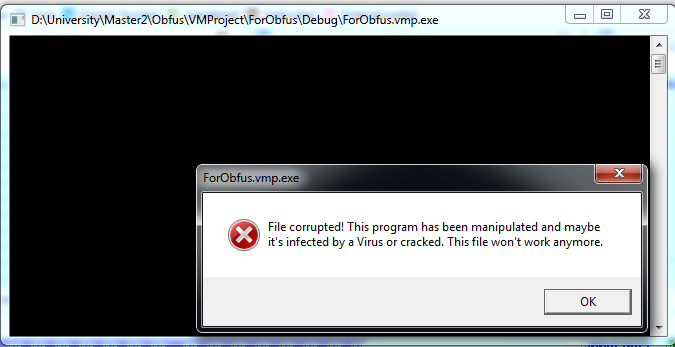


Рис.12. Запуск модифицированной программы в режиме отладки.

Таким образом можно сделать вывод, что обфускация машинного кода программы значительно затрудняет ее исследование.

Вывод

В данной работе был протестирован коммерческий обфускатор Stunnix

. С помощь его пробной версии, был обфусцирован исходный код программы. Рассмотрен обфускатор машинного кода VMProtect. Произведено сравнение ассемблерных кодов исходной и обфусцированной программы. В результате, установлено, что модифицированную программу достаточно сложно исследовать. Однако исследование не является невыполнимой задачей. Но на него необходимо затратить большое количество усилий, что может быть эквивалентно написанию аналогичной программы.