ВОЕННО-КОСМИЧЕСКАЯ АКАДЕМИЯ имени А.Ф.МОЖАЙСКОГО

Кафедра № 27 Математического и программного обеспечения

УТВЕРЖДАЮ

Начальник 27 кафедры

полковник

С. Войцеховский

«___» _____ 2022 г.

Практическое занятие № 1
по учебной дисциплине
«Защита информации»
на тему:

«**Защита программных средств от несанкционированного копирования, исследования, модификации**»

Рассмотрено и одобрено
на заседании кафедры № 27
« __ » _____ 2022 г. протокол № __

Санкт-Петербург
2022

I. ТЕМА И ЦЕЛЬ ПРАКТИЧЕСКОГО ЗАНЯТИЯ

Тема практического занятия: «Защита программных средств от несанкционированного копирования, исследования, модификации».

Учебная цель: овладение навыками составления и отладки модуля защиты ПО от копирования.

Время - 180 мин.

Место – аудитория (класс) по расписанию занятий.

Учебно-материальное и методическое обеспечение

1. Лабораторные установки – персональные ЭВМ с установленным на них программным обеспечением.
2. Методические разработки по программированию модулей защиты ПО от копирования, исследования и модификации.
3. Варианты типовых заданий на практическое занятие.

II. УЧЕБНЫЕ ВОПРОСЫ И РАСЧЕТ ВРЕМЕНИ

№ п\п	Учебные вопросы	Время, мин.
1.	Вступительная часть. Контрольный опрос.	10
2.	Учебные вопросы. ОСНОВНАЯ ЧАСТЬ: 1. Разработка программного модуля для защиты ПО от копирования. 2. Проверка работоспособности модуля путём установки его на любую программу. 3. Составление отчёта о проделанной работе, защита программы у преподавателя.	 80 40 45
3.	Заключительная часть. Задание и методические указания курсантам на самостоятельную подготовку	5

III. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПРЕПОДАВАТЕЛЮ ПРИ ПРОВЕДЕНИИ ПРАКТИЧЕСКОГО ЗАНЯТИЯ

Во вступительной части занятия производится контроль присутствия и готовности обучающихся к занятию. Объявляется тема, цель, учебные вопросы занятия и особенности его проведения.

Готовность группы к занятию проверяется контрольным опросом.

Вопрос 1: Что подразумевают под понятием защита от копирования?

Вопрос 2: Какими способами организуются защита от копирования?

Вопрос 3: Для чего необходим механизм защиты от копирования?

Вопрос 4: Перечислите простые методы защиты компакт дисков?

Вопрос 5: Какие особенности архитектуры ЭВМ могут использоваться в качестве эталонных характеристик?

При отработке первого вопроса занятия основное внимание обратить на усвоение обучающимися принципов построения модулей защиты программ от копирования программ и их реализацию средствами языка высокого уровня.

При отработке второго вопроса отметить необходимость и важность встраивания модуля защиты от копирования в структуру любой программы, как решающего фактора своевременного и правильного решения задачи защиты ПО.

При отработке третьего вопроса необходимо акцентировать внимание на структуре отчета о проделанной работе и защите его основных положений.

В заключительной части занятия подвести итоги, оценить действия обучающихся, ответить на вопросы.

Дать задание на самоподготовку. Объявить тему следующего занятия.

IV. УЧЕБНЫЕ МАТЕРИАЛЫ

1. Сведения из теории

Угроза несанкционированного копирования информации блокируется методами, которые могут быть распределены по двум группам:

- методы, затрудняющие считывание скопированной информации;
- методы, препятствующие использованию информации.

• **Методы, затрудняющие считывание скопированной информации** основываются на придании особенностей процессу записи информации, которые не позволяют считывать полученную копию на других накопителях, не входящих в защищаемую КС. Самым простым решением является нестандартная разметка (форматирование) носителя информации. Нестандартное форматирование защищает только от стандартных средств работы с накопителями. Использование специальных программных средств (например, DISK EXPLORER для IBM-совместимых ПЭВМ) позволяет получить характеристики нестандартного форматирования.

• **Методы, препятствующие использованию скопированной информации** имеет целью затруднить использование полученных копированием данных. Скопированная информация может быть программой или данными. Данные и программы могут быть защищены, если они хранятся на ВЗУ в преобразованном *криптографическими методами* виде. Наиболее действенным (после криптогра-

фического преобразования) методом противодействия несанкционированному выполнению скопированных программ является использование *блока контроля среды размещения программы*. Блок контроля среды размещения является дополнительной частью программ. Он создается при инсталляции (установке) программ. В него включаются эталонные характеристики среды, в которой размещается программа, а также средства получения и сравнения характеристик.

В качестве эталонных характеристик ЭВМ могут использоваться особенности архитектуры:

- тип и частота центрального процессора,
- номер процессора (если он есть),
- состав и характеристики внешних устройств, особенности их подключения (например, уникальный MAC-адрес сетевой карты),
- показатели быстродействия жесткого диска и процессора,
- режимы работы блоков и устройств,
- и т. п.

Общий *алгоритм механизма защиты от несанкционированного использования программ* в «чужой» среде размещения сводится к выполнению следующих шагов:

Шаг 1. Запоминание множества индивидуальных эталонных характеристик ЭВМ и (или) съемного носителя информации на этапе инсталляции защищаемой программы.

Шаг 2. При запуске защищенной программы управление передается на блок контроля среды размещения. Блок осуществляет сбор и сравнение характеристик среды размещения с контрольными характеристиками.

Шаг 3. Если сравнение прошло успешно, то программа выполняется, иначе - отказ в выполнении. Отказ в выполнении может быть дополнен выполнением деструктивных действий в отношении этой программы, приводящих к невозможности выполнения этой Программы, если такую самоликвидацию позволяет выполнить ОС.

Наиболее высокий уровень защиты программ от копирования достигается при *комбинировании различных способов привязки* к уникальным характеристикам аппаратно-программной среды компьютера.

Различают следующие **способы привязки к аппаратной конфигурации персонального IBM-совместимого компьютера:**

1) привязка к особенностям постоянного запоминающего устройства компьютера (ROM BIOS);

2) привязка к списку компьютерного оборудования.

В первом случае в качестве эталонных характеристик выступают контрольная сумма или дата изготовления BIOS. Дата изготовления BIOS хранится в восьми байтах внутренней памяти компьютера по адресу FOOO:FFF516.

При привязке к списку компьютерного оборудования в качестве эталонных характеристик выступает сам список оборудования, который можно получить путем использования соответствующей функции операционной системы.

Привязка программ к среде размещения требует повторной их инсталляции после проведения модернизации, изменения структуры или ремонта КС с заменой устройств. Для защиты от несанкционированного использования программ, могут применяться и электронные ключи. Наиболее надежным способом привязки к аппаратной конфигурации компьютера является *привязка к уникальному номеру процессора*. Однако для большинства персональных IBM-совместимых компьютеров этот способ нереализуем по причине невозможности программного доступа к этому уникальному номеру.

2. Практические особенности реализации рассмотренных механизмов с помощью API-функции:

GetUserName – Имя текущего пользователя.

GetComputerName – Имя компьютера.

GetVolumeInformation – Получение информации о носителе.

GlobalMemoryStatus – Информация об используемой системой памяти.

Рассмотрим как с помощью API реализовать возможность привязки программы к типу носителя или его серийному номеру.

Нам понадобится 2-е API-функции:

• GetDriveType - определяет и возвращает тип носителя;

• GetVolumeInformation - определяет информацию о носителе, среди которой содержится серийный номер.

Рассмотрим описание этих функций для C++ и Delphi. Первой будет функция GetDriveType, она очень простая и использует всего один параметр - указатель на том. Например “c:\”,”a:\” и т.д.

Функция возвращает одно из следующих значений:

```
DRIVE_UNKNOWN - 0 : диск не определён /не существует
DRIVE_NO_ROOT_DIR - 1 : неверный путь/ путь не указывает на том
DRIVE_REMOVABLE - 2 : тип устройства определяется как съемный (дискета, флэшка и т.д.)
DRIVE_FIXED - 3 : тип устройства - фиксированный диск (жесткий диск)
```

```
DRIVE_REMOTE - 4 : тип устройства – удаленный (сетевой) диск
DRIVE_CDROM - 5 : это устройство CD-ROM
DRIVE_RAMDISK - 6 : виртуальный диск, созданный в оперативной памяти
```

C/C++

```
UINT WINAPI GetDriveType(
  LPCTSTR lpRootPathName //путь к диску
);
```

Delphi

```
function GetDriveType(lpRootPathName: PChar): UINT; stdcall;
```

Здесь в качестве параметра передается путь к диску.

Замечание: Если в качестве параметра указать для C/C++ NULL, а для Delphi - nil то тип устройства будет определяться для текущего диска (с которого была запущена программа).

А теперь взглянем на функцию GetVolumeInformation. Тоже достаточно простая функция, однако использует параметров значительно больше.

C/C++

```
BOOL WINAPI GetVolumeInformation(
 LPCTSTR lpRootPathName, //путь к сетевому или локальному тому
 // пример: "\\MyServer\MyShare\" или "C:\".
 LPTSTR lpVolumeNameBuffer, //буфер - в котором будет храниться имя тома
 DWORD nVolumeNameSize, //размер буфера
 LPDWORD lpVolumeSerialNumber, //серийный номер тома
 LPDWORD lpMaximumComponentLength, //размер тома
 LPDWORD lpFileSystemFlags, //тип файловой системы
 LPTSTR lpFileSystemNameBuffer, //название файловой системы
 DWORD nFileSystemNameSize //размер буфера под название ФС
);
```

Delphi

```
function GetVolumeInformation(
 lpRootPathName: PChar; //путь к сетевому или локальному тому
 // пример: "\\MyServer\MyShare\" или "C:\".
 lpVolumeNameBuffer: PChar; //буфер - в котором будет храниться имя тома
 nVolumeNameSize: DWORD; //размер буфера
```

```
lpVolumeSerialNumber: PDWORD; //серийный номер тома
var lpMaximumComponentLength, lpFileSystemFlags: DWORD; //размер тома
// и тип файловой системы
lpFileSystemNameBuffer: PChar; //название файловой системы
nFileSystemNameSize: DWORD //размер буфера под название ФС
): BOOL; stdcall;
```

Замечание: Если в качестве первого параметра указать для C/C++ NULL, а для Delphi - nil то функция будет выполняется для текущего диска (с которого была запущена программа).

3. Пример разработки программного модуля

Вариант задания

Осуществить привязку любой программы к флэш-устройству.

Текст программы защиты от копирования

C/C++

```
#include
#include
#include
#include

using namespace std;

int main() {
 // Получаем тип носителя с которого запущена программа
 unsigned int drive_type = GetDriveType( NULL );

 char VolumeNameBuffer[100];
 char FileSystemNameBuffer[100];

 DWORD sz,fs;
 unsigned long drive_sn;
GetVolumeInformationA(NULL, VolumeNameBuffer, 100, &drive_sn,
sz, fs, FileSystemNameBuffer, 100);

 cout << "Volume serial number:\t";
 if(drive_sn == 1018821877) //сравниваем серийный номер
 cout << "correct" << endl;
```

```
  else
  cout << "invalid" << endl;

  cout << "Drive type:\t";
  if(drive_type == DRIVE_REMOVABLE)
  cout << "correct" << endl;
  else
  cout << "invalid" << endl;

  getch();
}
```

Delphi

```
program Project1;

{$APPTYPE CONSOLE}

uses SysUtils,windows;

var
  SerialNum,dtyp: DWORD;
  a,b: DWORD;
  Buffer,disk :Array [0..255] of char;

begin
  dtyp := GetDriveType(nil);
  if dtyp = DRIVE_REMOVABLE then
  writeln('Disk(type): Yes')
  else
  writeln('Disk(type): No');

  GetVolumeInformation(nil, Buffer, sizeof(Buffer), @SerialNum, a, b, nil, 0);
  if SerialNum = 1018821877 then //сравниваем серийный номер
  writeln('S\N: Yes')
  else
  writeln('S\N: No');

  readln;
end.
```

Замечание: Может возникнуть вопрос, а как узнать серийник диска, чтобы знать с чем сравнивать? Очень просто, для этого пишем тестовую прогу, в которой пишем следующий код:

C/C++

```
...
GetVolumeInformationA(NULL, VolumeNameBuffer, 100, &drive_sn, sz, fs,
  FileSystemNameBuffer, 100);
...
```

Delphi

```
...
GetVolumeInformation(nil, Buffer, sizeof(Buffer), @SerialNum, a, b, nil, 0);
writeln('S/N drive: ', SerialNum);
readln;
...
```

Исходные данные и результаты выполнения программы

Ввод
Текст любой программы к которой будет привязан программный модуль защиты от копирования

Вывод
Отсутствие возможности запуска программы с модулем защиты от копирования на другой ЭВМ.

По завершении реализации программы оформляется письменный отчет, в который помещается текст задания, текст программы и модуля, структурная схема программы и выводы по работе.
Пример отчета приведен в приложении к Методической разработке.

4. Общие методические указания курсантам (слушателям) по подготовке к практическим занятиям

Практические занятия по дисциплине «Защита информации» проводятся в классе ПЭВМ. Индивидуальные задания выполняются каждым курсантом лично.

Перед выполнением задания обучающийся изучает материал, приведенный в разделе «Учебные материалы», в ходе которого необходимо разобрать приведенные примеры и выполнить задания раздела. На следующем этапе работы обучающийся выполняет индивидуальное задание.

Результаты работы оформляются в виде отчета. Содержание отчета приведено в руководстве по соответствующему практическому занятию.

По готовности к защите работы курсант (слушатель) докладывает преподавателю.

5. Индивидуальные задания к практическому занятию №1

Задача: Используя любые методы для защиты программ от копирования изученные на лекционных занятиях, используя эталонные характеристики ПЭВМ разработать:

1. Алгоритм программного модуля для защиты ПО от копирования.
2. Написать программу которая запускается только на компьютере на котором она была инсталлирована.

Общие пояснения.

Для создания эталонных характеристик могут быть использованы данные полученные следующим образом:

Вариант 1.

Написать программу, определяющую:

а) тип ПЭВМ;

в) дату создания BIOS.

Указания:

а) Информация о типе ПЭВМ находится по адресу F000:FFFE (1 байт).

в) Дата создания BIOS находится по адресу F000:FFF5 и занимает 8 байт в формате мм/дд/гг.

Вариант 2.

Написать программу, определяющую аппаратную конфигурацию ПЭВМ. Указание:

Слово аппаратной конфигурации находится по адресу: 0040:0010. Использовать абсолютные переменные. Биты слова аппаратной конфигурации:

0 - наличие дискового пространства;
1 - наличие сопроцессора;
4-5 - текущий видеорежим;
6-7 - число дисководов;
9-11 - число RS232 портов (адаптеров коммуникации);
12 - наличие игрового адаптера;
14-15 - число установленных принтеров.

Вариант 3.

Задание и указания аналогичны Заданию 2.

а) Использовать прерывание 11h. Int 11h возвращает в AX биты.
б) Использовать API-функции.

Вариант 4.

Написать программу, определяющую:

а) тип диска A: (количество сторон и секторов);
б) количество доступной памяти в Кбайтах.

Указания:

а) Использовать API-функции.
б) Использовать функцию 4Ah прерывания 21h.
в) Использовать функцию 1Ch прерывания 21h.

Вариант 5.

Написать программу эмулирующую действия системной команды "Time" .

Указания:

а) Использовать API-функции.
б) Использовать 2Ch и 2Dh прерывания 21h (получение и установка времени соответственно).
в) Осуществить верификацию вводимых данных. При ошибке ввода повторить запрос.

Вариант 6.

Написать программу, определяющую размер свободного пространства на диске.

Указания:

а) Использовать API-функции.
б) Для определения свободного пространства на диске использовать функцию 36h прерывания 21h.

Задание повышенной сложности:

Написать программу, реализующую 1 уровень возможностей с точки зрения модели нарушителя (запуск программ из фиксированного списка) формирования экранной заставки с расположенными на ней кнопками запуска двух-трех разрешенных программ. Обеспечить блокировку комбинации клавишей Ctrl-Alt-Del и Alt-Tab на момент работы программы, внести программу в список резидентных программ.

6. Отчетность по работе

По выполнению работы каждый курсант должен представить отчет. Отчет должен содержать:

– название практического занятия;

– текст индивидуального задания;

– блок-схему алгоритма решения задачи;

– исходный текст программы;

– результаты тестирования решения.

В процессе выполнения индивидуального задания или после завершения его выполнения преподаватель проводит собеседование с каждым курсантом по теме выполненной работы, проверяя также практические навыки, приобретенные в ходе занятия. Отчетный материал предоставляется преподавателю, а результаты защищаются.

7. Заключительная часть

В заключительной части подводятся итоги проделанной работы, дается краткая оценка действиям участников, прослеживается связь с теоретическими положениями и перспективой на будущую деятельность

8. Задание и методические указания курсантам на самостоятельную подготовку:

1. Повторить по конспекту лекций и рекомендованной литературе основные методы защиты от копирования.
2. Быть готовыми к самостоятельному составлению программ с использованием программных модулей защиты от копирования.

V. ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА

1. Войцеховский С.В., Воробьёв Е.Г. Методы и средства защиты компьютерной информации: учебно-методическое пособие. – СПб.: ВКА имени А.Ф. Можайского 2013. – 134 с.
2. Вихорев С.В. Классификация угроз информационной безопасности. -

http://www2.cnews.ru/comments/security/elvis_class.shtml.
3. Грибунин В.Г., Оков И.Н., Туринцев И.В. Цифровая стеганография. – М.:СОЛОН-Пресс, 2002. – 272 с.
4. Войцеховский С.В., Воробьёв Е.Г. Методы и средства защиты компьютерной информации: учебно-методическое пособие. – СПб.: ВКА имени А.Ф. Можайского 2013. – 134 с.
5. Вихорев С.В. Классификация угроз информационной безопасности. - http://www2.cnews.ru/comments/security/elvis_class.shtml.

Доцент 27 кафедры
к.т.н.
подполковник

С. Краснов

«___»_____20__г.