



AVZ - справка по работе с программой

AVZ, (C) Зайцев О.В., <http://z-oleg.com/secur>

Note:

To change the product logo for your own print manual or PDF, click "Tools > Manual Designer" and modify the print manual template.

Title page 1

Use this page to introduce the product

by Заїцес О.В.

This is "Title Page 1" - you may use this page to introduce your product, show title, author, copyright, company logos, etc.

This page intentionally starts on an odd page, so that it is on the right half of an open book from the readers point of view. This is the reason why the previous page was blank (the previous page is the back side of the cover)

AVZ - справка по работе с программой

AVZ, (C) Зайцев О.В., <http://z-oleg.com/secur>

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Printed: Ноябрь 2007 in (whereever you are located)

Publisher

...enter name...

Managing Editor

...enter name...

Technical Editors

...enter name...

...enter name...

Cover Designer

...enter name...

Team Coordinator

...enter name...

Production

...enter name...

Special thanks to:

All the people who contributed to this document, to mum and dad and grandpa, to my sisters and brothers and mothers in law, to our secretary Kathrin, to the graphic artist who created this great product logo on the cover page (sorry, don't remember your name at the moment but you did a great work), to the pizza service down the street (your daily Capricciosas saved our lives), to the copy shop where this document will be duplicated, and and and...

Last not least, we want to thank EC Software who wrote this great help tool called HELP & MANUAL which printed this document.

Table of Contents

Foreword	12
Part I Общие сведения	14
1 О программе.....	14
2 Лицензионное соглашение.....	16
3 Установка и системные требования.....	16
4 Структура папок программы.....	17
5 Техническая поддержка.....	17
6 Назначение программы.....	18
Part II Работа с программой	22
1 Главное окно программы.....	22
Закладка 'Область поиска'	23
Закладка 'Типы файлов'	24
Закладка 'Параметры поиска'	26
Группа 'Параметры лечения'	28
Протокол	29
2 Обновление баз.....	29
3 Отложенное удаление файла.....	31
4 Профили настроек.....	32
Part III Карантин	34
1 Карантин и папка Infected.....	34
2 Автокарантин.....	35
3 Добавление в карантин по списку.....	36
Part IV Встроенные средства поиска	39
1 Поиск данных в реестре.....	39
2 Поиск файлов на диске.....	40
3 Поиск Cookie по данным.....	42
Part V Встроенные утилиты	45
1 Диспетчер процессов.....	45
2 Диспетчер служб и драйверов.....	46
3 Модули пространства ядра.....	48
4 Менеджер Winsock SPI (LSP, NSP, TSP).....	49
5 Открытые порты TCP/UDP.....	50
6 Менеджер автозапуска.....	50
7 Менеджер расширений IE.....	51

8 Менеджер расширений проводника.....	51
9 Менеджер апплетов панели управления (CPL).....	52
10 Менеджер расширений системы печати.....	52
11 Менеджер планировщика заданий (Task Scheduler).....	53
12 Менеджер внедренных dll.....	53
13 Менеджер протоколов и обработчиков.....	54
14 Менеджер Active Setup.....	54
15 Менеджер файла Hosts.....	55
16 Общие ресурсы и сетевые сеансы.....	55
Part VI Функции анализа и восстановления	57
1 Исследование системы.....	57
2 Восстановление системы.....	59
3 Стандартные скрипты.....	63
4 Резервное копирование.....	64
5 Мастер поиска и устранения проблем.....	65
Part VII Подсистема AVZGuard	68
1 О технологии.....	68
2 Управление системой.....	69
3 AVZGuard и антитрассировка.....	70
Part VIII Подсистема AVZPM	72
1 О технологии.....	72
2 Управление системой.....	72
Part IX Подсистема Boot Cleaner	75
1 О технологии.....	75
Part X Ревизор	78
1 О технологии.....	78
2 Создание базы.....	79
3 Сравнение.....	80
Part XI Категории вредоносных программ	84
1 Вирусы.....	84
2 AdWare.....	84
3 Spy или SpyWare.....	84
4 PornWare и Dialer.....	85
5 Hijacker.....	85
6 RiskWare.....	85

Part XII	Дополнительная информация	87
1	Что такое RootKit.....	87
2	Эвристический анализатор AVZ.....	87
3	Local Service Provider (LSP/SPI).....	89
Part XIII	Параметры командной строки	93
1	Основные параметры.....	93
2	Специализированные ключи.....	96
3	Коды возврата.....	97
4	Примеры	97
Part XIV	Часто задаваемые вопросы (FAQ)	99
1	Что делать, если AVZ обнаружил подозрение на вирус или вредоносную программу ?.....	99
2	Что делать, если AVZ выдал подозрение на keylogger ?.....	99
3	Что делать, если AVZ выдал подозрение на RootKit ?.....	99
4	Что такое 'Порт TCP 5000' и как с ним бороться ?.....	100
5	Что делать, если после распаковки AVZ не может найти файл *.avz ?	100
6	AVZ удалил вредоносные программы, но стартовая страница и прочие настройки браузера по прежнему повреждены.....	100
7	Во время сканирования диска мой антивирусный монитор сообщил, что файл avz*.tmp заражен вирусом или является вредоносной программой.....	101
8	Что делать, если выдается сообщение "Ошибка загрузки драйвера - проверка прервана".....	101
9	Что делать, если в ходе обновления баз возникает ошибка.....	101
10	Что делать, если в ходе загрузки архива с AVZ возникают ошибки	102
11	В ходе проверки AVZ в активное окно вводится текстовая строка с текстом "test".....	102
12	Под Vista не работают некоторые функции AVZ.....	102
13	Особенности применения AVZ на Windows Server 2003.....	102
14	Что делать, если вредоносная программа блокирует запуск AVZ ?	103
Part XV	Скрипты управления	105
1	Введение	105
2	Структура скрипта.....	105
3	procedure Sleep.....	106
4	procedure ActivateWatchDog.....	106
5	procedure SetupAVZ.....	106

6	procedure RunScan	107
7	procedure ExitAVZ	107
8	function GetSystemBootMode	107
9	function GetEnvironmentVariable	107
10	Информация о компьютере	108
	function GetComputerName	108
	function GetComputerComments	108
	function IsNT	108
11	Взаимодействие с пользователем	109
	procedure ShowMessage	109
	function InputBox	109
	function MessageDLG	109
12	Автоматическое обновление	110
	function ExecuteAVUpdate	110
	function ExecuteAVUpdateEx	111
13	Работа с протоколом	112
	procedure AddToLog	112
	procedure SaveLog	112
	procedure SaveCSVLog	112
	function AddLineToTxtFile	113
14	Карантин и папка Infected	113
	procedure ExecuteAutoQuarantine	113
	function CreateQuarantineArchive	113
	function CreateInfectedArchive	114
	function QuarantineFile	114
	function ClearQuarantine	115
15	Функции запроса информации о AVZ	115
	function GetAVZDirectory	115
	function GetScanPath	116
	function GetAVZVersion	116
	function GetAVZVersionTxt	117
	function GetAVZSvcName	117
16	Интерфейс AVZ	117
	procedure UnLockInterface	117
	procedure LockInterface	117
	procedure SetStatusBarText	117
17	Управление AVZPM	117
	function SetAVZPMStatus	117
	function GetAVZPMStatus	118
18	Управление AVZGuard	118
	function SetAVZGuardStatus	118
	function GetAVZGuardStatus	118
19	Управление Boot Cleaner	119
	function BC_Activate	119
	function BC_DeActivate	119
	function BC_Execute	119
	function BC_Clear	120
	function BC_LogFile	120
	function BC_QrFile	120

function BC_QrSvc	121
function BC_DeleteFile	121
function BC_CopyFile	122
function BC_DeleteReg	122
function BC_DeleteSvcReg	122
function BC_DeleteSvc	122
function BC_DisableSvc	122
function BC_ImportDeletedList	123
function BC_ImportQuarantineList	123
function BC_ImportALL	124
Типовые примеры	125
20 Эвристическая проверка системы.....	126
procedure SearchRootkit	126
procedure SearchKeylogger	126
procedure ExecuteSysChkEV	126
procedure ExecuteSysChkIPU	126
procedure ExecuteSysCheck	127
procedure ExecuteSysCheckEx	127
function ExecuteWizard	128
21 Эвристическая чистка системы.....	129
function ExecuteSysClean	129
procedure SysCleanAddFile	130
procedure SysCleanDelFilesList	130
22 Восстановление системы и стандартные скрипты.....	130
function ExecuteRepair	130
function ExecuteStdScr	131
23 Завершение работы и перезагрузка.....	131
procedure ShutdownWindows	131
procedure RebootWindows	131
24 Использование AV анализатора в скрипте.....	132
function CheckFile	132
function GetLastCheckTxt	132
25 Работа с CLSID и BHO.....	132
function CLSIDExists	132
function CLSIDFileExists	132
function CLSIDFileName	133
procedure DelCLSID	133
function BHOExists	134
procedure DelBHO	134
26 Работа со службами и драйверами.....	134
function ServiceExists	134
function ServiceAndFileExists	134
function StartService	134
function StopService	134
function DeleteService	135
function GetServiceFile	135
function GetServiceStart	135
function SetServiceStart	135
function GetServiceName	136
27 Работа с SPI/LSP.....	136
procedure DelSPIByFileName	136

procedure CheckSPI	137
procedure AutoFixSPI	137
28 Работа с автозапуском и Winlogon.....	137
procedure DelWinlogonNotifyByFileName	137
procedure DelWinlogonNotifyByKeyName	138
function DelAutorunByFileName	138
29 Работа с реестром.....	138
function RegKeyExists	138
procedure RegKeyDel	138
procedure RegKeyCreate	139
function RegKeyParamExists	139
procedure RegKeyParamDel	139
function RegKeyStrParamRead	139
function RegKeyIntParamRead	139
procedure RegKeyStrParamWrite	139
procedure RegKeyIntParamWrite	139
procedure RegKeyBinParamWrite	140
function RegSearch	140
function ExpRegKey	140
function ExpRegKeyEx	141
function BackupRegKey	141
function RegKeyEnumVal	142
function RegKeyEnumKey	142
Имена разделов	142
30 Работа с INI файлами.....	143
function INIStrParamRead	143
function INIStrParamWrite	143
function INISectionExists	143
function INIEraseSection	143
function INIEraseParam	143
31 Работа с файлом Hosts.....	144
function GetHostsFileName	144
function ClearHostsFile	144
32 Работа с файлами и папками.....	144
function GetCurrentDirectory	144
function GetSystemDisk	144
function GetTempDirectoryPath	145
function NormalDir	145
function NormalFileName	145
function ExtractFileName	145
function ExtractFilePath	146
function ExtractFileExt	146
function DirectoryExists	146
procedure CreateDirectory	147
procedure DeleteDirectory	147
function FileExists	147
function GetFileSize	147
function CalcFileMD5	148
function DeleteFile	148
function DeleteFileMask	148
function CopyFile	149
function RenameFile	149

function GetDriveType	149
Макросы, допустимые в именах файлов	150
33 Работа с текстовыми файлами и списками строк.....	150
Класс TStringList	150
Примеры	151
34 Поиск файла и папок.....	154
Класс TFileSearch	154
Примеры поиска	155
35 Работа с процессами	156
function ExecuteFile	156
function RefreshProcessList	157
function GetProcessCount	157
function GetProcessName	157
function GetProcessPID	157
function TerminateProcess	157
function TerminateProcessByName	158
Пример	158
36 Сигнатурный анализатор файла.....	158
function LoadFileToBuffer	158
function LoadFileToBufferEx	159
function FreeBuffer	159
function GetBufferSize	159
function GetBufferByte	159
function GetBufferWord	160
function GetBufferDWord	160
function GetBufferStr	160
function SearchSign	160
Пример реализации сигнатурного искателя	161
37 Работа с результатами исследования системы.....	162
function SC_INIT	162
function SC_FREE	163
function SC_SelectNode	164
function SC_GetItemsCount	164
function SC_GetParamVal	165
function SC_GetTagName	166
function SC_SearchItem	166
38 Функции запроса статистики.....	167
function GetSuspCount	167
function GetDetectedCount	168
39 Обработка параметров командной строки.....	168
function GetParamCount	168
function GetParamStr	168
function GetParamByName	168
Пример обработки параметров	169
40 Работа со строками.....	169
function StringReplace	169
function Pos	169
procedure Insert	170
procedure Delete	170
function Copy	170
function Length	170

function Trim	171
function UpperCase	171
function LowerCase	171
function Chr	171
function Ord	171
Преобразования в строку	171
function IntToStr.....	171
function FloatToStr.....	171
function DateToStr.....	171
function TimeToStr.....	172
function DateTimeToStr.....	172
41 Передача оповещения администратору.....	172
function SendNetMessage	172
function SendSyslogMessage	173
function SendEmailMessage	173
42 Типовые примеры.....	174
Параметры запуска	174
Сканирование компьютера	175
Блокировка сканирования по условию	175
Сканирование и автокарантин	175
Лечение заданных папок по условию	176
Автокарантин и сбор подозрительных файлов с ПК	176
Поиск файла на диске	177
Заготовка скрипта для сканирования сети	178
Обновление баз с протоколированием	179
Пример удаления троянской программы с известным именем	180
Поиск подозрительных объектов по именам	181
Блокировка повторного запуска скрипта в течение дня	183
Сканирование и отправка результатов по почте	183
Index	185

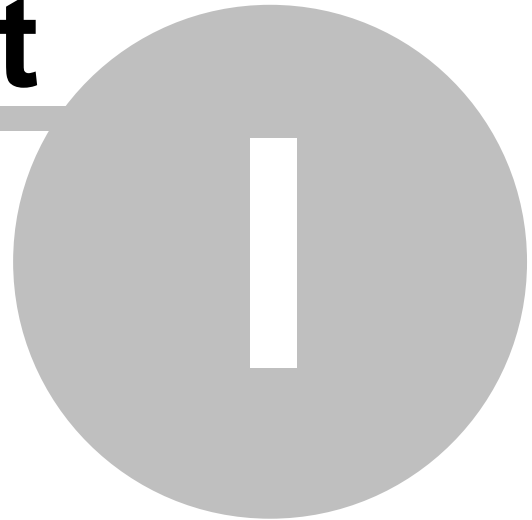
Foreword

This is just another title page
placed between table of contents
and topics

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



1 Общие сведения

1.1 О программе

Антивирусная утилита AVZ является инструментом для исследования и восстановления системы, и предназначена для автоматического или ручного поиска и удаления:

- SpyWare, AdvWare программ и модулей (это одно из основных назначений утилиты);
- Руткитов и вредоносных программ, маскирующих свои процессы
- Сетевых и почтовых червей;
- Троянских программ (включая все их разновидности, в частности Trojan-PSW, Trojan-Downloader, Trojan-Spy) и Backdoor (программ для скрытного удаленного управления компьютером);
- Троянских программ-звонилки (Dialer, Trojan.Dialer, Porn-Dialer);
- Клавиатурных шпионов и прочих программ, которые могут применяться для слежения за пользователем;

Утилита AVZ зарегистрирована в Российском Отраслевом Фонде Алгоритмов и Программ, номер свидетельства ОФАП 6427, код ЕСПД .52038851.00116-01.



Утилита является прямым аналогом программ Trojan Hunter и LavaSoft Ad-aware 6. Первичной задачей программы является удаление AdWare, SpyWare и троянских программ.

Сразу следует отметить, что программы категорий SpyWare, AdWare по определению не являются вирусами или троянскими программами. Они шпионят за пользователем и загружают информацию и программный код на пораженный компьютер в основном из маркетинговых соображений (т.е. передаваемая информация не содержит критических данных – паролей, номеров кредитных карт и т.п., а загружаемая информация является рекламой или обновлениями). Однако очень часто грань между SpyWare и троянской программой достаточно условна и

точная классификация затруднительна. Моя методика классификации и критерии описаны в данной справочной системе. Дополнительную информацию о вредоносных программах можно найти в моей книге ["Rootkits, SpyWare/AdWare, Keyloggers & BackDoors. Обнаружение и защита"](#)

Особенностью программы AVZ является возможность настройки реакции программы на каждую из категорий вредоносных программы – например, можно задать режим уничтожения найденных вирусов и троянских программ, но заблокировать удаление AdWare.

Другой особенностью AVZ являются многочисленные эвристические проверки системы, не основанные на механизме поиска по сигнатурам – это поиск RootKit, клавиатурных шпионов, различных Backdoor по базе типовых портов TCP/UDP. Подобные методы поиска позволяют находить новые разновидности вредоносных программ.

Кроме типового для программ данного класса поиска файлов по сигнатурам в AVZ встроена база с цифровыми подписями десятков тысяч системных файлов. Применение данной базы позволяет уменьшить количество ложных срабатываний эвристики и позволяет решать ряд задач. В частности, в системе поиска файлов есть фильтр для исключения известных файлов из результатов поиска, в диспетчере запущенных процессов и настроек SPI производится цветовое выделение известных процессов, при добавлении файлов в карантин производится блокировка добавления известных AVZ безопасных файлов.

Как показала моя практика, очень часто программа типа SpyWare может быть классифицирована как AdWare и наоборот (причины просты – целью шпионажа в большинстве случаев является целевая реклама). Для таких случаев в своей классификации я ввел обобщающую категорию Spy, которая грубо может трактоваться как AdWare+SpyWare. Термин Spy переводится как «шпион», «тайный агент», «следить», «подглядывать», «совать нос в чужие дела». Этот термин достаточно хорошо подходит к программам подобного класса.

Ограничения программы:

1. Т.к. утилита направлена в первую очередь на борьбу с SpyWare и AdWare модулями, и в настоящий момент она не поддерживает проверку архивов некоторых типов, PE упаковщиков и документов. Для борьбы со SpyWare в этом просто нет надобности. Тем не менее, утилита совершенствуется и появление подобных функций планируется;
2. Утилита не лечит программы, зараженные компьютерными вирусами. Для качественного и корректного лечения зараженной программы необходимы специализированные антивирусы (например, антивирус Касперского, DrWeb, Norton Antivirus, Panda и т.п.). Делать нечто похожее на них (изобретая тем самым велосипед) у меня нет никакого желания, тем более что вирусы такого рода встречаются все реже.

1.2 Лицензионное соглашение

1. Все авторские права на программу принадлежат ее автору - Зайцеву Олегу (<http://z-oleg.com/secur>);
2. Данная версия распространяется бесплатно и предназначена для некоммерческого применения;
3. Автор оставляет за собой право отменить действие данной лицензии для любой из следующих версий AVZ;
4. AVZ распространяется свободно, при условии, что настоящий дистрибутив не изменен. Ни одно частное лицо или организация не может брать плату за распространение AVZ без письменного разрешения автора.
5. Утилита AVZ поставляется по принципу "AS IS" ("как есть"). Никакие гарантии не прилагаются и не предусматриваются. Вы используете данное программное обеспечение на свой страх и риск. Автор программы не будет отвечать ни за какие потери или искажения данных, нарушения работоспособности других программ и системы, а также за любую упущенную выгоду в процессе использования или неправильного использования данного программного продукта. По умолчанию утилита работает в аналитическом режиме и ничего не изменяет в проверяемой системе. Решение о включении режима лечения принимается самим пользователем;
6. Вы не можете использовать, копировать, эмулировать, создавать новые версии, сдавать в наем или аренду, продавать, изменять, декомпилировать, дизассемблировать, изучать код программы другими способами и использовать программу и ее компоненты иначе, чем определено настоящим лицензионным соглашением. Любое такое нелегальное использование означает автоматическое и немедленное прекращение действия настоящего соглашения и может преследоваться по закону.
7. Все права, не предоставленные здесь явно, сохраняются за автором.
8. Установка и использование AVZ означает, что вы понимаете положения настоящего лицензионного соглашения и согласны с ними.
9. Если почему-либо вы не согласны с этим лицензионным соглашением, Вам необходимо удалить файлы дистрибутива AVZ с ваших устройств хранения информации и прекратить использование утилиты.

1.3 Инсталляция и системные требования

Антивирусная утилита AVZ не требует инсталляции на ПК (у текущей версии в комплекте нет инсталлятора и деинсталлятора - в них просто нет надобности). Естественно, не требуется и деинсталляция.

Единственно требование для нормального функционирования утилиты - размещение баз с описанием вирусов (они имеют расширение *.avz) в подкаталоге BASE утилиты. Антивирусная утилита AVZ при выключенном лечении (это режим по умолчанию) в процессе работы ничего не изменяет в настройках системы и реестре, не устанавливает и не удаляет классы и библиотеки, поэтому проверка компьютера при выключенном лечении не оказывает на ПК и операционную систему никакого воздействия. Единственным изменением, вносимым на время работы в конфигурацию компьютера, является загрузка драйвера AVZ.SYS, который на время проверки копируется в каталог Driver и удаляется после завершения проверки. Этот драйвер устанавливается и загружается только при включенном режиме поиска руткитов.

Работа утилиты в режиме лечения вносит изменения - в этом случае с ПК удаляются найденные вредоносные программы и (в ряде случаев) сопутствующая им информация (ключи реестра, ярлыки и т.п.)

Антивирусная утилита AVZ не выдвигает особых системных требований - ее работоспособность проверена на сотнях компьютеров с операционной системой W9x,

Windows NT, 2000 Professional и Server, XP Home Edition и XP Professional (SP1, SP2), Windows 2003 (SP1), Windows Vista. Технологии AVZPM, AVZGuard, BootCleaner не поддерживаются в Win 9x и в 64-bit версиях операционных систем XP/Vista.

Антивирусная утилита AVZ не обменивается информацией с сетью и Интернет (не передает и не принимает данные, не прослушивает порты), поэтому для ее эксплуатации не требуется дополнительная настройка Firewall. Исключением является обновление AV баз, которое запускается по команде пользователя - для обновления баз AVZ обращается к указанному в настройке сайту по порту 80 с использованием протокола HTTP.

В случае необходимости можно произвести принудительное удаление ключей реестра и файлов, которые AVZ мог создать на диске, что эквивалентно его деинсталляции - для этого следует воспользоваться микропрограммой номер 6 в разделе [Стандартные скрипты](#) ⁶³ или выполнить скрипт:

```
begin
  ExecuteStdScr(6);
end.
```

1.4 Структура папок программы

Утилита AVZ может устанавливаться в любую папку, но для определенности будем считать, что это папка с именем AVZ.

Структура папок:

AVZ

Base Базы AVZ с описаниями вредоносных программ и методик их лечения. Загруженные обновления необходимо помещать именно в эту папку

Backup Папка с резервными копиями критических настроек системы. Резервные копии создаются автоматически в процессе лечения и восстановления системы или выполняются по команде пользователя. Папка содержит подкаталоги с именами вида «ГГГГ-ММ-ДД», где ГГГГ – год, ММ – месяц, ДД – день.

Infected Папка с копиями удаленных вредоносных объектов

Quarantine Карантин – папка с копиями подозрительных объектов

Папки Infected и Quarantine внутри содержат подкаталоги с именами вида «ГГГГ-ММ-ДД», где ГГГГ – год, ММ – месяц, ДД – день. Такая организация существенно упрощает последующую работу с этими папками (в частности, упрощается их очистка и просмотр).

На заметку:

Папки Infected и Quarantine могут быть перемещены в любой каталог с помощью ключа командной строки [QuarantineBaseFolder](#) ⁹⁶

1.5 Техническая поддержка

Несмотря на то, что утилита AVZ является бесплатной программой, у нее имеется техническая поддержка по электронной почте:

- avz@z-oleg.com - по этому адресу Вы можете задать вопросы по работе программы, сообщить

об ошибках или прислать предложения по усовершенствованию программы и методик ее работы;

- newvirus@z-oleg.com - на этот адрес следует присылать недетектируемые AVZ вирусы / протоколы AVZ / подозрительные файлы для проверки (отправка подозрительного объекта должна производиться в архиве с паролем).
- VirusInfo.Info - в этой конференции вирусологов есть раздел "Помогите", в котором можно описать свою проблему и приложить протокол исследования системы AVZ (подробности запроса помощи описаны в правилах данной конференции). AVZ в частности обучается по найденным в данной конференции вредоносным программам, анализ протоколов и формирование рекомендаций происходит достаточно оперативно - в большинстве случаев проблему удастся решить в день обращения за помощью. С помощью сервиса "[Прислать чистые файлы](#)" данного сайта можно прислать архив с чистыми файлами для базы безопасных AVZ, а в разделе "[Beta тестирование](#)" идет обсуждение новых версий AVZ и предложений по его доработке.
- По адресу <http://www.z-oleg.com/secur/avz/report.php> расположена форма, при помощи которой можно сообщить об ошибке, возникающей в процессе работы с AVZ. Следует понимать, что обращение через WEB форму анонимно и рекомендуется применять ее для рапортов об ошибках. Нет смысла задавать через эту форму вопросы, не указав параметры обратной связи
- По адресу <http://www.z-oleg.com/secur/avz/uploadvir.php> расположен сервис, предназначенный для отправки подозрительных файлов для анализа и вредоносных программ, которые не детектируются AVZ для их включения в базы

При написании письма желательно четко сформулировать тему письма, кратко и точно изложить суть проблемы, пожелания или вопроса. От этого будет зависеть время реакции на письмо (согласитесь, заголовок "Подозрительный объект для проверки" гораздо информативнее, чем "Спасите!", "Помогите", "SOS" и т.п.). Если Вы присылаете подозрительный объект, о котором у Вас есть информация (например, AVZ заподозрил некую DLL как клавиатурный шпион, а Вы знаете, что она является компонентом программы X) - пожалуйста, укажите ее - это упростит и ускорит анализ.

При написании письма желательно придерживаться следующих правил:

1. Создавать письма с информативными заголовками. Это упростит анализ и ускорит время реакции. Например "Подозрительный файл (срабатывание антикейлоггера)" или "Подозрение на RootKit" - куда информативнее, чем заголовки типа "SOS", "Спасите", "Помогите", "Караул", "от Васи Пупкина" и т.п.

2. В тексте письма желательно указать:

2.1 Версию операционной системы

2.2 Краткую формулировку проблемы (т.е. что и где обнаружилось, где, какие симптомы)

2.3 Желательно приложить протокол AVZ и результаты исследования системы (Файл/Исследование системы)

3. В случае, если подозрительные файлы в архиве занимают большой объем (более 1-2 МБ), то необходимо предварительно прислать письмо с описанием проблемы, протоколами и указанием размера подозрительных файлов.

1.6 Назначение программы

Назначением программы является:

- AV база. Позволяет диагностировать известные AVZ malware-программы и удалить их. Удаление предполагает автоматическую зачистку критичных для работы системы следов вредоносной программы в реестре и INI файлах. В этом плане удобно применять AVZ для экспресс-зачистки зараженного компьютера перед применением "тяжелой артиллерии" - установки мощного антивирусного пакета и проведение сканирования с его помощью. Сканер может проверять архивы распространенных типов, файлы электронной почты, потоки NTFS.

AV сканер может интегрироваться с TheBat при помощи плагина. Базы AVZ обновляются ежедневно.

- Оперативное [автоматическое исследование компьютера](#)^[57] с формированием протокола в формате HTML. В ходе исследования автоматические отфильтровываются файлы, прошедшие проверку по базе безопасных AVZ и каталогу безопасности Microsoft, что существенно уменьшает размер протоколов. Данный режим удобен для оперативного изучения подозрительного компьютера администратором, а так-же для удаленного исследования системы. Возможность запуска сканирования системы и карантина из скрипта позволяет полностью автоматизировать данную операцию, сведя действия пользователя на месте к запуску bat файла;
- [Автоматический карантин файлов](#)^[35], не имеющих ЭЦП Microsoft и не описанных в базе безопасных AVZ для их последующего изучения вручную или антивирусными программами. Данный режим удобен для быстрого сбора всех неопознанных файлов для их последующего анализа. Кроме того, в AVZ предусмотрен карантин по списку и команды добавления в карантин в скриптах, что упрощает дистанционный сбор подозрительных файлов с проверяемых компьютеров;
- Выполнение поиска руткитов и иных перехватчиков API с функцией поиска скрытых процессов. Кроме анализа перехватов AVZ обладает функцией нейтрализации перехватчиков UserMode и KernelMode;
- [Выполнение восстановления системы](#)^[59]. AVZ содержит микропрограммы для автоматического исправления типовых повреждений настроек Internet Explorer и проводника, сброса настроек рабочего стола, нейтрализация устанавливаемых троянскими программами правил Policy. Данные операции не производятся антивирусами и очень часто после удаления троянской программы или SpyWare нормальная работа системы не восстанавливается
- Автоматическая проверка настроек SPI/LSP и исправление ошибок в автоматическом режиме. Позволяет устранить большинство типовых проблем с LSP, возникающих после удаления некоторых AdWare. В случае невозможности восстановления настроек предусмотрено их полное пересоздание;
- [Поиск файлов на диске](#)^[40]. Поиск защищен антируткитом AVZ и обладает рядом полезных для поиска вирусов/троянов функций, в частности исключение из списка найденных файлов, прошедших проверку по базе безопасных AVZ и каталогу безопасности Microsoft, что позволяет существенно сузить область поиска;
- [Скриптовый язык](#)^[105], позволяющий управлять AVZ. Скрипты позволяют применять AVZ в корпоративной сети - в этом случае AVZ может запускаться из logon-скрипта или автозапуска, и работать по скрипту, созданному администратором. Кроме того, скрипт позволяют автоматизировать большинство операций, выполняемых AVZ.
- [Встроенный ревизор диска](#)^[78]. Ревизор создает базы, содержащие информацию о файлах в соответствии с настройками пользователя (задаются каталоги, маски поиска). Данные базы могут применяться для отслеживания изменений на диске.
- [Анализатор запущенных процессов](#)^[45], позволяющий в режиме максимальной эвристики производить поиск подозрительных объектов
- Система [AVZGuard](#)^[68], позволяющая защитить AVZ и любые указанные им приложения от действующих в системе вредоносных программ и ограничит влияние вредоносных программ на систему.
- Система прямого доступа к диску для работы с заблокированными файлами. Работает на FAT16/FAT32/NTFS, поддерживается на всех операционных системах линейки NT, позволяет сканеру анализировать заблокированные файлы и помещать их в карантин.
- Драйвер мониторинга процессов и драйверов [AVZPM](#)^[72]. Предназначен для отслеживания запуска и останова процессов и загрузки/выгрузки драйверов для поиска маскирующихся драйверов и обнаружения искажений в описывающих процессы и драйверы структурах, создаваемых DKOM руткитами.
- Драйвер [Boot Cleaner](#)^[75]. Предназначен для выполнения чистки системы (удаление файлов, драйверов и служб, ключей реестра) из KernelMode. Операция чистки может выполняться как в процессе перезагрузки компьютера, так и в ходе лечения.
- Поиск потенциальных уязвимостей. Предназначен для поиска некорректных настроек ПК, которые могут отрицательно влиять на безопасность

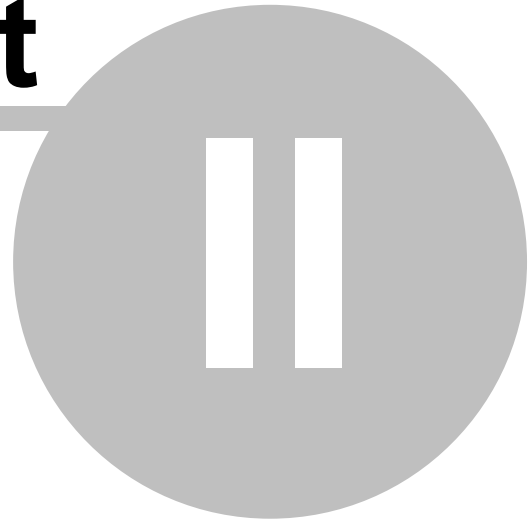
- [Резервное копирование](#)⁶⁴. Предназначено для выполнения резервных копий критических настроек системы. Резервное копирование выполняется по команде пользователя или автоматически в ходе лечения и восстановления системы.

Таким образом **AVZ позиционируется как интерактивный инструмент, предназначенный для полуавтоматического изучения ПК с целью поиска и уничтожения вредоносных программ. Одной из основных особенностей AVZ является поддержка внешних скриптов, под управлением которых возможен автоматический анализ, чистка компьютера и карантин файлов. Система команд скриптового языка и примеры описаны справке**

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



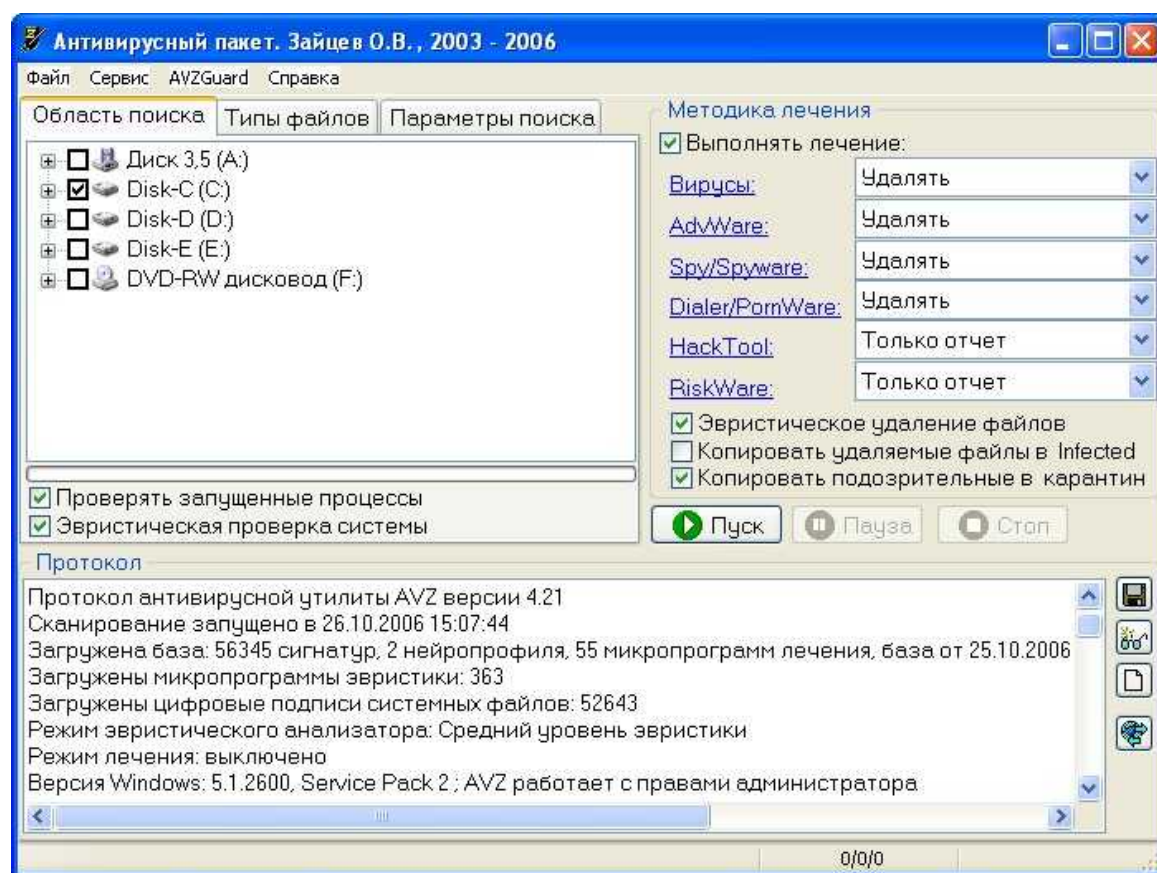
2 Работа с программой

2.1 Главное окно программы

Перед началом работы с программой на время проверки и лечения рекомендуется отключить средства фоновых мониторингов изменений реестра и антивирусные мониторы. Данное условие не является обязательным, однако его несоблюдение может привести к существенному увеличению длительности сканирования - несколько часов вместо нескольких минут. Причина этого состоит в том, что антивирусный монитор проверяет все открываемые файлы в момент их открытия.

Главное окно программы

Главное окно программы содержит все основные элементы управления - главное меню, настройки процесса поиска и лечения, окно просмотра протокола и строку статуса.



Кнопка "Пуск" позволяет запустить проверку выбранных дисков и каталогов.

Кнопка "Пауза" позволяет временно приостановить проверку файлов. Данная кнопка автоматически разблокируется при выполнении операций, для которых допустима приостановка. Возобновление сканирования производится повторным нажатием на кнопку "Пауза".

Кнопка "Стоп" позволяет в любой момент прервать процесс проверки.

На время проверки большинство элементов интерфейса AVZ блокируются.

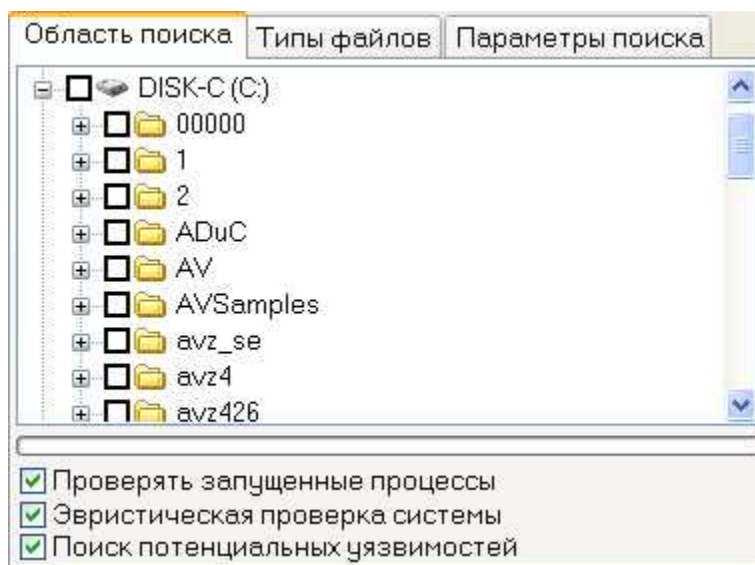
См. также:

[Закладка 'Область поиска'](#) ²³

[Закладка 'Типы файлов'](#) ²⁴
[Закладка "Параметры поиска"](#) ²⁶
[Группа "Параметры лечения"](#) ²⁸
[Протокол](#) ²⁹

2.1.1 Закладка 'Область поиска'

Закладка "Область поиска" предназначена для задания области поиска вредоносных программ. Она содержит древовидный список дисков и папок. Напротив каждого элемента списка имеется переключатель, который позволяет включить любой из элементов списка в поиск или исключить из него.



При нажатии над списком папок правой кнопки мыши выводится меню, позволяющее:

- Отметить все жесткие диски (HDD)
- Отметить все CD-ROM и DVD-ROM диски
- Отметить все дисководы и Flash диски
- Обновить список дисков. Обновление списка возможно по нажатию клавиши F5
- Добавить папку. Выполнение этого пункта меню приводит к выводу диалогового окна с запросом имени папки. Данная функция удобна для быстрой отметки каталога в случае, если имеется полный путь к нему.
- Проверять только файлы в указанной папке (без подкаталогов). Отмечает текущую папку в особом режиме - в нем сканируются только файлы, размещенные в данной папке - без сканирования вложенных каталогов.

Переключатели напротив каталогов определяют режим сканирования, возможны варианты:

- ☐ - файлы и подкаталоги данного каталога не сканируются
- ☒ - сканируются все файлы и подкаталоги данного каталога
- ☒ - сканируются все файлы данного каталога и часть его подкаталогов (эта ситуация возникает, если подкаталогов несколько и отмечена часть из них)
- ☐ - сканируются только файлы данного подкаталога, подкаталоги не сканируются. Включить этот режим сканирования каталога можно через контекстное меню, вызываемое по правой кнопки мыши - пункт "Проверять только файлы в указанной папке (без подкаталогов)".

Переключатель "Проверять запущенные процессы"

Переключатель "Проверять запущенные процессы" позволяет включить проверку всех запущенных процессов и загруженных библиотек перед началом проверки системы. Этот переключатель включен по умолчанию, и отключать его не рекомендуется. Если разрешено лечение, то при обнаружении процесса вредоносной программы AVZ принудительно завершает процесс перед удалением файла.

Переключатель "Эвристическая проверка системы"

Переключатель "Эвристическая проверка системы" включает дополнительную эвристическую проверку системы. Особенность этой проверки состоит в анализе запущенных процессов, реестра и диска для поиска неизвестных для AVZ разновидностей вредоносных программ по их характерным признакам. Если этот режим включен и AVZ обнаружит подозрительные объекты, то он сообщит об этом в протоколе. Включение эвристической проверки увеличивает общее время сканирования системы. Основой эвристической проверки системы являются так называемые микропрограммы эвристики, которые хранятся в базах данных и постоянно обновляются.

Переключатель "Поиск потенциальных уязвимостей"

Переключатель "Поиск потенциальных уязвимостей" включает подсистему, которая ищет потенциальные проблемы в области безопасности. Следует понимать, что выдаваемые данной системой сообщения носят рекомендательный характер из раздела "На это стоит обратить внимание". Если после выполнения поиска потенциальных уязвимостей выполнить исследование системы, то в средство формирования скрипта протокола исследования системы будут добавлены пункты для создания скрипта устраняющего найденные проблемы (при условии, что их можно исправить скриптом).

На закладке "Область поиска" также размещается прогресс-индикатор, показывающий примерное состояние процесса сканирования.

2.1.2 Закладка 'Типы файлов'

Закладка "Типы файлов" позволяет настроить типы проверяемых файлов.

Переключатель "Типы файлов".

Данный переключатель позволяет выбрать типы файлов, проверяемые AVZ. По умолчанию выбран пункт "Потенциально опасные файлы" - в этом случае AVZ проверяет только файлы с определенными расширениями (EXE, DLL, OCX, SYS, CAB, INF ...) - т.е. исполняемые файлы или файлы, которые могут содержать данные для установки вредоносных программ или опасные скрипты.

Выбор пункта "Все файлы" приводит к проверке абсолютно всех файлов, независимо от их расширения. При сканировании диска в этом режиме проверяется большее количество файлов, в результате скорость проверки замедляется. В этом режиме блокируется переключатель "Включая файлы по маске", т.к. в данном случае он теряет смысл. Выбор пункта "Файлы по маскам пользователя" приводит к тому, что поиск ведется только по маскам, заданным пользователем в полях "Включая файлы по маске" и "Исключая файлы по маске".

На заметку: Если на закладке "Область поиска" включен переключатель "Проверять запущенные процессы", то запущенные программы и загруженные библиотеки проверяются в любом случае, независимо от их расширения и настройки в радиогруппе "Типы файлов".

Переключатель "Включая файлы по маске"

Переключатель "Включая файлы по маске:" позволяет включить в поиск файлы, соответствующие заданной пользователем маске. Этот переключатель теряет смысл в режиме "Все файлы" и при выборе этого режима автоматически выключается и становится недоступным.

Переключатель "Исключая файлы по маске"

Переключатель "Исключая файлы по маске:" позволяет исключить из проверки файлы, соответствующие заданной пользователем маске. Исключение файлов по маске может использоваться совместно с включением по маске или режимов "Все файлы". Указание одинаковых элементов в списке включения и исключения не является ошибкой, исключение имеет приоритет над включением - например, при указании включения в сканирование файлов *.vlg и исключения *.vlg приоритетно исключение. Более того, исключение по более общей маске доминирует над включением, например при включении trojan*.exe и исключении *.exe файл с именем trojan.exe будет исключен из проверки

На заметку: Поля с маской могут включать несколько масок, разделенных запятой, пробелом или символом ";". В маске могут присутствовать символы "?" (любой символ в позиции знака ?) и * (любые символы).

* * - поиск всех файлов

*.exe - поиск файлов с расширением exe

dialer.exe - поиск файла с конкретным именем "dialer.exe"

dialer.exe, *.dll, trojan*.sys - поиск файлов с именем dialer.exe или файлов с любым именем и расширением DLL или файлов с именем Trojan<любые символы>.sys

*.exe, *.dll, *.sys, *.osx - поиск файлов с указанными расширениями

Переключатель "Отчет о чистых объектах"

Переключатель "Отчет о чистых объектах" позволяет включить в протокол список файлов, в которые проверены AVZ и, по его мнению, не являются вирусами или вредоносными программами.

Переключатель "Проверять чистые объекты по базе безопасных"

Данный переключатель доступен только при включенном переключателе "Отчет о чистых объектах". При его включении каждый файл, не являющийся по мнению AVZ вирусом или вредоносной программой проверяется по базе безопасных файлов и результаты этой проверки заносится в протокол. Данная операция замедляет сканирование и полезна для системных администраторов и вирусологов, проводящих сортировку файлов перед анализом.

Переключатель "Проверять потоки NTFS"

Данный переключатель включает проверку потоков в NTFS файлах. Потоки не видны при просмотре диска при помощи файловых менеджеров и проводника, поэтому многие вредоносные программы маскируют свои исполняемые файлы, размещая их в потоке. AVZ может проверять потоки и удалять из них найденные вредоносные объекты. Переключатель

включен по умолчанию и его отключение не рекомендуется

Переключатель "Проверять архивы"

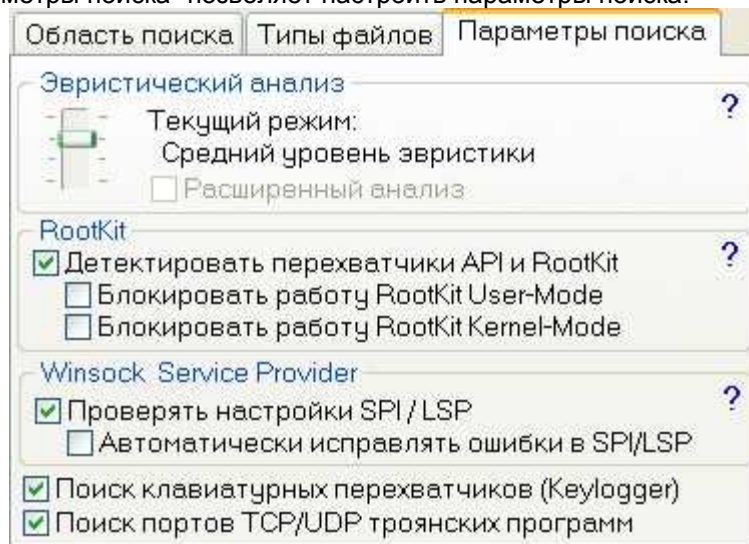
Управляет проверкой архивов. Переключатель включен по умолчанию. Под архивом понимаются так-же СНМ файлы, МНТ файлы, файлы электронной почты и прочие структуры, которые могут содержать внутри себя другие файлы.

Переключатель "Не проверять архивы более xx мб."

Данный переключатель активен, если включена проверка архивов. По умолчанию он включен и архивы размером более 10 мб не проверяются. Пороговый размер задается пользователем, но он не может быть менее 1 мб.

2.1.3 Закладка 'Параметры поиска'

Закладка "Параметры поиска" позволяет настроить параметры поиска:



Группа "Эвристический анализ" содержит регулятор уровня срабатывания эвристики. AVZ поддерживает 4 уровня эвристического анализа:

- Эвристический анализ отключен. В этом случае обнаруживаются только известные AVZ вредоносные объекты при полном совпадении проверяемого объекта с описанием в базе;
- Минимальный уровень эвристики - в этом режиме AVZ выдает предупреждения при обнаружении объектов, очень похожих на вредоносные;
- Средний уровень эвристики - аналогичен минимальному, но порог срабатывания ниже. Это рекомендуемый режим работы, в котором уровень ложных срабатываний обычно не превышает 10-15%;
- Максимальный уровень эвристики - этот режим часто называют "параноидальным", так как предупреждения о подозрении на вирус выдаются даже при обнаружении отдаленного сходства проверяемого объекта с вредоносным. В этом режиме возможны многочисленные ложные срабатывания. При установке максимального уровня эвристики становится доступным переключатель "Расширенный анализ". Включение данного переключателя задействует дополнительные проверки, которые позволяют обнаруживать подозрительные объекты по второстепенным признакам типа двойного расширения, наличия в имени большого количества пробелов перед расширением и т.п. Расшифровка основных сообщений эвристического анализатора приведена в разделе [Эвристический анализатор AVZ](#);

Группа "RootKit"

Группа "RootKit" содержит настройки системы обнаружения и блокирования [RootKit](#)^[87]. Переключатель "Детектировать RootKit" позволяет включить детектор [RootKit](#)^[87], который пытается обнаружить присутствие в системе [RootKit](#)^[87] по характерным для него признакам. Следует отметить, что возможны своего рода ложные срабатывания - в качестве RootKit могут детектироваться системные утилиты, антивирусные мониторы, Firewall. Это нормальное явление, т.к. подобные программы и утилиты вмешиваются в работу других приложений и искажают работу стандартных API функций. Переключатель "Блокировать RootKit" доступен только при включенном переключателе "Детектировать RootKit" и позволяет включить систему активного противодействия RootKit. Следует отметить, что включение противодействия RootKit может привести к непредсказуемым последствиям и нужно быть готовым к зависанию программы AVZ и системы в целом. Поэтому перед активацией противодействия RootKit необходимо закрыть все программы, желательно отключиться от сети, и затем выгрузить антивирусный монитор и Firewall.

При включении блокирования RootKit автоматически включается система эвристического поиска процессов и файлов RootKit. Принцип действия системы основан на анализе системы до и после блокировки перехватчиков, что позволяет обнаружить маскируемые процессы, сервисы и драйверы.

Система [AVZGuard](#)^[68] и антируткит режима ядра являются взаимоисключающими, при включении AVZGuard нейтрализация руткитов в режиме ядра отключается.

На заметку:

В случае проверки системы с нейтрализацией Kernel RootKit необходимо перезагрузиться !! Это связано с тем, что нейтрализация перехватчиков может нарушить работу антивирусных мониторов, компонент Firewall и прочих программ, отвечающих за безопасность. Перезагрузка актуальна только в случае восстановления перехваченных функций - в этом случае AVZ указывает на необходимость перезагрузки в конце протокола.

Группа "Winsock Service Provider"

Группа "Winsock Service Provider" позволяет настроить работу системы проверки и настройки анализатора SPI (который более известен как LSP благодаря утилите LSP Fix). Настройки SPI модифицируют многие современные вирусы и SpyWare, типовые проявления повреждения настроек является появление сбоев при работе в Интернет. Переключатель "Проверять настройки SPI / LSP" позволяет включить анализатор настроек - при этом проверка производится автоматически и все найденные проблемы и ошибки выводятся в протокол. Установка переключателя "Автоматически исправлять ошибки в SPI/LSP" позволяет включить систему автоматического исправления найденных ошибок - исправление ошибок производится в автоматическом режиме и не требует от пользователя понимания тонкостей работы SPI. Начиная с версии 4.26 в случае обнаружения ошибок перед их автоматическим исправлением AVZ автоматически создает резервную копию настроек SPI в папке BackUp. Резервная копия является стандартным REG файлом и может быть импортирована в реестр в случае необходимости. Для устранения ошибок в ручном режиме рекомендуется использовать Менеджер Winsock SPI (LSP, NSP, TSP).

Переключатель "Поиск клавиатурных шпионов (Keylogger)"

Переключатель "Поиск клавиатурных шпионов (Keylogger)" позволяет включить детектор клавиатурных шпионов и троянских DLL (под троянской DLL понимается библиотека, внедряемая в адресное пространство запущенных процессов). Поиск ведется без применения механизма сигнатур, в результате чего результаты носят вероятностный характер. Для оценки степени похожести обнаруженных библиотек на клавиатурный шпион (KeyLogger) применяется нейросеть, распознающая характерные особенности клавиатурного перехватчика. Кроме того, AVZ производит поведенческий анализ всех заподозренных библиотек и выводит собранную информацию в протокол.

На заметку: Анализатор AVZ и нейросеть в состоянии найти подозрительные библиотеки и оценить степень их похожести на типовые перехватчики событий оконных событий, событий клавиатуры и мыши (т.н. hook). Однако анализатор не может отличить вредоносный перехватчик (предназначенный для слежения за пользователем) и перехватчик, выполняющий полезные действия.

Переключатель "Поиск портов TCP/UDP троянских программ"

Переключатель "Поиск портов TCP/UDP троянских программ" включает автоматический анализ списка открытых портов TCP и UDP. В составе AVZ имеется специальная база, в которой описаны несколько сотен портов, применяемых распространенными троянскими программами, Backdoor и вирусами. При обнаружении открытых портов, описанных данной базе, в протокол выводится предупреждение с указанием номера порта и списка вредоносных программ, применяющих данный порт

2.1.4 Группа 'Параметры лечения'

Группа "Параметры лечения" содержит настройки, определяющие действия AVZ при обнаружении вирусов.



Основным в этой группе является переключатель **"Удалять найденные вирусы"**. По умолчанию данный переключатель отключен и AVZ работает в диагностическом режиме - информация обо всех найденных вредоносных объектах заносится в протокол, но никаких действий над ними не производится (во время сканирования AVZ ничего не изменяет в реестре или на диске).

Переключатель "Удалять найденные вирусы"

При включении переключателя "Удалять найденные вирусы" AVZ переходит в режим лечения, при этом становятся доступными настройки для каждой из категорий вредоносных объектов. Для каждой категории объектов можно выбрать одно из двух действий - "удалять" (в этом случае производится удаление вредоносного объекта) и "Только отчет" - в этом случае лечение блокируется, но в протокол выводится сообщение об обнаруженном объекте.

Переключатель "Эвристическое удаление файлов"

Переключатель "Эвристическое удаление файлов" позволит включить "разумное" удаление вредоносных файлов. Оно состоит в том, что после удаления файла производится анализ системы и удаляются все обнаруженные ссылки на удаленный файл (ключи реестра для автозапуска, зарегистрированные в реестре классы, элементы автозапуска и WinLogon, элементы SPI/LSP и т.п.). В результате удаление файла получается максимально корректным и данный режим рекомендуется включить.

Переключатель "Копировать удаляемые файлы в Infected"

Переключатель "Копировать удаляемые файлы в Infected" включает режим создания резервных копий удаляемых файлов в папке Infected. Эта папка автоматически создается в каталоге AVZ. Для просмотра файлов в папке Infected в AVZ предусмотрен специальный просмотрщик, вызываемый из меню "Файл".


Переключатель "Копировать подозрительные в карантин"

Переключатель "Копировать подозрительные в карантин" позволяет включить режим копирования подозрительных объектов в карантин. Это очень удобно для последующей отправки файлов на анализ, т.к. AVZ сам делает копии подозрительных файлов и снабжает каждый файл информационным ярлыком, в котором указано местоположение файла на диске, дата и причина помещения в карантин. В окне просмотра карантина имеется возможность автоматического создания защищенного паролем архива для его пересылки по электронной почте.

2.1.5 Протокол

Утилита AVZ ведет два протокола:

- Текстовый протокол AVZ – он размещен в нижней части главного окна программы. Текстовый протокол содержит разнообразную информацию, описывающий ход проверки системы, обнаруженные ошибки, найденный вредоносные файлы и т.п.
- Протокол обнаруженных вредоносных файлов и подозрительных объектов

(вызывается при нажатии кнопки  на панели управления справа от текстового протокола). Данный протокол представляет собой таблицы вида «файл» - «описание». В описании файла указывается название вируса или описание причин, по которым файл признан подозрительным. Каждая строка таблицы снабжена переключателем, который позволяет отметить файл. Отмеченные файлы можно вручную скопировать в карантин или удалить.

Текстовый протокол может автоматически фильтроваться, управление режимом фильтрации производится при помощи ключа командной строки [MiniLog](#)^[96]. Кроме того, предусмотрено дублирование выводимой в протокол информации параллельным выводом в текстовый файл - управление этим режимом производится при помощи ключа [SpoolLog](#)^[96]

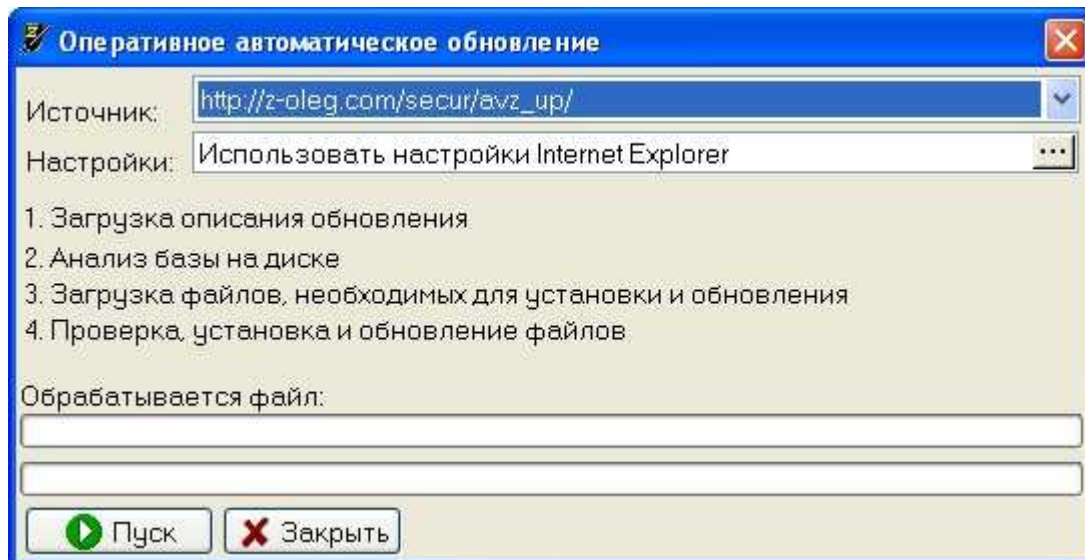
2.2 Обновление баз

Обновление баз в AVZ начиная с версии 4.00 может производиться через Интернет. Для выполнения обновления баз необходимо выполнить пункт меню "Файл/Обновление баз". Загрузка ведется с одного из двух сайтов - avz.virusinfo.info или z-oleg.com/secur, выбор сайта производится автоматически случайным образом, но есть возможность задать сайт вручную. Обновление баз состоит из трех операций:

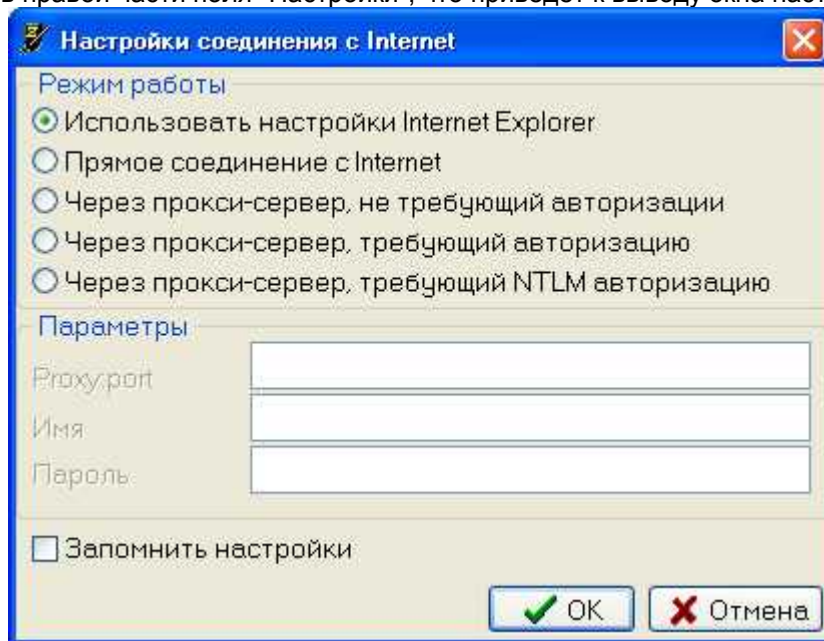
1. Загрузка файл с описанием актуального состояния AV баз
2. Сравнение базы локального компьютера с описанием
3. В случае обнаружения расхождений (ввиду появления в базе новых файлов, обновления существующих или обнаружения повреждения существующих) производится загрузка необходимых файлов и их установка

В случае сбоя в ходе загрузки файлов все успешно загруженные файлы остаются в папке Temp

и при повторной попытках обновления их повторная загрузка не производится.



По умолчанию для обращения в Интернет применяются настройки, заданные в Internet Explorer. Однако в ряде случаев (например, если не для работы с Интернет применяется IE или он не настроен) требуется вручную задать настройку обмена с Интернет. Для этого необходимо нажать кнопку в правой части поля "Настройки", что приведет к выводу окна настройки:



В данном окне можно задать один из четырех режимов работы:

- Использовать настройки Internet Explorer. В этом режиме все настройки обмена берутся из настроек Internet
- Прямое соединение с Internet. Обмен с Интернет ведется напрямую
- Через прокси-сервер, не требующий авторизации. Обмен с Интернет ведется через прокси-сервер, в этом режиме обязательно указать имя или IP адрес прокси-сервера в формате имя:порт, например my_proxy:3128 или 192.168.0.1:8080
- Через прокси-сервер, требующий авторизацию. Отличается от предыдущего режима тем, что прокси сервер требует передачи имени и пароля

Переключатель "запомнить настройки" позволяет сохранить введенные настройки для того, чтобы не выполнять перенастройку при каждом запуске AVZ. Настройки сохраняются в файле AVZ.INI и автоматически загружаются при следующем запуске AVZ.

На заметку: В случае применения AVZ в корпоративной сети возможно обновление баз с любого адреса и с любыми параметрами, заданными администратором. Для этого необходимо применить скрипт, содержащий команду [ExecuteAVUpdateEx](#)^[11]

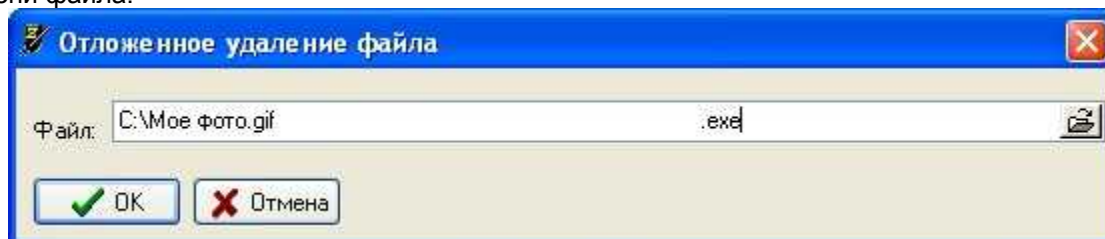
2.3 Отложенное удаление файла

Отложенное удаление файла является встроенной функцией AVZ и применяется автоматически сканером для удаления известных вредоносных программ. Отложенное удаление позволяет удалять файлы запущенных приложений, загруженных DLL или файлы, открытые с монопольным доступом.

AVZ выполняет отложенное удаление в два этапа:

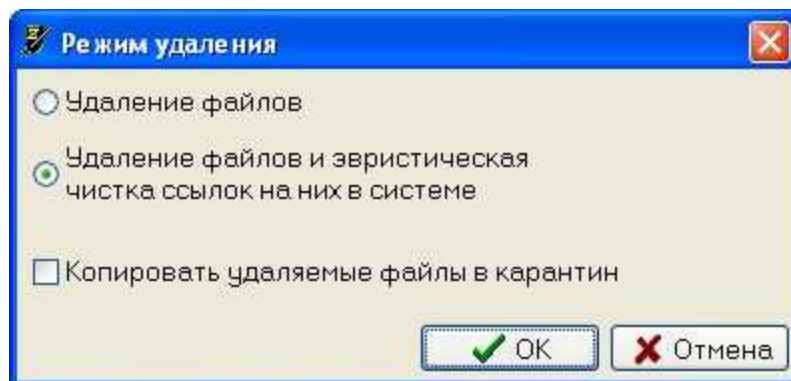
1. Делается попытка удаление файлы обычным способом. Если эта попытка успешна и файл удаляется, то отложенное удаление не применяется
2. Если удаление файла на шаге 1 было неуспешным, то файл регистрируется для отложенного удаление.

Отложенное удаление файла производится при помощи пункта меню "Файл\Отложенное удаление файла". После выполнения этого пункта меню выводится диалоговое окно для ввода имени файла.



Имя файла можно ввести вручную, скопировать через буфер обмена или выбрать в диалоговом окне, которое вызывается при нажатии кнопки в правой части поля для ввода имени файла. После ввода имени файла необходимо нажать ОК для его удаления или кнопку "Отмена" для закрытия окна без удаления файла.

В случае нажатия кнопки "ОК" выводится диалоговое окно с настройками параметров удаления файла.



Поддерживается два режима удаления:

1. Обычное удаление указанного файла (производится при выборе опции "Удаление файлов"). Этот режим не рекомендуется применять
2. Удаление файла и эвристическая чистка ссылок на файл в системе. В этом режиме кроме удаления файла производится автоматическая зачистка автозапуска и ряда ключей реестра, в

которых мог быть зарегистрирован удаляемый файл (Winlogon, регистрация CLSID, расширения Internet Explorer и проводника и т.п.). Подобная очистка производится автоматически и делает удаление более корректным.

Если включить переключатель "Копировать удаляемые файлы в карантин", то перед удалением файла в карантине будет создана его копия.

2.4 Профили настроек

AVZ позволяет сохранять настройки в виде именованных профилей. Профили являются текстовыми и содержат параметры, идентичные [параметрам командной строки](#)^[93]. После сохранения профили можно редактировать вручную.

Для сохранения текущих настроек в виде профиля необходимо вызвать пункт меню "Файл\Сохранить профиль настроек". В открывшемся диалоговом окне сохранения файла необходимо задать имя файла и местоположение. Расширение файла не играет роли, но по умолчанию у профилей используется расширение prf.

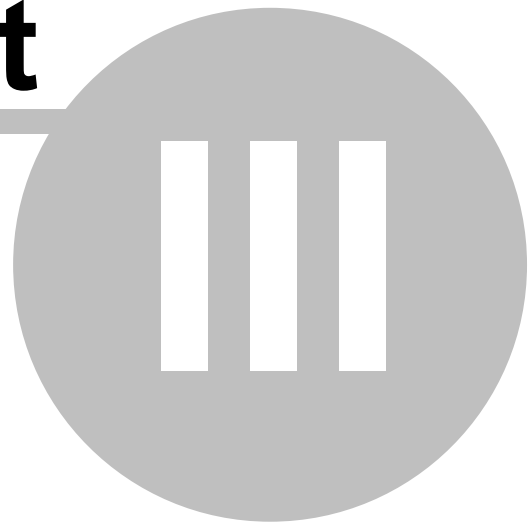
Загрузка профиля производится при помощи пункта меню "Файл\Загрузить профиль настроек". При выполнении этого пункта меню выводится диалоговое окно открытия файла, в котором необходимо выбрать нужный профиль.

У AVZ предусмотрена поддержка *профиля по умолчанию*. Этот профиль автоматически загружается при старте AVZ и должен располагаться в рабочей папке AVZ под именем "avz.prf".

Top Level Intro

This page is printed before a new
top-level chapter starts

Part

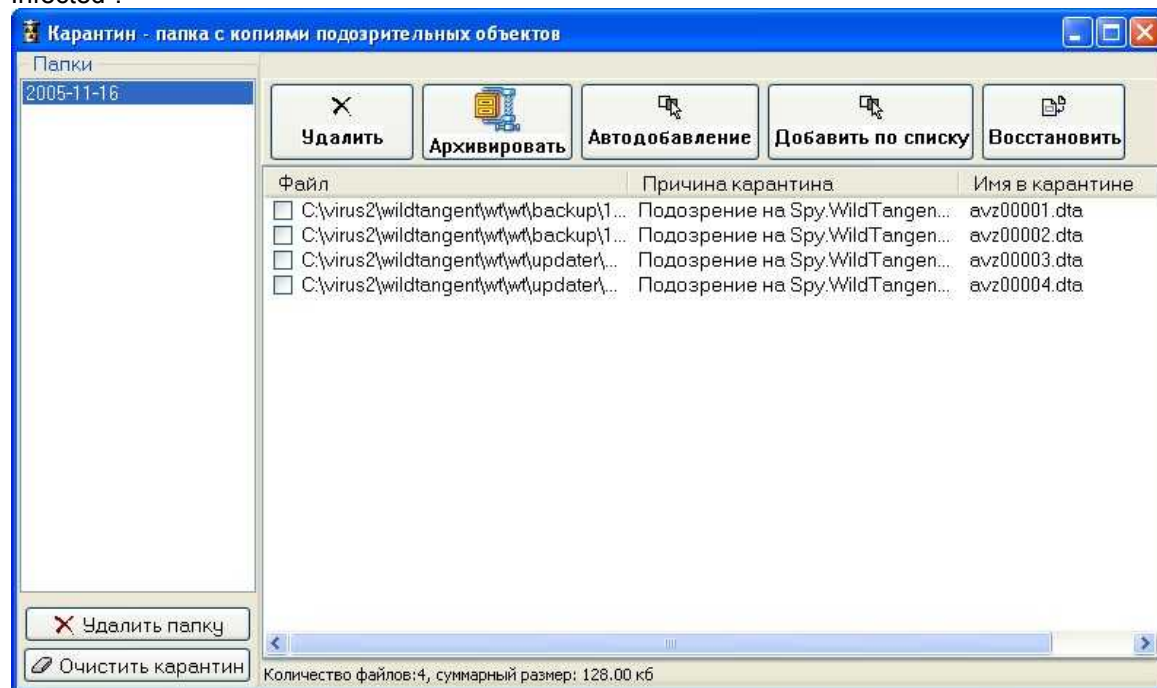


3 Карантин

3.1 Карантин и папка Infected

В утилите AVZ предусмотрено две папки для хранения копий удаляемых вредоносных файлов (папка Infected - карантин для инфицированных файлов) и подозрительных файлов (папка "карантин"), найденных эвристическим анализатором или помещенных в карантин пользователем вручную.

Работа с карантином и папкой Infected абсолютно идентична и производится в специальном окне, вызываемом из меню "Файл". Для работы с карантином необходимо воспользоваться меню "Просмотр карантина", для работы с папкой Infected - "Файл/ Просмотр папки Infected".



Дальнейшую работу для определенности рассмотрим на примере папки "Карантин".

В левой части окна размещается список вложенных папок папки "Карантин". Для упрощения работы папки именуются автоматические - имя папки формируется по маске ГГГГ-ММ-ДД, где ГГГГ - четыре цифры года, ММ - две цифры месяца, ДД - две цифры дня. Вложенные папки создаются автоматически по мере надобности, и такая структура существенно упрощает анализ файлов в карантине - они оказываются сгруппированными по дате занесения в карантин. Текущая папка подсвечивается маркером. Кнопки под списком папок позволяют удалить текущую папку (Кнопка "Удалить папку") и произвести полную очистку карантина (кнопка "Очистить карантин"). Полная очистка карантина приводит к удалению всех файлов и папок, находящихся в карантине.

При выборе папки в левой части окна отображается список находящихся в карантине файлов. Для каждого файла отображается полное исходное имя файла, причина помещения файла в карантин (имя вируса, сообщение о подозрении на вирус, информация о занесении файла вручную), имя файла в карантине и его размер.

Каждый файл можно отметить при помощи переключателя перед его именем. Для быстрой отметки всех файлов можно воспользоваться меню, вызываемым при нажатии правой кнопки над списком (меню содержит пункты, позволяющие отметить все файлы, снять

отметку со всех файлов и инвертировать отметку).

Над списком файлов располагается панель инструментов с кнопками, позволяющими выполнять операции над отмеченными файлами. Нажатие кнопки "Удалить отмеченные файлы" позволяет избирательно удалить один или несколько отмеченных файлов, что полезно для чистки карантина. Кнопка "Архивировать отмеченные файлы" позволяет создать архив, в который помещаются отмеченные файлы. Архив по формату полностью совместим с ZIP архивом и автоматически защищается паролем "virus". Пароль полезен для отправки файлов на анализ, т.к. незащищенные паролем файлы могут быть остановлены почтовым антивирусом. Архивация выполняется встроенным в AVZ архиватором (для работы этой функции устанавливать и настраивать архиваторы типа WinZip не требуется). Если при создании архива указывается имя существующего файла zip, то файлы дописываются к существующему архиву. Таким образом, если возникает необходимость поместить в архив несколько файлов из разных папок карантина, то эту задачу можно решить, указывая в диалоге "Поместить в архив" один и тот же архивный файл.

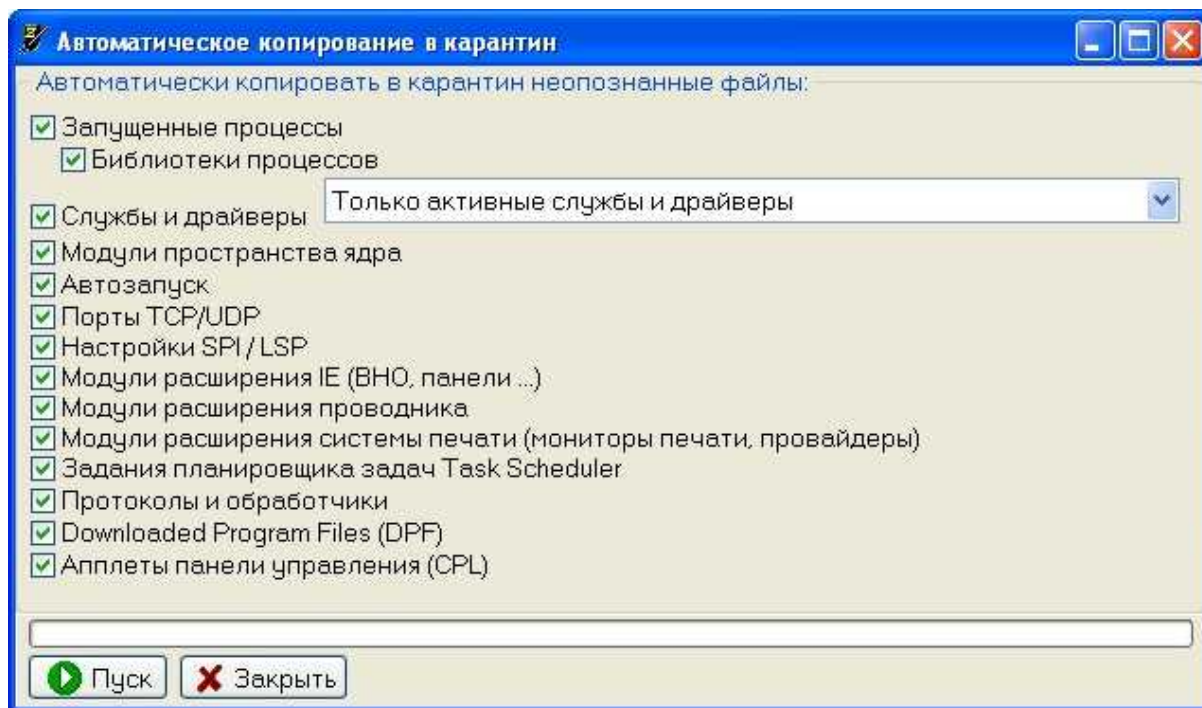
Файлы в карантине (и соответственно в архиве) переименовываются - имена файлов формируются как avzNNNN.dta (где NNNNN - порядковый номер файла в папке). Для каждого файла avzNNNN.dta создается avzNNNN.ini, содержащий описание файла (исходное имя и местоположение файла в системе, дата и время помещения в карантин, причины карантина и размер файла).

Кнопка "Добавить по списку" вызывает [диалоговое окно карантина файлов по списку](#)³⁶. Кнопка "Автодобавление" вызывает диалоговое окно [системы автоматического карантина файлов](#)³⁵.

3.2 Автокарантин

Автокарантин - это встроенное средство AVZ, позволяющее автоматически поместить в карантин все файлы, которые AVZ не опознает как компоненты системы при помощи каталога безопасности Microsoft или как безопасные объекты при помощи базы безопасных объектов. Автоматический карантин очень удобен для оперативного сбора файлов на исследуемом компьютере для последующего изучения или проверки несколькими антивирусами в стационарных условиях.

Автокарантин вызывается из меню "Файл\Автокарантин" или при нажатии кнопки "Автодобавление" в [окне просмотра содержимого карантина](#)³⁴.



Диалоговое окно системы автоматического копирования в карантин

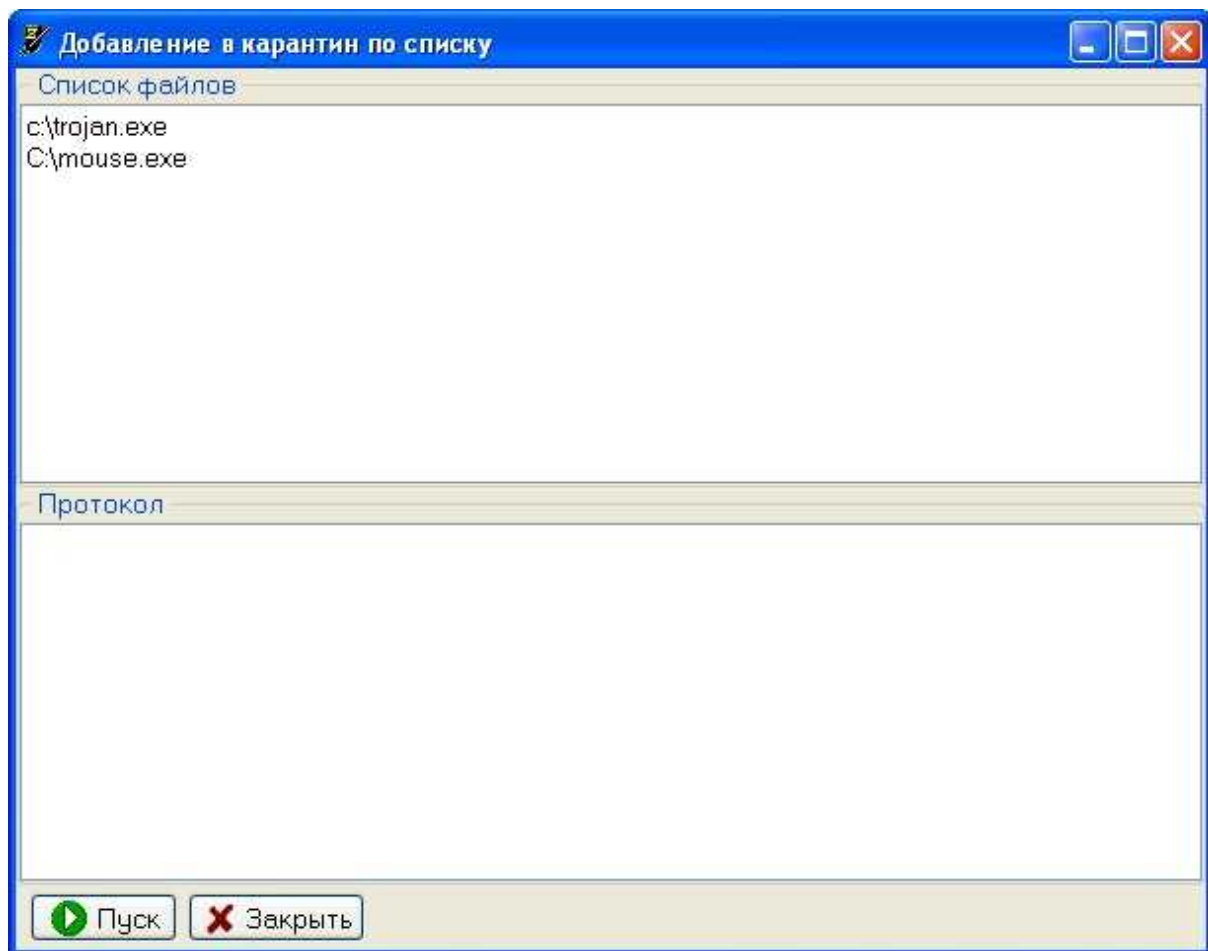
Переключатели в диалоговом окне автоматического карантина позволяют указать, файлы каких категорий необходимо поместить в карантин. По умолчанию отмечены все категории. Для категории "Службы и драйвера" предусмотрена дополнительная настройка - по умолчанию в карантин копируются только активные службы и драйверы. Однако при помощи выпадающего списка можно выбрать другой режим - "все службы и драйверы".

Копирование в карантин начинается при нажатии кнопки "Пуск" и может продолжаться достаточно продолжительное время (в зависимости от конкретной конфигурации, наличия антивирусного монитора и мощности компьютера автокарантин работает от 5-10 секунд до 2-3 минут). Процесс анализа отображается прогресс-индикатором.

Нажатие кнопки "Закреть" закрывает окно автоматического копирования в карантин.

3.3 Добавление в карантин по списку

Сервис "Добавление в карантин по списку" удобен для массового добавления файлов в карантин по заранее известному списку. Для вызова диалогового окна карантина по списку можно применить меню "Файл\Добавление в карантин по списку" или нажать кнопку "Добавить по списку" в [окне карантина](#) ³⁴.



Список файлов необходимо внести в поле "Список файлов", в одной строке должно быть по одному имени файла. В именах файлов допускается указание символов маски "*" и "?". После заполнения списка файлов необходимо нажать кнопку "Пуск" для выполнения карантина. В поле "Протокол" при этом формируется протокол выполнения карантина файлов.

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



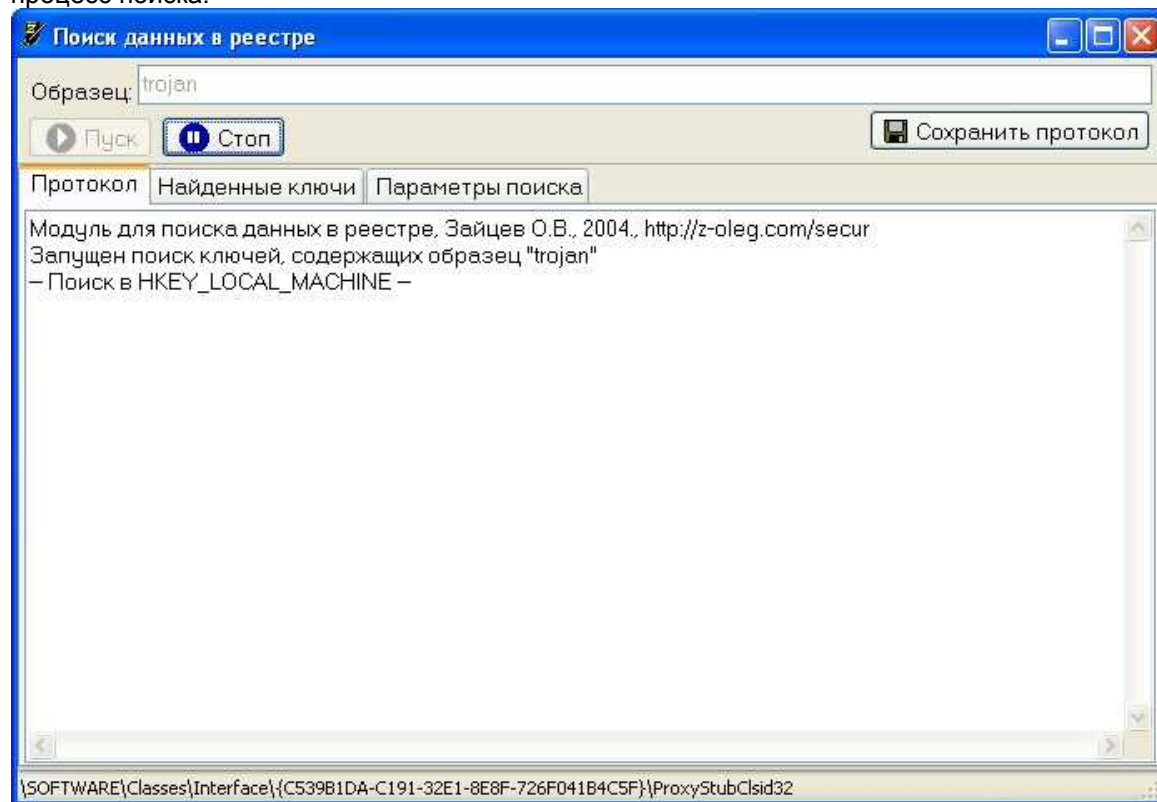
IV

4 Встроенные средства поиска

4.1 Поиск данных в реестре

В окне системы поиска в реестре размещается строка с образцом для поиска. В этой строке необходимо ввести образец для поиска, например имя DLL файла или фрагмент URL, установившегося в качестве домашней страницы.

При нажатии кнопки "Пуск" производится запуск процесса поиска (сам процесс может длиться несколько минут - скорость поиска зависит от производительности компьютера, количества ключей в реестре и настроек программы). Кнопка "Стоп" позволяет прервать процесс поиска.



Закладка "Протокол"

Закладка "Протокол" содержит протокол работы программы. В протоколе содержатся найденные ключи, служебная информация, информация об удаленных ключах. Протокол может быть скопирован через буфер обмена для размещения в конференции или отправки по электронной почте. Кроме того, можно сохранить протокол в текстовый файл (меню "Файл/Сохранить протокол").

Закладка "Найденные ключи"

Закладка "Найденные ключи" содержит таблицу с отсортированным по имени ключа списком всех найденных ключей (в таблице три столбца - "ключ", "параметр" и "значение"). Каждая строка таблицы содержит переключатель (checkbox), который позволяет отметить строку.

Под таблицей размещены две кнопки:

- "Создать рег файл с отмеченными ключами". При нажатии на эту кнопку создается стандартный рег файл, в который экспортируются отмеченные ключи реестра. Данная

функция полезна для создания резервных копий перед удалением ключей. Важным моментом является то, что в файл экспортируются только отмеченные ключи (если отмеченный ключ имеет параметры или вложенный ключи, то все они не будут экспортированы). Для полного экспорта одного или нескольких ключей следует воспользоваться программой Regedit;

- "Удалить отмеченные ключи". Удаляет отмеченные ключи (или значения ключей). При удалении ключей следует соблюдать большую осторожность, т.к. повреждение реестра может привести к краху системы. Поэтому если Вы не уверены в назначении удаляемых ключей, то Вам необходимо обязательно проконсультироваться со специалистами; При нажатии над таблицей правой кнопки мыши выводится всплывающее меню, позволяющее отметить все строки, снять пометку со всех строк и инвертировать отметку.

Закладка "Параметры"

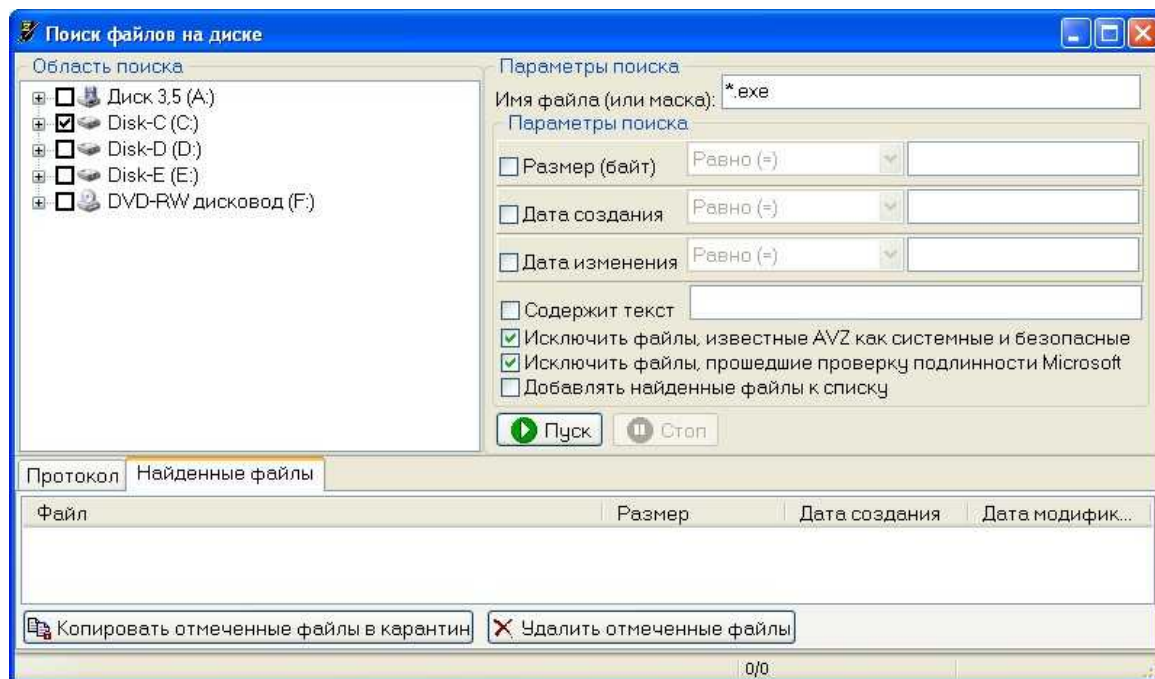
Данная закладка содержит настройки поиска и настройки программы.

- Группа параметров "Искать в ключах" позволяет указать, в ключах каких ветвей реестра необходимо производить поиск.
- Группа параметров "Просматривать при поиске" позволяет настроить область поиска. По умолчанию поиск ведется в именах и значениях параметров

4.2 Поиск файлов на диске

Для поиска файлов на диске в AVZ встроена специальная сервисная функция, вызываемая из меню "Сервис / Поиск файлов на диске". Поиск файлов средствами AVZ имеет ряд преимуществ над системным поиском по ряду причин:

- Система поиска файлов AVZ защищена встроенной в AVZ системой противодействия RootKit - это сущности основная причина, по которой в AVZ встроена система поиска файлов. Маскировка процессов в памяти и файлов на диске присуща многим современным троянским программам, производится эта маскировка как правило перехватом API. В результате системные средства поиска позволяют обнаружить на диске маскируемые файлы(и соответственно не позволяют скопировать их для отправки на анализ).
- Система поиска файлов связана с карантином AVZ - любой из найденных файлов может быть скопирован в карантин;
- Поиск производится независимо от расширения файла, его местоположения и атрибутов - системные средства поиска часто при поиске показывают не все файлы (это особенно важно для системных папок типа Downloaded Program Files)



Для выполнения поиска необходимо отметить диски (или папки) в древовидном списке дисков и папок "Область поиска". Меню, вызываемое по правой кнопке мыши над проводником "Область поиска", позволяет быстро отметить жесткие диски, дисководы и CDROM. После задания области поиска необходимо указать имя файла или маску. Можно указать несколько имен файлов или несколько масок, перечисляя их через пробел, запятую или точку с запятой. Можно сочетать имена файлов и маски.

Примеры:

. - поиск всех файлов

*.exe - поиск файлов с расширением exe

dialer.exe - поиск файла с конкретным именем "dialer.exe"

dialer.exe, *.dll, trojan*.sys - поиск файлов с именем dialer.exe или файлов с любым именем и расширением DLL или файлов с именем Trojan<любые символы>.sys

*.exe, *.dll, *.sys, *.ocx - поиск файлов с указанными расширениями

В маске можно использовать стандартные символы * (на месте * могут встречаться любые символы в любом количестве) и ? (любой символ в заданной позиции).

Пример:

Trojan.ex? - этой маске к примеру могут соответствовать файлы Trojan.exe и Trojan.ex_
Маска является обязательной (для поиска файлов с любым именем необходимо указать маску *.*).

Кроме маски можно задать ряд условий поиска - размер, дату создания, дату модификации. Для включения условия необходимо установить переключатель, выбрать тип условия их списка и заполнить параметры в полях. Если параметр заполнен неправильно, то он подчеркивается красной волнистой линией и поиск блокируется. При наведении курсора мыши на некорректное поле выводится всплывающая подсказка, поясняющая причину ошибки.

Особого внимания заслуживает опция фильтра "Исключить файлы, известные AVZ как системные и безопасные". Включение данного переключателя приводит к тому, что каждый файл, соответствующий всем прочим условиям поиска проверяется по базе известных AVZ

безопасных и системных файлов. Опознанные по базе безопасные файлы исключаются из результатов поиска. Это, как правило, очень существенно сокращает количество найденных файлов. Данный режим фильтрации очень удобен для поиска вирусов и SpyWare - можно задать маску поиска "*.exe *.dll", указать папку Windows, задать диапазон дат (как правило имеет смысл искать файлы, созданные за последний месяц от момента появления проблем с ПК) и включить опцию "Исключить файлы, известные AVZ как системные и безопасные"

Поиск запускается при нажатии кнопки "Пуск". Для остановки поиска применяется кнопка "Стоп". Результаты поиска выводятся в протокол (закладка "Протокол") и в таблицу (закладка "Найденные файлы"). В таблице отображаются основные параметры найденных файлов, любой из файлов можно отметить при помощи переключателя перед его именем. Отмеченные файлы можно скопировать в карантин или удалить (эти операции выполняются при нажатии соответствующих кнопок под списком).

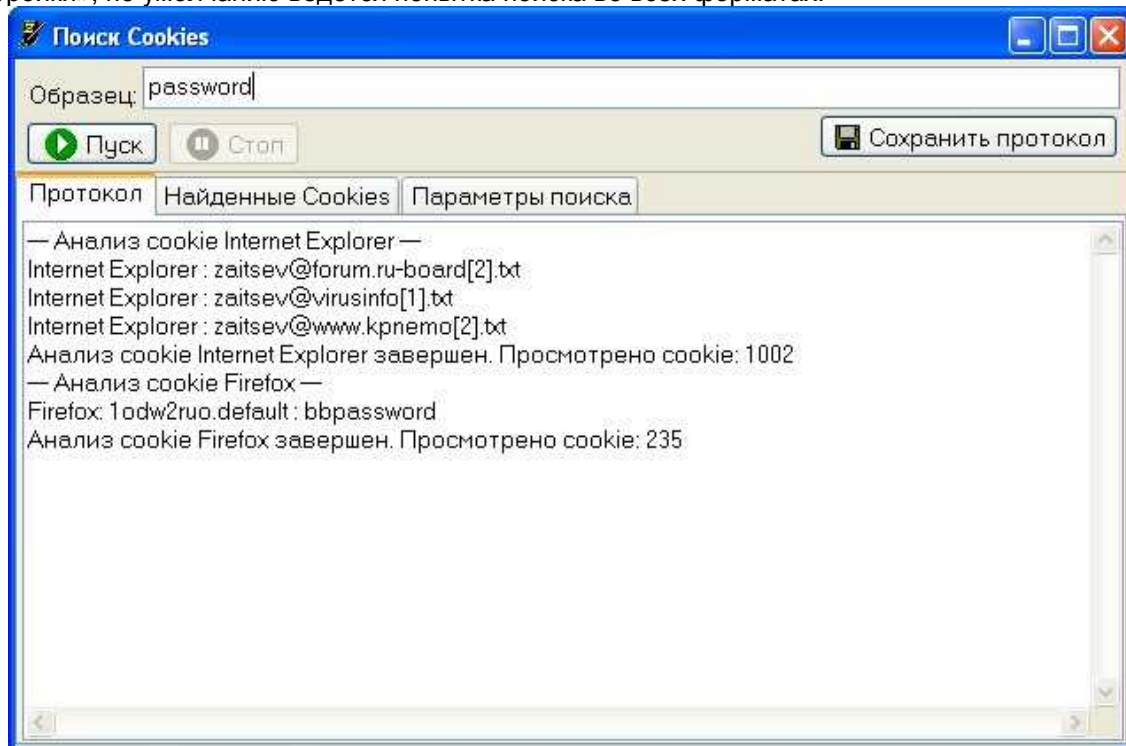
На заметку:

У таблицы найденных файлов есть меню, вызываемое по правой кнопке мыши. Данное меню позволяет отметить все найденные файлы, снять и инвертировать отметку.

4.3 Поиск Cookie по данным

Для оперативного анализа содержимого хранимых на компьютере Cookies можно применить "Поиск Cookie по данным". Этот анализатор изучает Cookie, которые сохраняются браузерами Internet Explorer и Mozilla Firefox. Окно системы поиска вызывается из меню «Сервис\Поиск cookies». Данный анализатор позволяет пользователю узнать, какие сайты сохраняют в cookies критичную для него информацию - в последствии для этих сайтов можно создать правила, блокирующие прием от них cookies.

Особенностью системы поиска является то, что поиск может вестись одновременно по нескольким текстовым образцам (при этом образцы разделяются пробелом или «;»). Поиск ведется с учетом того, что данные в cookie могут быть представлены в формате Base64, UUE, url-encoding или quoted-printable. Анализируемые форматы можно выбрать на закладке «Настройки», по умолчанию ведется попытка поиска во всех форматах.



Для проведения анализа в строке «Образец» необходимо ввести фрагменты e-mail адресов пользователя, применяемые для регистрации на Интернет сайтах имена и пароли, фрагменты номеров кредитных карт или иную информацию, которая вводилась в WEB формы и утечка которой, по мнению пользователя, представляет для него угрозу. При указании образцов поиска следует учитывать, что по сути ведется поиск вхождения указанных образцов в данных cookie, поэтому часто для эффективного поиска достаточно указать уникальные фрагменты, например "newvirus" вместо «newvirus@z-oleg.com» или последние 5 цифр номера кредитной карты вместо ее полного номера.

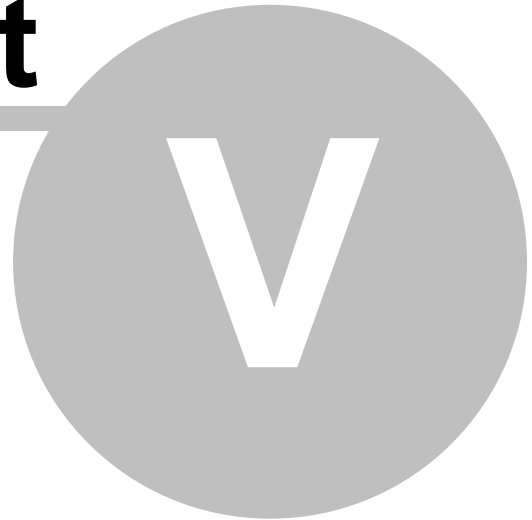
После задания образцов необходимо нажать кнопку «Пуск» для запуска поиска. Поиск может занять некоторое время, как правило, не более 10-20 секунд. После завершения поиска формируется протокол, в котором указано, в каких cookie встречались указанные образцы. На закладке «Найденные Cookies» можно просмотреть список найденных Cookies, при нажатии кнопки «Просмотреть Cookie» на экран выводится содержимое текущего Cookie для детального анализа.

На закладке "Настройки" задаются настройки поиска. Настройки позволяют указать типы браузеров, для которых необходимо анализировать Cookie и типы кодировок, которые необходимо применять в ходе анализа содержимого Cookie.

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



5 Встроенные утилиты

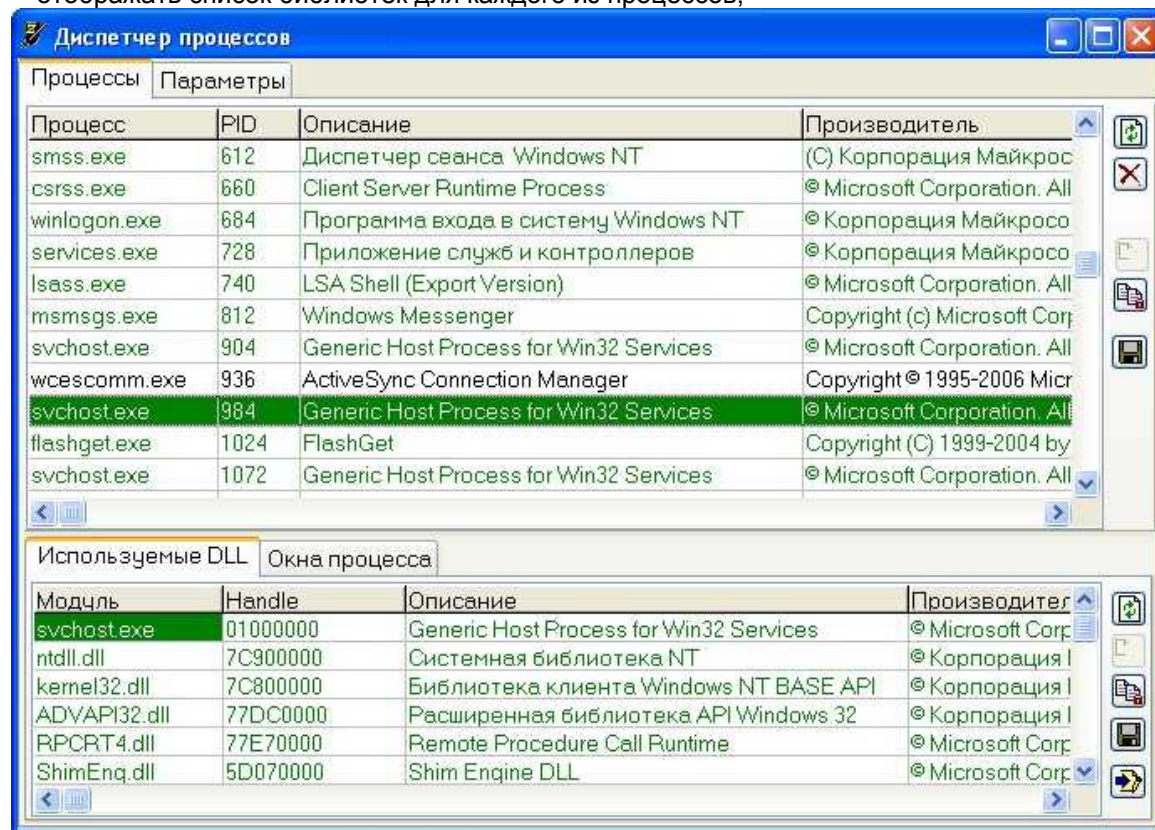
5.1 Диспетчер процессов

Для поиска шпионского ПО и троянских программ вручную в программу AVZ встроен собственный диспетчер процессов.

Вызов диспетчера производится из меню "Сервис \ Диспетчер процессов".

Диспетчер процессов позволяет:

- отображать список запущенных процессов (список процессов строится различными способами, для Windows NT предусмотрено построение списка процессов штатными средствами, при помощи Native API, при помощи подсистемы поиска скрытых процессов);
- отображать список библиотек для каждого из процессов;



Для всех объектов диспетчер процессов может вычислять контрольные суммы MD5 и собирает дополнительную информацию.

Другой особенностью диспетчера процессов AVZ является возможность использования в его работе системы противодействия RootKit. В результате диспетчер процессов AVZ, как правило, в состоянии обнаружить и остановить процессы, маскируемые RootKit. Кроме того, если в систему установлен драйвер AVZ PM, то собранная им информация используется диспетчером процессов, что в частности позволяет удалять маскирующиеся процессы из KernelMode.

Закладка "Процессы"

Закладка "Процессы" содержит список процессов (верхняя таблица) и информацию о текущем процессе (на закладках под таблицей процессов). Таблица процессов обновляется при нажатии кнопки "Обновить".

Кнопка "Удалить процесс" позволяет завершить работу текущего процесса. Кнопка "Копировать в карантин" позволяет отправить копию исполняемого файла текущего процесса в папку "Карантин". Данная кнопка доступна только в случае, если файл не опознан по базе безопасных

Кнопка "Копировать в карантин все процессы, не опознанные как безопасные" позволяет отправить копию исполняемых файлов всех процессов, не опознанных по базе безопасных в папку "Карантин".

Кнопка "Сохранить протокол".

Производит формирование HTML протокола, который сохраняется на диск в указанное пользователем место (для сохранения протокола выводится диалог сохранения файла стандартного вида). После сохранения протокола выдается предложение открыть его (открытие производится применяемым по умолчанию браузером).

Закладка "Параметры"

На закладке "Параметры" размещены настройки диспетчера процессов:

- Переключатель "Вычислять контрольные суммы MD5 для файлов" - включение этого переключателя активирует расчет контрольных сумм для каждого файла. Этот процесс требует больших затрат ресурсов и обновление списка происходит медленнее;
- Переключатель "Проверять файлы по базе безопасных файлов" - включение этого переключателя активирует проверку исполняемых файлов и DLL по базам безопасных объектов, идущим в составе AVZ. Файлы, опознанные как безопасные, выделяются при просмотре зеленым цветом. Данный режим полезен для поиска неопознанных процессов и DLL

Контекстное меню списка DLL

- Копировать в карантин. Выполняет копирование текущего файла в карантин (безопасные файлы в карантин не копируются)
- Поиск в Google - открывает браузер по умолчанию и передает ему строку URL, выполняющего поиск в Google по имени файла
- Поиск в Yandex - открывает браузер по умолчанию и передает ему строку URL, выполняющего поиск в Yandex по имени файла
- Поиск в Rambler - открывает браузер по умолчанию и передает ему строку URL, выполняющего поиск в Rambler по имени файла
- Поиск в реестре по имени файла - вызывает окно поиска в реестре, в образце поиска которого уже задано имя текущего файла
- Поиск в реестре по имени файла - вызывает окно поиска в реестре, в образце поиска которого уже задано полное имя текущего файла
- Принудительно выгрузить DLL - выполняет принудительную выгрузку указанного модуля.

На заметку

Принудительная выгрузка модуля в большинстве случаев приведет к неустранимой ошибке в приложении, из адресного пространства которого выгружена DLL. Собственно, ошибка в этом приложении любом случае произойдет при первой же попытке приложения вызвать функцию выгруженной таким образом DLL.

5.2 Диспетчер служб и драйверов

Для поиска шпионского ПО и троянских программ вручную в программу AVZ встроен собственный диспетчер служб и драйверов (он работает только под NT и последующими системами на ее основе - Windows 2000, XP).

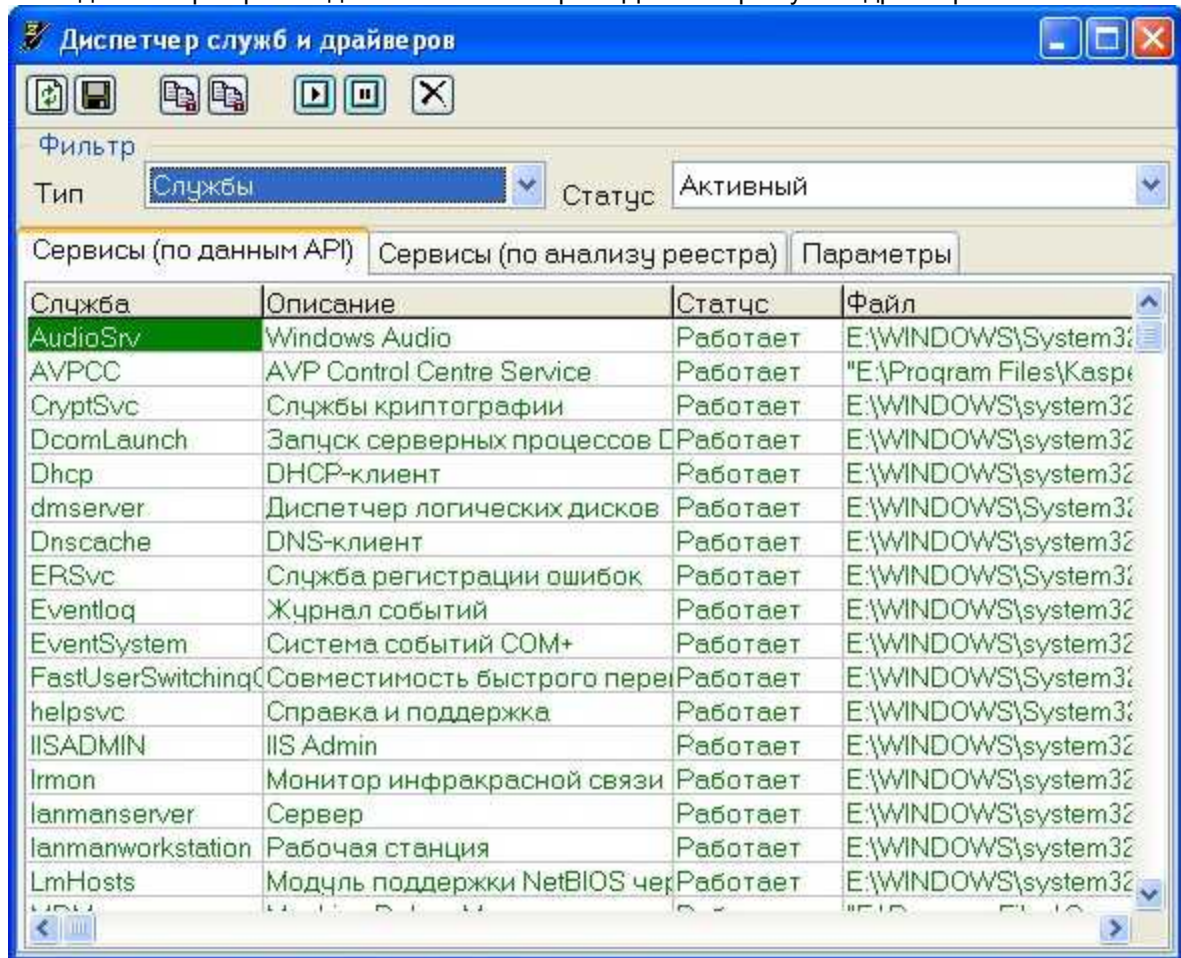
Диспетчер служб и драйверов позволяет:

- Отображать список служб и драйверов с фильтрацией по типу (служба, драйвер) и статусу (запущен, не запущен);

- Останавливать и запускать службы (или загружать/выгружать драйвера);
- Копировать файлы в карантин;
- Формировать HTML протоколы.

Особенностью диспетчера служб и драйверов AVZ является возможность использования в его работы системы противодействия RootKit.

Вызов диспетчера производится из меню "Сервис\Диспетчер служб и драйверов"



Фильтр

Фильтр позволяет управлять информацией, отображаемой на закладках с данными. Тип задает тип записей (Служба, Драйвер, Все), Поле статус - статус службы/драйвера в текущий момент (Активный, Не активный, Все). При включении закладки "Сервисы (по анализу реестра)" фильтрация по статусу становится невозможной - тип фильтра автоматически устанавливается в положение "Все".

Закладки

В окне диспетчера служб и драйвером предусмотрено три закладки:

"Сервисы (по данным API)" - отображает службы и (или) драйверы по данным, возвращаемым API

"Сервисы (по анализу реестра)" - отображает службы и (или) драйверы по данным, зарегистрированным в реестре

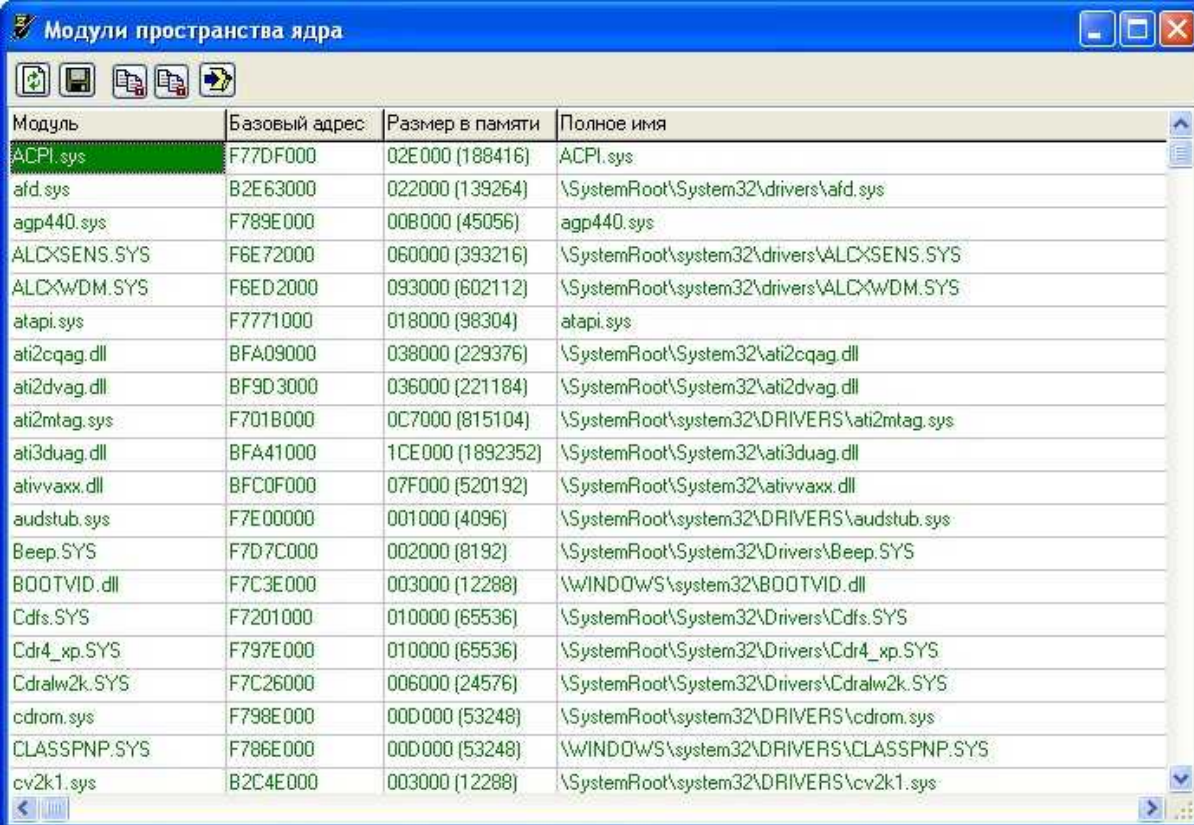
"Параметры" - параметры диспетчера. Содержит единственный переключатель, управляющий проверкой файлов по базе безопасных.

Кнопки панели управления

- Кнопка "Обновить". Обновляет отображаемый список. Обновление производится автоматически при смене параметров фильтрации
- Кнопка "Сохранить протокол". Производит формирование HTML протокола, который сохраняется на диск в указанное пользователем место (для сохранения протокола выводится диалог сохранения файла стандартного вида). После сохранения протокола выдается предложение открыть его (открытие производится применяемым по умолчанию браузером).
- "Копировать в карантин" - копирует текущий файл в карантин. Кнопка доступна только для службы и драйверов, не опознанных по базе безопасных и каталогу Microsoft
- "Копировать в карантин все файлы, не опознанные по как безопасные" - копирует в карантин все службы/драйверы, не опознанные по базе безопасных и каталогу Microsoft
- "Запустить службу (загрузить драйвер)" - производит попытку запуска службы или загрузки драйвера
- "Остановить службу (выгрузить драйвер)" - производит попытку остановки службы или выгрузки драйвера
- "Удалить текущую службу\драйвер" - производит удаление текущей службы или драйвера (файл с диска при этом не удаляется).

5.3 Модули пространства ядра

Менеджер модулей пространства ядра предназначен для отображения модулей, загруженных в пространство ядра. Запуск менеджера производится из меню "Сервис/Модули пространства ядра".



Модуль	Базовый адрес	Размер в памяти	Полное имя
ACPI.sys	F77DF000	02E000 (188416)	ACPI.sys
afd.sys	B2E63000	022000 (139264)	\SystemRoot\System32\drivers\afd.sys
agp440.sys	F789E000	008000 (45056)	agp440.sys
ALCXSENS.SYS	F6E72000	060000 (393216)	\SystemRoot\system32\drivers\ALCXSENS.SYS
ALCXWDM.SYS	F6ED2000	093000 (602112)	\SystemRoot\system32\drivers\ALCXWDM.SYS
atapi.sys	F7771000	018000 (98304)	atapi.sys
ati2cqag.dll	BFA09000	038000 (229376)	\SystemRoot\System32\ati2cqag.dll
ati2dvag.dll	BF9D3000	036000 (221184)	\SystemRoot\System32\ati2dvag.dll
ati2mtag.sys	F701B000	0C7000 (815104)	\SystemRoot\system32\DRIVERS\ati2mtag.sys
ati3duag.dll	BFA41000	1CE000 (1892352)	\SystemRoot\System32\ati3duag.dll
ativvaxx.dll	BFC0F000	07F000 (520192)	\SystemRoot\System32\ativvaxx.dll
audstub.sys	F7E00000	001000 (4096)	\SystemRoot\system32\DRIVERS\audstub.sys
Beep.SYS	F7D7C000	002000 (8192)	\SystemRoot\System32\Drivers\Beep.SYS
BOOTVID.dll	F7C3E000	003000 (12288)	\WINDOWS\system32\BOOTVID.dll
Cdfs.SYS	F7201000	010000 (65536)	\SystemRoot\System32\Drivers\Cdfs.SYS
Cdr4_xp.SYS	F797E000	010000 (65536)	\SystemRoot\System32\Drivers\Cdr4_xp.SYS
Cdrblw2k.SYS	F7C26000	006000 (24576)	\SystemRoot\System32\Drivers\Cdrblw2k.SYS
cdrom.sys	F798E000	00D000 (53248)	\SystemRoot\system32\DRIVERS\cdrom.sys
CLASSPNP.SYS	F786E000	00D000 (53248)	\WINDOWS\system32\DRIVERS\CLASSPNP.SYS
cv2k1.sys	B2C4E000	003000 (12288)	\SystemRoot\system32\DRIVERS\cv2k1.sys

Менеджер модулей пространства ядра позволяет:

- Просматривать список модулей с сортировкой по имени, базовому адресу или размеру

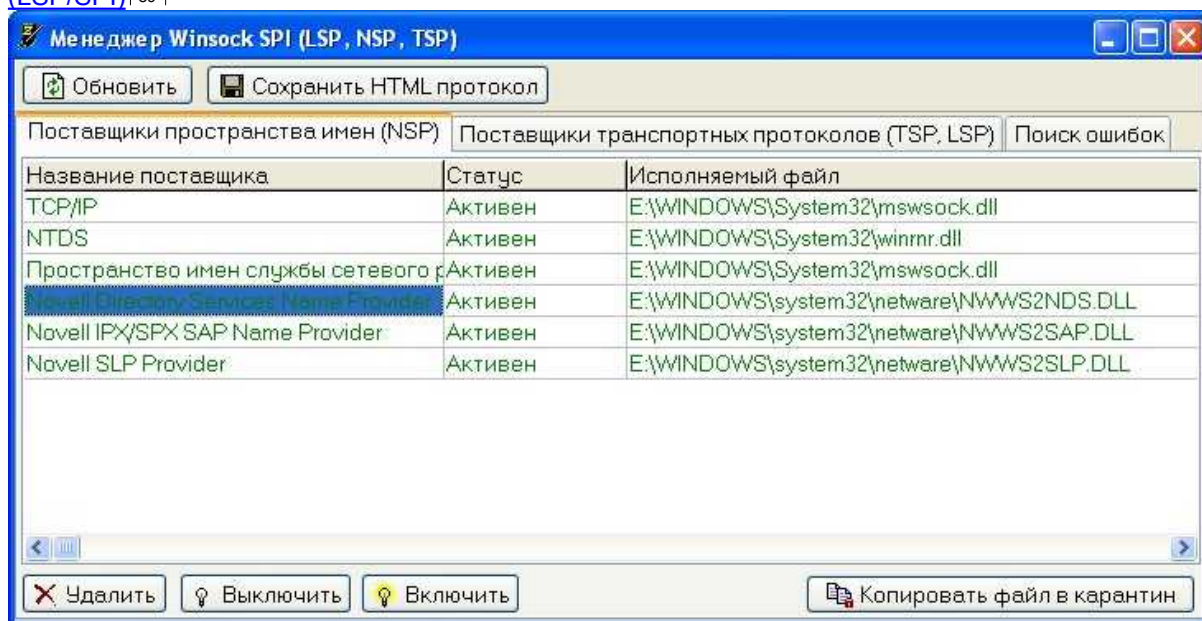
модуля;

- Копировать в карантин модули, не опознанные по базе безопасных AVZ;
- Выполнять дампирование любого модуля для последующего изучения.
- Формировать HTML протоколы

Менеджер модулей пространства ядра связан с автокарантином, исследованием системы и системой [AVZPM](#)^[72]

5.4 Менеджер Winsock SPI (LSP, NSP, TSP)

Менеджер SPI/LSP позволяет просматривать списки поставщиков службы имен (NSP) и транспорта (TSP). Каждый поставщик может быть удален пользователем вручную. Подробно про SPI/LSP/TSP/NSP провайдеры можно почитать в отдельном разделе - [Local Service Provider \(LSP/SPI\)](#)^[89]



****^[89]

В окне менеджера Winsock SPI размещено три закладки:

- Поставщики пространства имен (NSP - Name Service Provider). Отображает список библиотек, зарегистрированных в качестве NPS поставщиков. Для каждой библиотеки отображается название поставщика, статус и полное имя исполняемого файла. Файлы могут быть скопированы в карантин, включены или отключены или удалены при помощи кнопок на панели под списком поставщиков
- Поставщики транспортных протоколов. Данная закладка аналогична предыдущей, но для поставщиков транспортных протоколов предусмотрена только операция удаления
- Поиск ошибок. Данная закладка содержит список ошибок в настройках LSP, обнаруженных в ходе их изучения. На данной закладке предусмотрена кнопка "Автоматическое исправление найденных ошибок", которая запускает процесс автоматического устранения ошибок в настройках.

На заметку

Поиск ошибок может некорректно работать в случае запуска AVZ из терминальной сессии

На заметку

Автоматическое исправление ошибок доступно из диалогового окна "[Восстановление системы](#)"^[59] и из скриптового языка (см. команду [AutoFixSPI](#)^[137]). Начиная с версии 4.26 перед исправлением ошибок AVZ автоматически создает резервную копию настроек в папке BackUp.

Резервная копия является стандартным REG файлом.

5.5 Открытые порты TCP/UDP

В AVZ встроен менеджер, позволяющий просматривать:

- Открытые порты TCP
- Активные соединения по протоколу TCP/IP
- Открытые порты UDP

Для каждого открытого порта (или соединения) выводится имя исполняемого файла, но эта возможность действует только в Windows XP и Windows 2003. Кроме того, у AVZ имеется особая база с описанием назначения наиболее распространенных портов и степени их опасности (опасными считаются порты, используемые сетевыми вирусами, троянскими и Backdoor программами). Открытый порт, который используется опасной программой, выделяется красным цветом и в описании указано, какие вредоносные программы могут использовать данный порт.

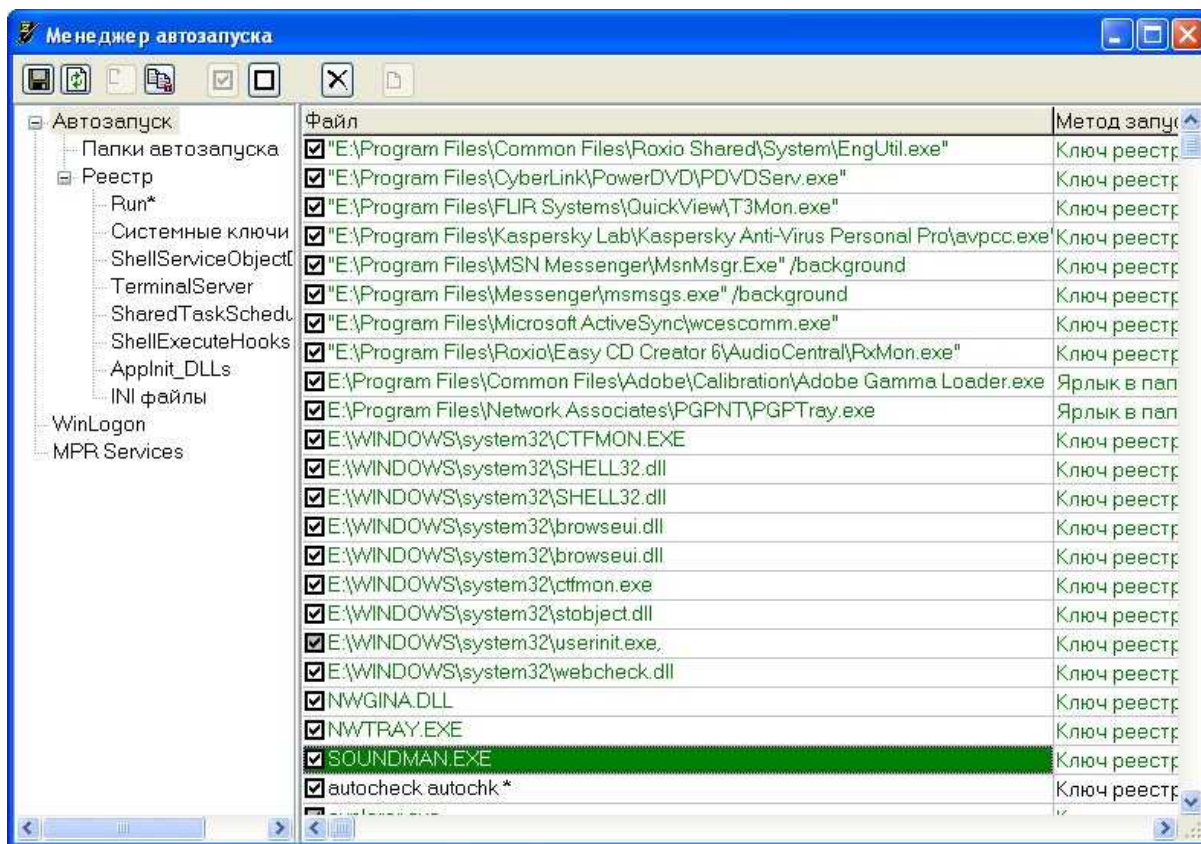
Кроме возможности просмотра списка открытых портов в AVZ встроена автоматическая проверка, выполняемая в ходе сканирования системы. В ходе данной проверки производится построение списка открытых портов TCP и UDP, после чего AVZ сопоставляет этот список с базой портов, применяемых троянскими программами и сетевыми вирусами. При обнаружении совпадений в протокол выводятся предупреждения с указанием номера порта, его описания и имени программы (имя программы выводится только в XP, Windows 2003).

5.6 Менеджер автозапуска

Менеджер автозапуска отображает список программ и библиотек, загружаемых в ходе загрузки системы. Менеджер позволяет:

- Отображать список автозапускаемых программ с указанием метода запуска;
- Временно отключать элементы автозапуска;
- Удалять элементы автозапуска;
- Копировать файлы в карантин;
- Формировать HTML протоколы

Анализатор менеджера автозапуска подключен к исследованию системы и автокарантину.



AVZ - менеджер автозапуска

Окно менеджера автозапуска разделено на две части. В левой отображается список категорий автозапуска - папки автозапуска, реестре, Winlogon, MPR Services. Для просмотра конкретной группы автозапуска необходимо выбрать ее в древовидном списке категорий.

5.7 Менеджер расширений IE

Менеджер расширений IE отображает список подключенных к Internet Explorer модулей расширения. Модули расширения - это BHO и различные панели.

Менеджер позволяет:

- Отображать список модулей расширения IE;
- Временно отключать расширения;
- Удалять модули расширения (при этом исполняемый файл не удаляется - удаляется только регистрационная информация в реестре);
- Копировать файлы в карантин;
- Формировать HTML протоколы

Анализатор менеджера расширений IE подключен к исследованию системы и автокарантину.

5.8 Менеджер расширений проводника

Менеджер расширений проводника отображает список подключенных к Explorer модулей расширения.

Менеджер позволяет:

- Отображать список модулей расширения проводника;
- Временно отключать расширения;
- Удалять модули расширения (при этом исполняемый файл не удаляется - удаляется

только регистрационная информация в реестре);

- Копировать файлы в карантин;
- Формировать HTML протоколы

Анализатор менеджера расширений Explorer подключен к исследованию системы и автокарантину.

5.9 Менеджер апплетов панели управления (CPL)

Менеджер апплетов панели управления отображает список CPL файлов. CPL файл - это динамическая библиотека, экспортирующая определенные функции для взаимодействия с панелью управления.

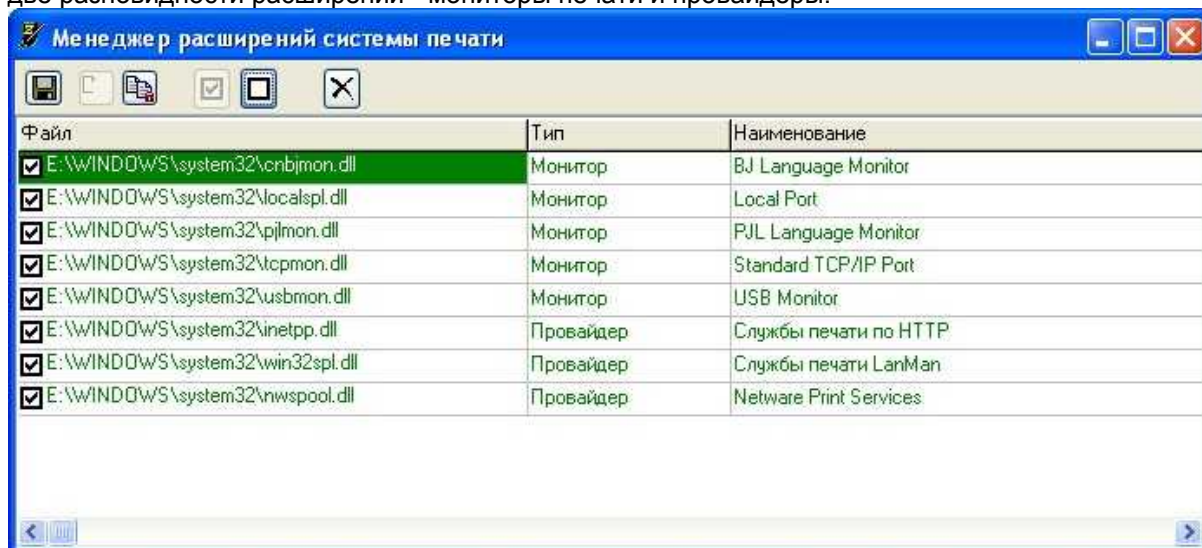
Менеджер позволяет:

- Отображать апплетов панели управления;
- Временно отключать любой из апплетов;
- Удалять апплеты (при этом **удаляется** исполняемый CPL файл);
- Копировать файлы в карантин;
- Формировать HTML протоколы

Анализатор апплетов панели управления подключен к исследованию системы и автокарантину.

5.10 Менеджер расширений системы печати

Менеджер расширений проводника отображает список расширений системы печати. Известно две разновидности расширений - мониторы печати и провайдеры.



Менеджер расширений системы печати

Менеджер позволяет:

- Отображать список модулей расширений системы печати;
- Временно отключать расширения;
- Удалять модули расширения (при этом исполняемый файл не удаляется - удаляется только регистрационная информация в реестре);
- Копировать файлы в карантин;
- Формировать HTML протоколы

Анализатор менеджера расширений печати подключен к исследованию системы и автокарантину.

5.11 Менеджер планировщика заданий (Task Scheduler)

Менеджер планировщика заданий отображает задания, поставленные в очередь планировщиком задач или командой АТ. Анализ заданий полезен ввиду того, что некоторые SpyWare и троянские программы используют планировщик для своего запуска или для запуска дроппера, который восстанавливает удаленные компоненты SpyWare.

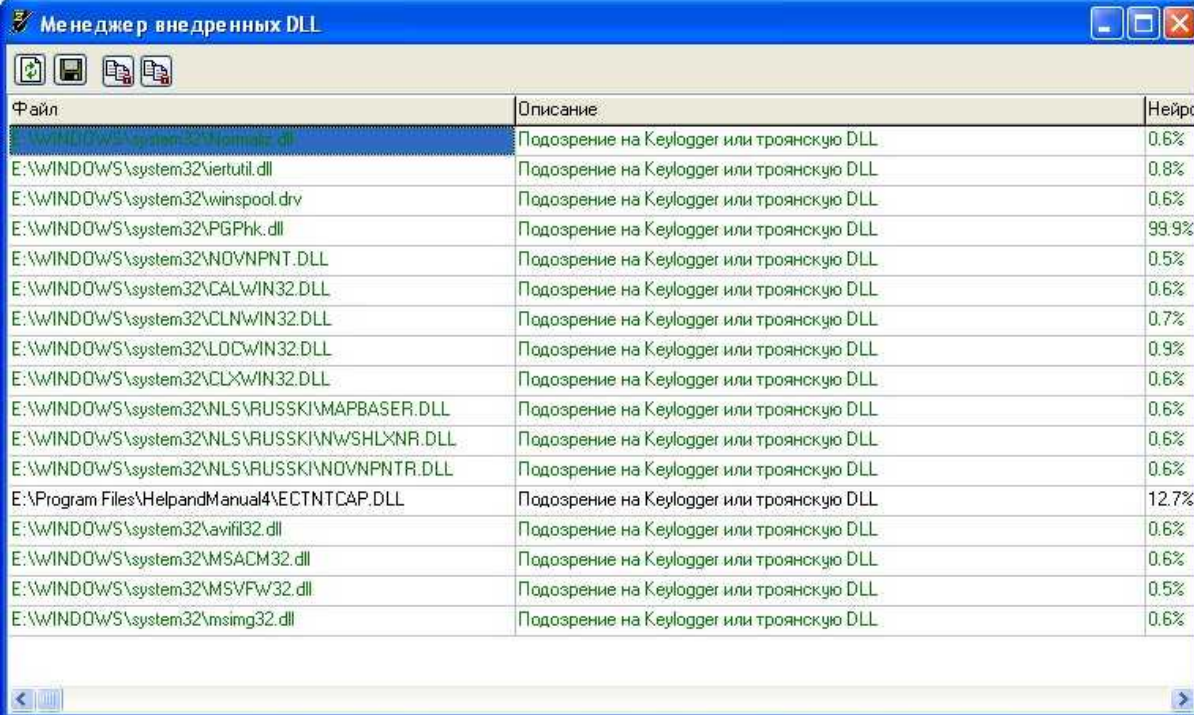
Менеджер позволяет:

- Отображать список заданий планировщика;
- Удалять задания;
- Копировать файлы в карантин;
- Формировать HTML протоколы

Анализатор менеджера планировщика заданий подключен к исследованию системы и автокарантину.

5.12 Менеджер внедренных dll

Менеджер внедренных DLL позволяет просмотреть список всех посторонних DLL, внедренных в процесс AVZ. Для каждой DLL выводится дополнительная информация (в частности, определенная нейросетью степень похожести на типовой перехватчик). Цветовая раскраска таблицы стандартна для AVZ - опознанные по базе безопасных библиотеки выделяются зеленым цветом.



Файл	Описание	Нейрс
E:\WINDOWS\system32\ntoskrnl.exe	Подозрение на Keylogger или троянскую DLL	0.6%
E:\WINDOWS\system32\iertutil.dll	Подозрение на Keylogger или троянскую DLL	0.8%
E:\WINDOWS\system32\winspool.drv	Подозрение на Keylogger или троянскую DLL	0.6%
E:\WINDOWS\system32\PGPhk.dll	Подозрение на Keylogger или троянскую DLL	99.9%
E:\WINDOWS\system32\NOVNPNT.DLL	Подозрение на Keylogger или троянскую DLL	0.5%
E:\WINDOWS\system32\CALWIN32.DLL	Подозрение на Keylogger или троянскую DLL	0.6%
E:\WINDOWS\system32\CLNWIN32.DLL	Подозрение на Keylogger или троянскую DLL	0.7%
E:\WINDOWS\system32\LOCWIN32.DLL	Подозрение на Keylogger или троянскую DLL	0.9%
E:\WINDOWS\system32\CLWIN32.DLL	Подозрение на Keylogger или троянскую DLL	0.6%
E:\WINDOWS\system32\NLS\RUSSKI\MAPBASER.DLL	Подозрение на Keylogger или троянскую DLL	0.6%
E:\WINDOWS\system32\NLS\RUSSKI\NWSHLXNR.DLL	Подозрение на Keylogger или троянскую DLL	0.6%
E:\WINDOWS\system32\NLS\RUSSKI\NOVNPNT.DLL	Подозрение на Keylogger или троянскую DLL	0.6%
E:\Program Files\HelpandManual4\ECTNTCAP.DLL	Подозрение на Keylogger или троянскую DLL	12.7%
E:\WINDOWS\system32\avifil32.dll	Подозрение на Keylogger или троянскую DLL	0.6%
E:\WINDOWS\system32\MSACM32.dll	Подозрение на Keylogger или троянскую DLL	0.6%
E:\WINDOWS\system32\MSVFW32.dll	Подозрение на Keylogger или троянскую DLL	0.5%
E:\WINDOWS\system32\msimg32.dll	Подозрение на Keylogger или троянскую DLL	0.6%

Основание назначения анализатора - поиск причины сбоев в работе программ, вызванных установкой в систему некоторых перехватчиков - он показывает все внедренные DLL, в то время как искатель троянских DLL и кейлоггеров отображает только подозрительные.

Менеджер обладает типовой для остальных менеджеров AVZ функциональностью - карантин отдельного файла или всех неопознанных файлов, сохранение HTML протокола.

5.13 Менеджер протоколов и обработчиков

Менеджер протоколов и обработчиков отображает список модулей, зарегистрированных в системе в качестве обработчиков для протоколов (например, AP GZIP Encoding/Decoding Filter) и обработчиков (Handler-s), например ftp: Asynchronous Pluggable Protocol Handler или http: Asynchronous Pluggable Protocol Handler).

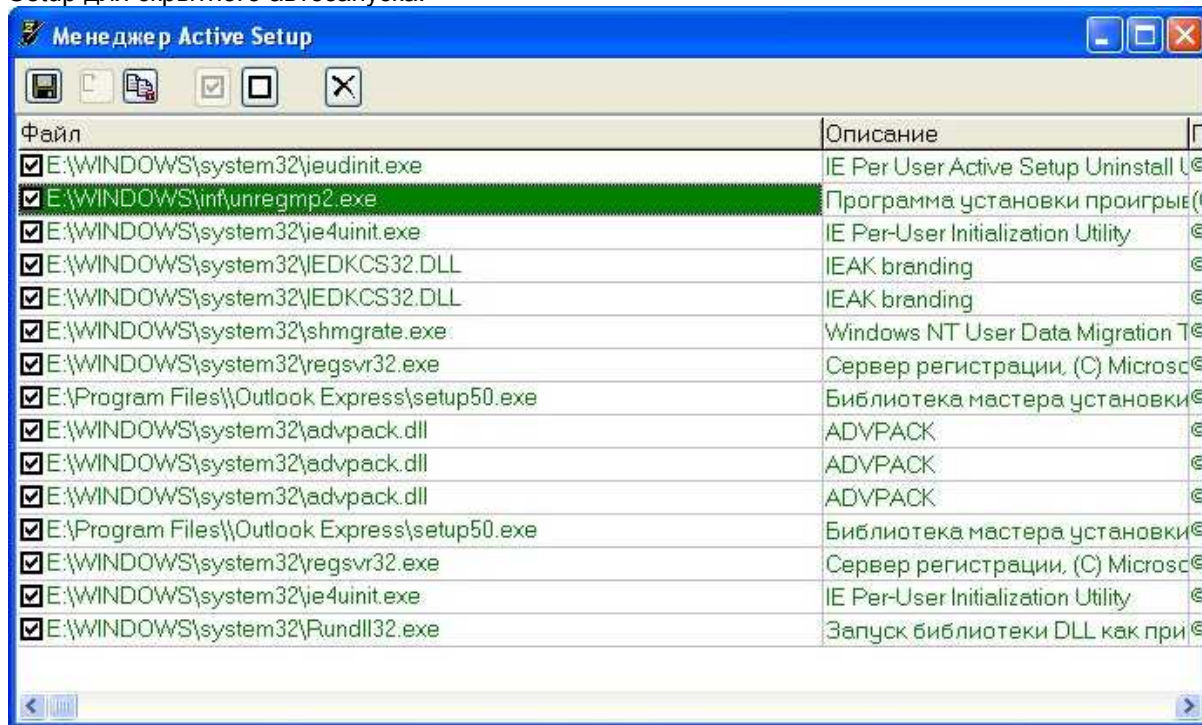
Менеджер позволяет:

- Отображать список модулей, зарегистрированных в качестве обработчиков;
- Временно отключать любой из обработчиков;
- Удалять обработчики (при этом исполняемый файл не удаляется - удаляется только регистрационная информация в реестре);
- Копировать файлы в карантин (возможно автоматическое копирование всех неопознанных файлов);
- Формировать HTML протоколы

Анализатор протоколов и обработчиков подключен к исследованию системы и автокарантину.

5.14 Менеджер Active Setup

Менеджер Active Setup отображает список приложений, зарегистрированных в качестве Active Setup инсталляций. Известен ряд вредоносных программ, применяющих регистрацию в Active Setup для скрытого автозапуска.



Менеджер Active Setup

Менеджер позволяет:

- Отображать список приложений, зарегистрированных в Active Setup;
- Временно отключать любое из приложений;
- Удалять записи Active Setup (при этом исполняемый файл не удаляется - удаляется только регистрационная информация в реестре);
- Копировать файлы в карантин (возможно автоматическое копирование всех неопознанных файлов);
- Формировать HTML протоколы

Анализатор Active Setup подключен к исследованию системы и автокарантину.

5.15 Менеджер файла Hosts

Менеджер файла Hosts позволяет просмотреть содержимое файла Hosts с возможностью удаления любой из его строк. Кроме удаления строк файла Hosts предусмотрена кнопка очистки данного файла, которая оставляет в файле только одну строку "127.0.0.1 localhost", что полезно для оперативной чистки данного файла после его повреждения троянскими программами. Анализатор менеджера файла Hosts подключен к исследованию системы.

5.16 Общие ресурсы и сетевые сеансы

Данный менеджер выполняет функции, аналогичные штатной системной утилите "Инспектор сети". В окне менеджера имеется три закладки - "Общие ресурсы", "Сетевые сеансы" и "Открытые файлы". Информация на данных закладках обновляется автоматически с интервалом 2 секунды.

Top Level Intro

This page is printed before a new
top-level chapter starts

Part

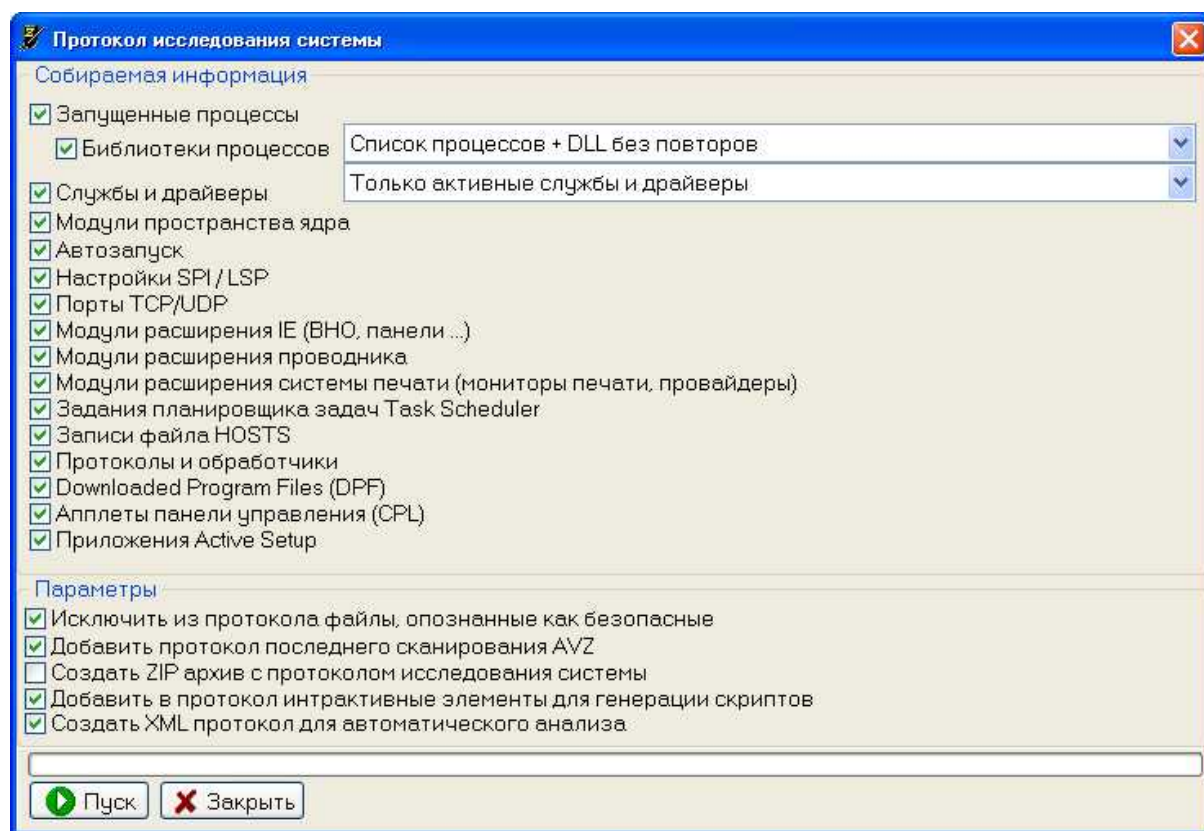
A large, light gray circle containing the Roman numeral 'VI' in white, bold, sans-serif font. The circle is positioned to the right of the 'Part' header and the horizontal line.

6 Функции анализа и восстановления

6.1 Исследование системы

Исследование системы - это функция, позволяющая изучить систему и сформировать HTML протокол, позволяющий обнаружить подозрительные файлы и программы. Исследование системы может применяться для оперативного изучения системы и (или) последующей отправки протокола для анализа в конференции вирусологов.

Запуск исследования системы производится из меню "Файл / Исследование системы", при выполнении которого выводится диалоговое окно исследования системы.



Переключатели позволяют указать, какую информацию необходимо собрать в ходе исследования системы. По умолчанию все переключатели включены и сбор информации идет по максимуму.

На исследование системы оказывает влияние антивирус (поскольку нейтрализация перехватов может привести к отображению маскируемой руткитом информации) и наличие в системе [драйвера расширенного мониторинга системы AVZPM](#)^[72].

Группа параметров "Собираемая информация"

Переключатель "Запущенные процессы"

Если он включен, то в отчет исследования системы выводится таблица со списком процессов.

Переключатель "Библиотеки процессов"

Данный переключатель доступен, если включен переключатель "Запущенные процессы". Если он включен, то в отчет исследования системы выводится таблица со списком библиотек, используемых запущенными процессами. Поддерживается два вида списка - список DLL,

совмещенный со списком процессов (после каждого процесса перечисляются его DLL) и список DLL в виде отдельной таблицы без повторов. Последний вариант выбран по умолчанию, т.к. список DLL без повторов гораздо компактнее (в этом списке есть столбец со списком PID процессов, использующих DLL)

Переключатель "Службы и драйверы"

Если он включен, то в отчет исследования системы выводится таблица со списком служб и драйверов изучаемого компьютера. Данная функция не работает на Windows 9x

Переключатель "Модули пространства ядра"

Если он включен, то в отчет исследования системы выводится таблица со списком модулей пространства ядра. Данная функция не работает на Windows 9x

Переключатель "Автозапуск"

Если он включен, то в отчет исследования системы выводится таблица со списком автозапускаемых программ. Для каждого элемента автозапуска в примечаниях указано, каким образом он стартует.

Переключатель "Настройки SPI/LSP"

Если он включен, то в отчет исследования системы выводится таблица со списком модулей расширения SPI (NSP и TSP провайдеры)

Переключатель "Порты TCP/UDP"

Если он включен, то в отчет исследования системы выводится таблица открытых портов TCP/UDP (на Windows XP и 2003 выводится информация о прослушивающем порт приложении)

Переключатель "Модули расширения IE"

Если он включен, то в отчет исследования системы выводится таблица со списком модулей расширения Internet Explorer (BHO, панели и т.п.)

Переключатель "Модули расширения проводника"

Если он включен, то в отчет исследования системы выводится таблица со списком модулей расширения проводника (explorer.exe). Модули расширения проводника регистрируются в системном реестре и известны вредоносные программы, регистрирующие себя как расширение проводника для обеспечения своего скрытого запуска

Переключатель "Модули расширения системы печати"

Если он включен, то в отчет исследования системы выводится таблица со списком модулей расширения системы печати - мониторы печати, провайдеры печати. Модуль расширения является обычной библиотекой DLL и известен ряд троянских программ, применяющих такой метод автозапуска.

Группа параметров "Параметры"

Переключатель "Исключить из протокола файлы, опознанные как безопасные"

Если он включен, то из отчета исследования системы исключаются все файлы, опознанные по базе безопасных. В большинстве случаев включение данной опции приводит к существенному сокращению размера протокола и упрощению его анализа.

Переключатель "Добавить протокол последнего сканирования AVZ"

Если он включен, то к протоколу исследования системы добавляется текущий протокол AVZ, сформированный в ходе последнего сканирования. Включение данной опции удобно в случае необходимости отправки протокола вместе с результатами исследования системы.

Переключатель "Создать ZIP архив с протоколом исследования системы"

Если данный переключатель включен, то помимо протоколов исследования автоматически создается архив формата ZIP, содержащий протокол исследования. Эта опция удобна в случае,

если протокол необходимо отправить по почте или поместить в конференцию.

Переключатель "Добавить в протокол интерактивные элементы для генерации скриптов"

Включение данного переключателя приводит к добавлению в протокол интерактивных элементов, позволяющих полуавтоматически создавать скрипты. Следует учесть, что при открытии такого протокола в IE может выводиться сообщение системы безопасности о том, что документ содержит активные элементы. Если не разрешить их использование, то протокол откроется, но интерактивные функции в нем будут недоступны.

Переключатель "Создать XML протокол для автоматического анализа"

Включение данного переключателя приводит к созданию XML файла, дублирующего HTML протокол. XML файл предназначен для использования в автоматических анализаторах протоколов.

Кнопка "Пуск"

Кнопка пуск запускает исследование системы. Перед началом исследования запрашивается имя файла для сохранения протокола. После выбора имени файла проводится исследование системы, формируется файл протокола и выводится запрос на просмотр протокола. В случае согласия пользователя протокол открывается в браузере, применяемом в системе по умолчанию.

Кнопка "Заккрыть"

Закрывает окно исследования

На заметку:

Следует учитывать, что наличие в системе антивирусного монитора может замедлить исследование и оно может длиться от 1-2 до 5 минут. Типовое время формирования составляет не более 30 сек. В процессе исследования в нижней части окна выводится прогресс-индикатор.

На заметку:

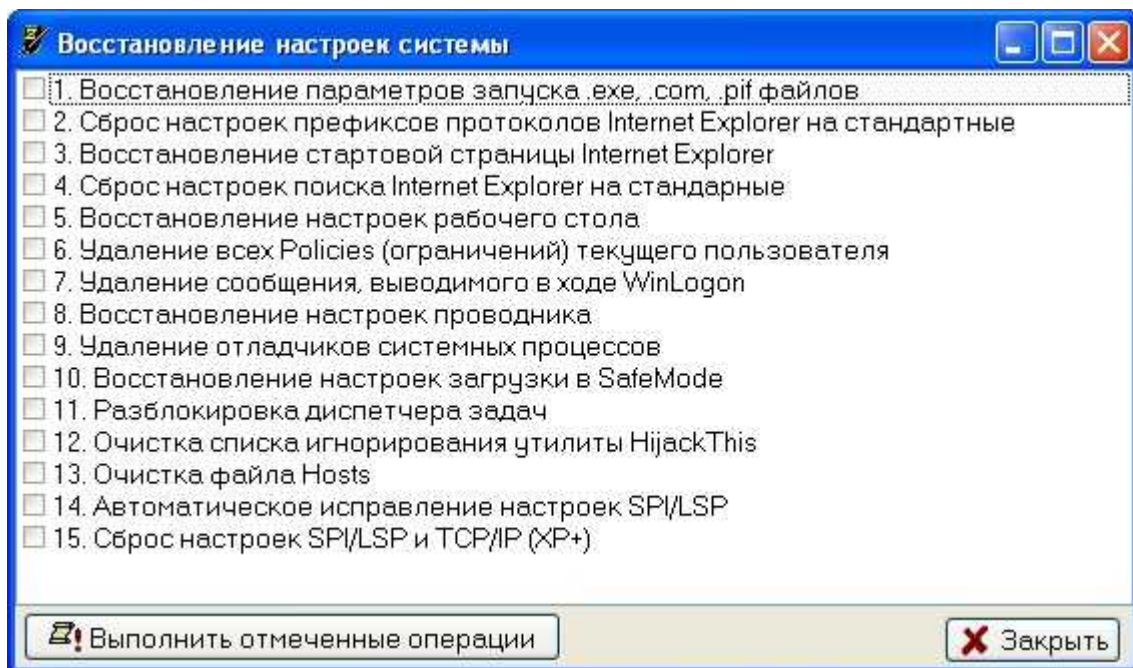
Перед запуском исследования системы рекомендуется:

1. Закрыть все неиспользуемые программы
2. Запустить браузер - это позволит анализатору изучить загруженные в его адресное пространство библиотеки.
3. Во время исследования системы не рекомендуется запускать и завершать программы, как-то работать с системой - это может повлиять на результаты анализа.

6.2 Восстановление системы

Восстановление системы - это особая функция AVZ, которая позволяет восстановить ряд системных настроек, поврежденных вредоносными программами.

Микропрограммы восстановления системы хранятся в антивирусной базе и обновляются по мере необходимости.



Восстановление системы

В настоящее время в базе есть следующие микропрограммы:

1. Восстановление параметров запуска .exe, .com, .pif файлов

Данная микропрограмма восстанавливает реакцию системы на файлы exe, com, pif, scr.

Показания к применению: после удаления вируса перестают запускаться программы.

2. Сброс настроек префиксов протоколов Internet Explorer на стандартные

Данная микропрограмма восстанавливает настройки префиксов протоколов в Internet Explorer

Показания к применению: при вводе адреса типа www.yandex.ru идет его подмена на что-то вида www.seque.com/abcd.php?url=www.yandex.ru

3. Восстановление стартовой страницы Internet Explorer

Данная микропрограмма восстанавливает стартовую страницу в Internet Explorer

Показания к применению: подмена стартовой страницы

4. Сброс настроек поиска Internet Explorer на стандартные

Данная микропрограмма восстанавливает настройки поиска в Internet Explorer

Показания к применению: При нажатии кнопки "Поиск" в IE идет обращение к какому-то постороннему сайту

5. Восстановление настроек рабочего стола

Данная микропрограмма восстанавливает настройки рабочего стола. Восстановление подразумевает удаление всех активных элементов ActiveDesktop, обоев, снятие блокировок на меню, отвечающее за настройки рабочего стола.

Показания к применению: Исчезли закладки настройки рабочего стола в окне "Свойства:экран", на рабочем столе отображаются посторонние надписи или рисунки

6. Удаление всех Policies (ограничений) текущего пользователя

Windows предусматривает механизм ограничений действий пользователя, называемый Policies. Этой технологией пользуются многие вредоносные программы, поскольку настройки хранятся в реестре и их несложно создавать или модифицировать.

Показания к применению: Заблокированы функции проводника или иные функции системы.

7.Удаление сообщения, выводимого в ходе WinLogon

Windows NT и последующие системы в линейке NT (2000, XP) позволяют установить сообщение, отображаемое в ходе автозагрузки. Этим пользуется ряд вредоносных программ, причем уничтожение вредоносной программы не приводит к уничтожению этого сообщения.

Показания к применению: В ходе загрузки системы вводится постороннее сообщение.

8.Восстановление настроек проводника

Данная микропрограмма сбрасывает ряд настроек проводника на стандартные (сбрасываются в первую очередь настройки, изменяемые вредоносными программами).

Показания к применению: Изменены настройки проводника

9.Удаление отладчиков системных процессов

Регистрация отладчика системного процесса позволяют осуществить скрытый запуск приложения, что и используется рядом вредоносных программ

Показания к применению: AVZ обнаруживает неопознанные отладчики системных процессов, возникают проблемы с запуском системных компонент, в частности после перезагрузки исчезает рабочий стол.

10.Восстановление настроек загрузки в SafeMode

Некоторые вредоносные программы, в частности червь Bagle, повреждают настройки загрузки системы в защищенном режиме. Данная микропрограмма восстанавливает настройки загрузки в защищенном режиме.

Показания к применению: Компьютер не загружается в защищенном режиме (SafeMode).

Применять данную микропрограмму следует *только в случае проблем с загрузкой в защищенном режиме*.

11.Разблокировка диспетчера задач

Блокировка диспетчера задач применяется вредоносными программами для защиты процессов от обнаружения и удаления. Соответственно выполнение данной микропрограммы снимает блокировку.

Показания к применению: Блокировка диспетчера задач, при попытке вызова диспетчера задач выводится сообщение "Диспетчер задач заблокирован администратором".

12.Очистка списка игнорирования утилиты HijackThis

Утилита HijackThis хранит в реестре ряд своих настроек, в частности - список исключений. Поэтому для маскировки от HijackThis вредоносной программе достаточно зарегистрировать свои исполняемые файлы в списке исключений. В настоящий момент известен ряд вредоносных программ, использующих данную уязвимость. Микропрограмма AVZ выполняет очистку списка исключений утилиты HijackThis

Показания к применению: Подозрения на то, что утилита HijackThis отображает не всю информацию о системе.

13. Очистка файла Hosts

Очистка файла Hosts сводится к поиску файла Hosts, удалению из него всех значащих строк и добавлению стандартной строки "127.0.0.1 localhost".

Показания к применению: Подозрения на то, файл Hosts изменен вредоносной программой. Типичные симптомы - блокировка обновления антивирусных программ. Проконтролировать содержимое файла Hosts можно при помощи менеджера Hosts файла, встроенного в AVZ.

14. Автоматическое исправление настроек SPI/LSP

Выполняет анализ настроек SPI и в случае обнаружения ошибок производит автоматическое исправление найденных ошибок. Данную микропрограмму можно запускать повторно неограниченное количество раз. После выполнения данной микропрограммы рекомендуется перезагрузить компьютер. *Обратите внимание ! Данную микропрограмму нельзя запускать из терминальной сессии*

Показания к применению: После удаления вредоносной программы пропал доступ в Интернет.

15. Сброс настроек SPI/LSP и TCP/IP (XP+)

Данная микропрограмма работает только в XP, Windows 2003 и Vista. Ее принцип работы основан на сбросе и пересоздании настроек SPI/LSP и TCP/IP при помощи штатной утилиты netsh, входящей в состав Windows. Подробно про сброс настроек можно прочитать в базе знаний Microsoft - <http://support.microsoft.com/kb/299357> **Обратите внимание ! Применять сброс настроек нужно только в случае необходимости при наличии неустранимых проблем с доступом в Интернет после удаления вредоносных программ !**

Показания к применению: После удаления вредоносной программы пропал доступ в Интернет и выполнение микропрограммы "14. Автоматическое исправление настроек SPI/LSP" не дает результата.

16. Восстановление ключа запуска Explorer

Восстанавливает системные ключи реестра, отвечающие за запуск проводника.

Показания к применению: В ходе загрузки системы не запускается проводник, но запуск explorer.exe вручную возможен.

17. Разблокировка редактора реестра

Разблокирует редактор реестра путем удаления политики, запрещающей его запуск.

Показания к применению: Невозможно запустить редактор реестра, при попытке выводится сообщение о том, что его запуск заблокирован администратором.

18. Полное пересоздание настроек SPI

Выполняет резервное копирование настроек SPI/LSP, после чего уничтожает их и создает по эталону, который хранится в базе.

Показания к применению: Тяжелые повреждения настроек SPI, неустранимые скриптами 14 и 15. **Применять только в случае необходимости !**

Для выполнения восстановления необходимо отметить один или несколько пунктов и нажать кнопку "Выполнить отмеченные операции". Нажатие кнопки "ОК" закрывает окно.

На заметку:

Восстановление бесполезно, если в системе работает троянская программа, выполняющая подобные перенастройки - необходимо сначала удалить вредоносную программу, а затем восстанавливать настройки системы

На заметку:

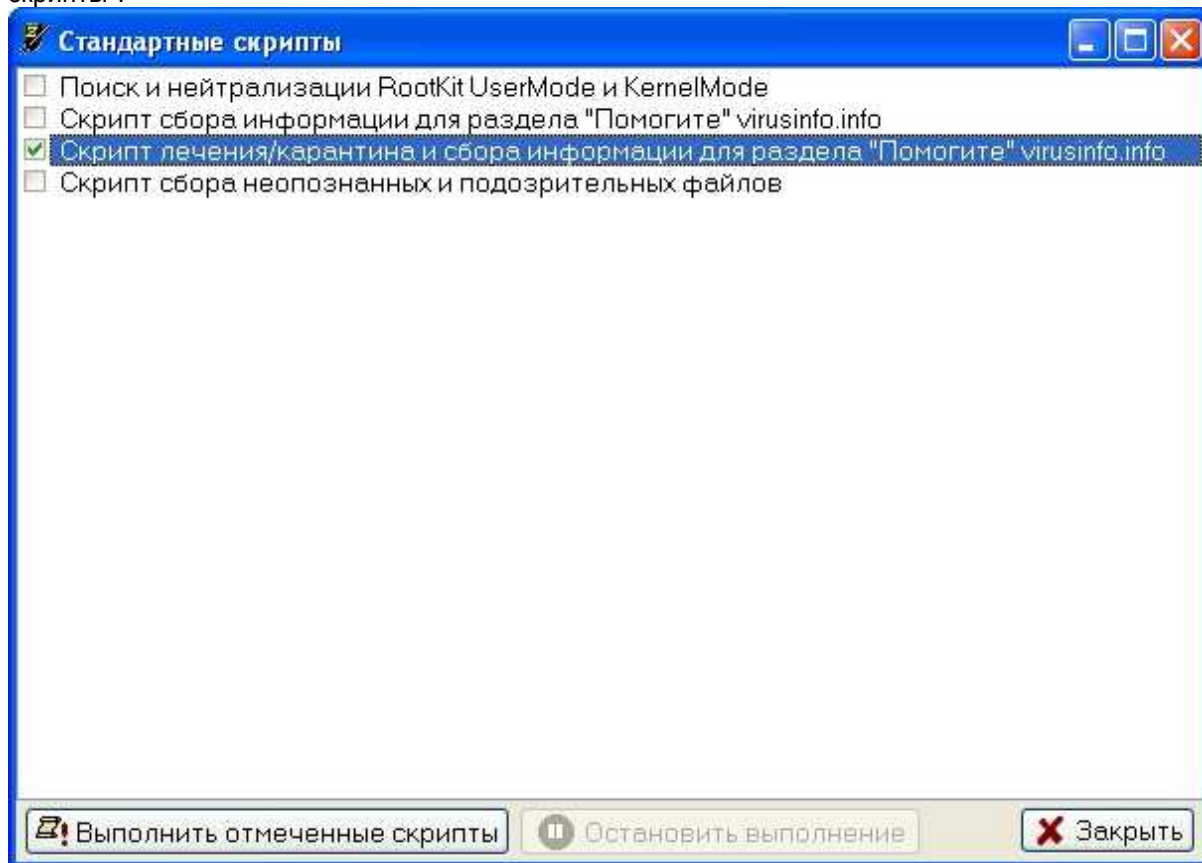
Для устранения следов большинства Hijacker необходимо выполнить три микропрограммы - "Сброс настроек поиска Internet Explorer на стандартные", "Восстановление стартовой страницы Internet Explorer", "Сброс настроек префиксов протоколов Internet Explorer на стандартные"

На заметку:

Любую из микропрограмм можно выполнять несколько раз подряд без ущерба для системы. Исключения - "5. Восстановление настроек рабочего стола" (работа этой микропрограммы сбросит все настройки рабочего стола и придется заново выбирать расцветку рабочего стола и обои) и "10. Восстановление настроек загрузки в SafeMode" (данная микропрограмма пересоздает ключи реестра, отвечающие за загрузку в безопасном режиме).

6.3 Стандартные скрипты

Стандартные скрипты предназначены для автоматизации основных операций, выполняемых при помощи AVZ. Они хранятся в базе AVZ и обновляются при помощи автоматического обновления. Вызов окна "Стандартные скрипты" производится из меню "Файл/Стандартные скрипты".



Для выполнения одного или нескольких стандартных скриптов необходимо отметить их в списке и затем нажать кнопку "Выполнить отмеченные скрипты". Окно можно закрыть только после завершения последнего из выбранных скриптов - это сделано специально для блокировки органов управления AVZ на время выполнения стандартных скриптов.

Для аварийной остановки процесса выполнения скриптов необходимо нажать кнопку "Остановить выполнение". Следует учитывать, что остановка происходит не мгновенно - с момента нажатия кнопки до остановки может пройти несколько секунд. Это нормальное явление, т.к. остановка некоторых операций в ходе их выполнения невозможна.

В настоящий момент поддерживаются следующие операции:

- 1. Поиск и нейтрализации RootKit UserMode и KernelMode.** Выполняет поиск руткитов с нейтрализацией всех обнаруженных перехватов
- 2. Скрипт сбора информации для раздела "Помогите" virusinfo.info.** Выполняет операции по проверке компьютера (без лечения) и исследования системы, после чего создает папку LOG в рабочем каталоге AVZ и помещает туда протокол и архив с заподозренными файлами
- 3. Скрипт лечения/карантина и сбора информации для раздела "Помогите" virusinfo.info.** Выполняет операции по проверке компьютера с лечением и выполняет исследование системы, после чего создает папку LOG в рабочем каталоге AVZ и помещает туда протокол и архив с заподозренными файлами
- 4. Скрипт сбора неопознанных и подозрительных файлов.** Выполняет две операции - сканирование системы с карантином всех заподозренных файлов + автоматический карантин с максимальными настройками. В результате в карантин попадают файлы, которые заподозрены AVZ или не опознаны по базе безопасных объектов.

5. Обновление баз с автоматической настройкой. Производит обновление баз, используя различные настройки. Данная операция полезна в случае, если обновление стандартным способом не проходит и выводится сообщение об ошибке.

6. Удаление всех драйверов и ключей реестра AVZ. Производит автоматическое удаление всех драйверов и ключей реестра, которые могли быть созданы в процессе работы AVZ. В обычной ситуации это не требуется, так как AVZ автоматически удаляет ключи и файлы, установленные в систему. Является аналогом деинсталляции и рекомендуется к применению на ПК после завершения использования AVZ

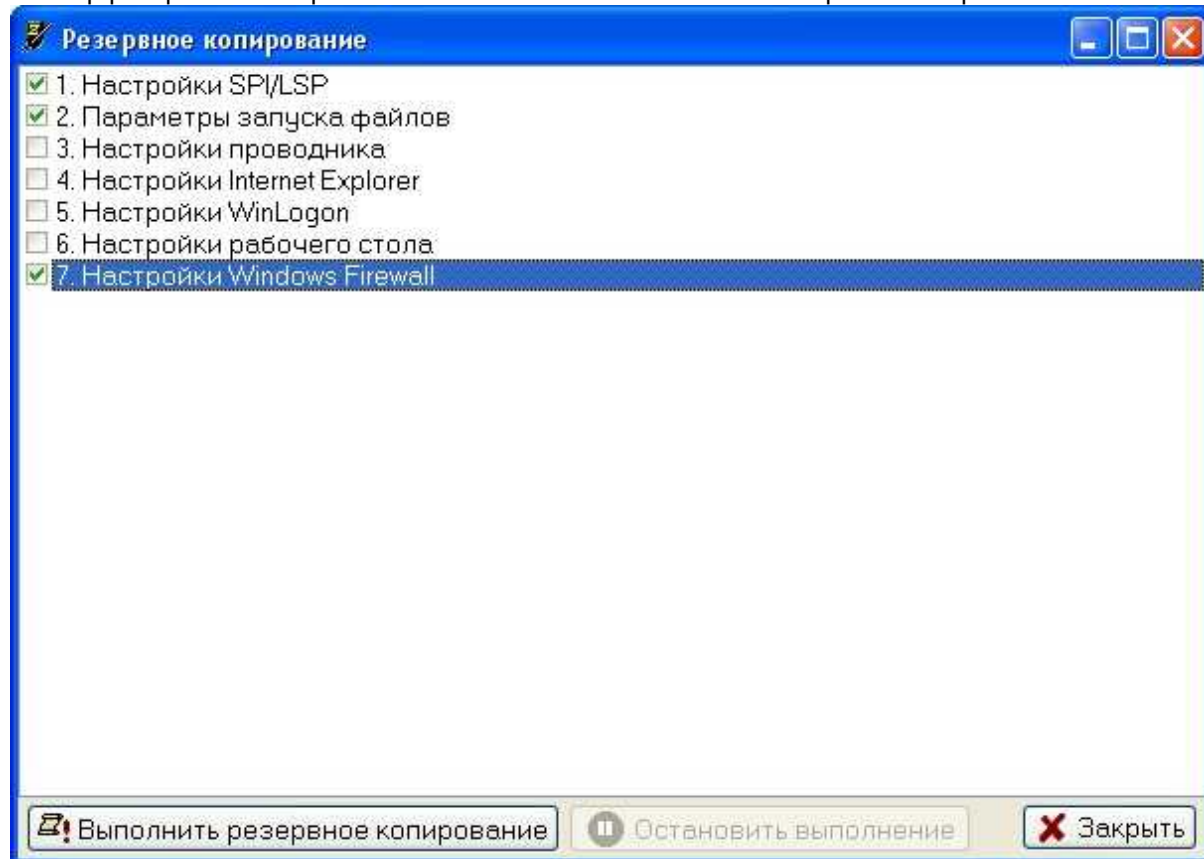
Номера скриптов уникальны и не меняются при дополнении и обновлении базы. Стандартные скрипты могут вызываться из скриптов пользователя при помощи функции [ExecuteStdScr](#)^[13]

6.4 Резервное копирование

Начиная с версии 4.16 в AVZ поддерживается три вида копирования:

- Автоматического резервное копирование. Выполняется AVZ автоматически по мере необходимости при выполнении некоторых операций, в частности модификации настроек LSP, настроек запуска файлов, политик безопасности, настроек IE и проводника. Эти копии помещаются в папку Backup\DD-MM-YYYY
- Ручное резервное копирование. Вызывается из меню "Файл\Резервное копирование", результаты так-же сохраняются в папке Backup\DD-MM-YYYY
- Резервное копирование под управлением скрипта. В этом случае результаты резервного копирования могут сохраняться в папке Backup\DD-MM-YYYY или в любой другой папке по усмотрению разработчика скрипта.

Мастер резервного копирования вызывается из меню "Файл\Резервное копирование"



Для выполнения резервного копирования следует отметить одну или несколько позиций списка и нажать кнопку "Выполнить резервное копирование". Нажатие кнопки "Остановить выполнение"

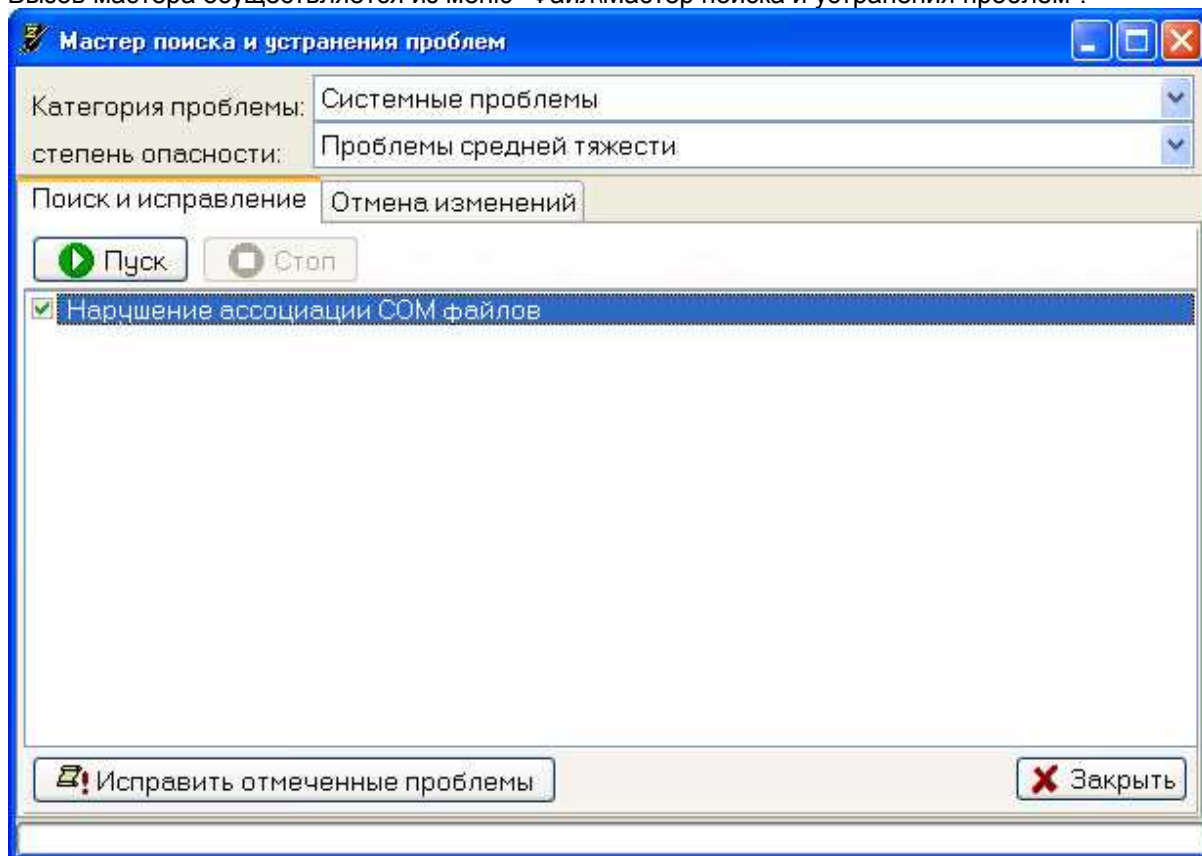
позволяет аварийно прервать выполнение резервного копирования. Процедуры резервного копирования хранятся в обновляемой базе, поэтому список резервируемых настроек может расширяться. После выполнения резервного копирования рекомендуется сохранить созданные AVZ файлы в надежном месте.

Восстановление резервной копии выполняется запуском соответствующего REG файла - после этого система предложит импортировать содержимое REG файла в реестр.

6.5 Мастер поиска и устранения проблем

Мастер поиска и устранения проблем предназначен для автоматического анализа системы, обнаружения устранимых ошибок и аномалий и их устранения. Мастер во многом аналогичен восстановлению системы AVZ, его главное отличие от восстановления состоит в том, что он автоматически ищет проблемы (в восстановлении системы пользователь должен сам принять решение о том, какие функции восстановления задействовать) и производит более точное и гибкое устранение проблем.

Вызов мастера осуществляется из меню "Файл\Мастер поиска и устранения проблем".



Для начала работы с мастером требуется выбрать категорию проблемы. Поддерживаются следующие категории:

- Системные проблемы. Это всевозможные ошибки в реестре или ключи, созданные/измененные вредоносными программами для нарушения или ограничения функциональности системы
- Настройки и твики браузера. Выполняет анализ настроек браузера, ищет опасные настройки (например, разрешение загружать ActiveX без запроса) или предлагает модифицировать настройки для повышения защищенности браузера

- Приватность. Позволяет осуществить чистку системы с целью удаления следов работы пользователя (временные файлы, cookies, кеш и журналы браузеров, всевозможные протоколы)

После выбора категории следует задать степень опасности проблемы. Анализатор поддерживает три степени опасности - от незначительных проблем до опасных. Степень опасности по сути является порогом срабатывания анализатора

После настройки степени опасности следует нажать кнопку "Пуск" и дождаться завершения процесса анализа (прогресс-индикатор в нижней части окна показывает ход процесса). Найденные проблемы или рекомендуемые операции отображаются в виде списка с возможностью отметки одной или нескольких позиций. Для выполнения процедуры исправления следует отметить один или несколько пунктов в списке и нажать кнопку "Исправить отмеченные проблемы".

Исправление некоторых настроек может негативно сказаться на функционировании системы. На этот случай в мастере поиска и исправления проблем предусмотрена функция записи данных для отката изменений. Для выполнения отката следует выбрать категорию проблемы, после чего перейти на закладку "Отмена изменений". На данной закладке отобразится список отмены - для отмены следует пометить один или несколько пунктов и затем нажать кнопку "Отменить отмеченные изменения".

На заметку:

Система отмены изменений действует только на реестр - удаление файлов (например кукизов или кешей браузера) является необратимой операцией

На заметку:

Базы данных системы отмены изменений хранятся в папке BackUp. Если данная папка недоступна для записи, то мастер поиска будет успешно работать, но данные для отмены изменений записываться не будут. Кроме того, следует учесть, что в случае применения мастера для удаления приватных данных следует также удалить базы системы отмены изменений.

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



VII

7 Подсистема AVZGuard

7.1 О технологии

Технология AVZGuard основана на KernelMode драйвере, который разграничивает доступ запущенных приложений к системе. Драйвер может функционировать в системах, основанных на платформе NT (начиная с NT 4.0 и заканчивая Vista Beta 1). Основное назначение - борьба с трудноудаляемыми вредоносными программами, которые активно противодействуют процессу лечения компьютера.

В момент активации все приложения делятся на две категории - доверенные и недоверенные. На доверенные приложения драйвер не оказывает никакого влияния, в то время как недоверенным запрещаются следующие операции:

- Создание, модификация и удаление параметров реестра
- Создание файлов с расширениями *.exe, *.dll, *.sys, *.ocx, *.scr, *.cpl, *.pif, *.bat, *.cmd на любом диске
- Обращение к устройствам \device\rawip, \device\udp, \device\tcp, \device\ip
- Доступ к device\physicalmemory (что блокирует операции с физической памятью из UserMode)
- Установка драйверов (является следствием блокировки работы с реестром)
- Запуск процессов
- Открытие запущенных процессов с уровнем доступа, допускающим его остановку или запись в его адресное пространство
- Открытие потоков других процессов (при этом недоверенному процессу не запрещается открывать и останавливать свои потоки)

Исходно доверенным является только AVZ, но из меню "AVZGuard\Запустить приложение как доверенное" можно запустить любое приложение. Запускаемое таким образом приложение получает статус доверенного. По умолчанию для доверенных приложений действует принцип наследования доверительных отношений - все запускаемые доверенным приложением процессы так-же считаются доверенными.

Назначение:

Основным назначением системы является:

- Борьба с трудноудаляемыми троянскими программами, которые восстанавливают ключи реестра и удаленные файлы, запускают остановленные процессы или иными способами препятствуют своему удалению. Это основное назначение системы;
- Защита доверенных приложений от недоверенных. Позволяет защитить AVZ и запущенные им доверенные приложения от воздействия работающих вредоносных программ;
- Распространение действия антируткита UserMode AVZ на другие процессы. Пример - утилиты типа VBA Console scanner, DrWeb Cure IT, HijackThis и т.п. не обладают функциями детектирования и нейтрализации руткитов. В этом случае можно запустить AVZ, провести нейтрализацию руткитов для его процесса, а затем включить AVZGuard и запустить тот-же DrWeb Cure IT как доверенное приложение. В этом случае драйвер AVZGuard возьмет на себя функцию защиты запущенного процесса, и в том числе не позволит руткиту модифицировать его. Естественно, данная технология не является панацеей от всех видов UserMode руткитов, но основные их типы (в частности Hacker Defender) могут быть нейтрализованы подобным образом.

Рекомендации:

- Перед включением системы необходимо закрыть все приложения кроме AVZ. Это важный момент, поскольку все запущенные приложения в момент запуска начнут считаться недоверенными и это может заблокировать их работу. На самом деле экспериментально установлено, что большинство приложений сохраняют свою функциональность, так как не совершают блокируемых системой действий. В частности, Internet Explorer, Outlook Express, TheBat продолжают нормально работать в качестве недоверенных приложений;

- Особенностью блокировки операций с реестром является то, что операция блокируется, но приложение получает статус успешного выполнения и считает, что реестр был изменен. Это сделано специально для совместимости с рядом программ, которые в случае ошибки записи в реестр начинают работать неадекватно. Логике этой блокировки легко изучить при помощи Regedit - его необходимо запустить до включения AVZGuard, далее включить AVZGuard, и с помощью RegEdit изменить значение какого-либо ключа реестра. С точки зрения regedit операция пройдет успешно, но если обновить данный при помощи F5, то можно убедиться, что реестр не изменился.
- В случае, если проводилось лечение системы, необходимо перезагрузиться не выключая AVZGuard. Это важный момент, связанный с тем, что в системе после лечения могут быть загружены вредоносные DLL, будут продолжать работу троянские потоки и т.п.
- Управление AVZGuard идет из контекста утилиты AVZ, в случае завершения работы AVZ контроль над AVZGuard будет потерян и он будет функционировать автономно до перезагрузки.
- Система AVZGuard может многократно включаться/отключаться в процессе работы с AVZ по мере необходимости в ограничении работы запущенных процессов

Особенности выключения ПК при включенном AVZGuard

- Процесс выключения и перезагрузки при активной системе AVZGuard может занять до 2-3 минут. Это связано с тем, что система не может принудительно закрыть процессы.
- Некоторые приложения в момент завершения могут выдавать сообщение о ошибках, связанные с ограничением их деятельности
- Сам AVZ невозможно закрыть по Alt-F4 или при помощи кнопки в заголовке окна. Для завершения работы AVZ при активном AVZGuard необходимо применить пункт меню "Файл/Выход"

Поведение различных программ при включенном AVZGuard непредсказуемо, поэтому настоятельно рекомендуется закрыть все приложения перед включением AVZGuard и перезагрузиться после лечения с его использованием. Важно отметить, что AVZGuard не является системой проактивной защиты или антивирусным монитором - активировать его нужно только на время борьбы с трудноудаляемыми malware.

7.2 Управление системой

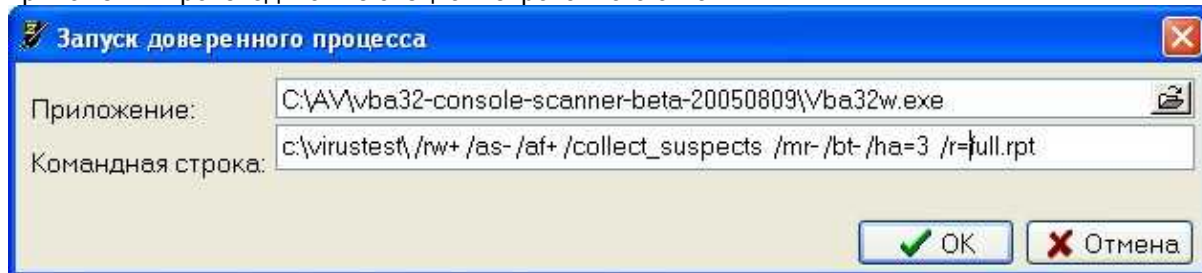
Управление системой AVZ Guard производится из главного меню.

Включение AVZGuard

Пункт меню "AVZGuard\Включить AVZGuard" включает систему. По результатам активации выводится диалоговое окно с сообщением о успешном включении или о ошибке. После включения системы пункт меню "Включить AVZGuard" становится неактивным.

Запуск доверенных приложений

Пункт меню "AVZGuard\Запустить приложение как доверенное" производит запуск указанного приложения в качестве доверенного. На доверенное приложение не распространяется действие AVZGuard и драйвер защищает доверенное приложение от удаления или модификации. Запуск приложения производится из специализированного окна.



Если запускаемое приложение находится в системной папке или путь к нему задан в строке

PATN, то допускается указывать только имя исполняемого файла. Поле "командная строка" позволяет указать командную строку запускаемого приложения (необязательно к заполнению). Кнопка в правой части поля "Приложение" позволяет вызвать стандартный диалог выбора файла.

Отключение AVZGuard

Пункт меню "AVZGuard\Отключить AVZGuard" производит отключение системы AVZGuard. Перед отключением выводится запрос подтверждения операции отключения. После выключения AVZGuard все работающие приложения вновь получают полный контроль над системой, драйвер выгружается.

Добавление работающего процесса в список доверенных

В AVZGuard предусмотрена возможность добавления работающего процесса в список доверенных. Выполнять данную операцию не рекомендуется, т.к. запущенный до включения AVZ Guard процесс может содержать троянские потоки или руткит-перехватчики. Тем не менее в случае необходимости данная операция выполняется из диспетчера процессов AVZ - при включенном AVZ Guard всплывающее меню таблицы процессов содержит подпункт "AVZGuard - считать процесс доверенным".

Включение AVZ Guard при запуске AVZ

Если в проверяемой системе работает противодействующее работе AVZ вредоносное приложение, то включение AVZGuard рекомендуется осуществить при помощи ключа командной строки "AG=Y". Данный ключ анализируется непосредственно в момент запуска AVZ, до инициализации его рабочих окон.

Управление из скриптов

Для управления подсистемой AVZGuard из скриптов предусмотрены функции

[SetAVZGuardStatus](#)^[118] и [GetAVZGuardStatus](#)^[118]

7.3 AVZGuard и антируткит

После включения AVZGuard антируткит режима ядра блокируется, а антируткит режима UserMode продолжает нормальную работу. Особенностью AVZGuard является возможность распространения действия UserMode антируткита AVZ на другие процессы. Для этого необходимо действовать по следующей схеме:

1. Закрыть все приложения
2. Запустить AVZ, включить противодействие руткитам UserMode и KernelMode и пролечить систему
3. Включить AVZGuard (это приведет к автоматическому отключению антируткита KernelMode)
4. Для надежности повторно пролечить систему (при этом в сущности производится проверка, удалены ли перехваты UserMode из процесса AVZ)
5. При помощи меню "AVZGuard\Запустить приложение как доверенное" запустить любое приложение, которое мы хотим запустить в защищенном от руткита режиме. С высокой степенью вероятности запущенное таким образом приложение не будет поражено руткитом.

Эту особенность очень удобно применять для запуска по описанному выше алгоритму антивирусных приложений, которые не оснащены функцией защиты от руткитов. Экспериментально установлено, что по описанному алгоритму можно нейтрализовать HackerDefender и аналогичные ему руткиты UserMode.

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



VIII

8 Подсистема AVZPM

8.1 О технологии

Технология AVZPM основана на KernelMode Boot драйвере, который осуществляет слежение за запуском процессов и загрузкой драйверов. Драйвер написан в строгом соответствии с MSDN и рекомендациями Microsoft для минимизации его воздействия на систему и снижения вероятности конфликта с другим антивирусным программным обеспечением.

Назначение:

- Мониторинг запуска/остановки процессов
- Мониторинг загрузки/выгрузки драйверов
- Ведение списков процессов и загруженных модулей пространства ядра, независимых от системных.

Собираемая драйвером информация используется различными системами AVZ:

- Антируткитом в ходе проверки системы. Если в системе доступен драйвер мониторинга, то собранные драйвером данные применяются для поиска маскирующихся процессов и драйверов, а так-же искажений в системных структурах (подмена PID, модификация имени модуля и т.п.)
- Диспетчером процессов AVZ - получаемые от драйвера сведения применяются для отображения маскирующихся процессов
- Диспетчером "Модули пространства ядра" - получаемые от драйвера сведения применяются для отображения маскирующихся драйверов
- Исследованием системы - получаемые от драйвера сведения применяются в ходе построения отчета по драйверам и модулям пространства ядра

Применение подобного драйвера является эффективным и документированным путем борьбы с многими DKOM руткитами. Как известно, основная проблема борьбы с DKOM руткитом состоит в том, что он модифицирует системные структуры в памяти и вносит в них заведомо ложную и некорректную информацию. Классический пример - модификация имени процесса и его PID в структуре EPROCESS, практикуемая многими DKOM руткитами. В результате при желании можно обнаружить маскирующийся процесс, но невозможно определить, откуда он запустился, каково имя исполняемого файла и реальный PID его процесса. Аналогично дело обстоит с драйверами - несложно обнаружить маскируемый драйвер в памяти, но практически невозможно вычислить его имя, поскольку грамотно построенный руткит просто уничтожит эту информацию в процессе маскировки. Мониторинг системных событий решает эту проблему - анализатор получает возможность опираться на собственные данные, а не на искаженные руткитом структуры ядра.

Совместимость

Драйвер мониторинга может применяться совместно с антируткитом и системой AVZ Guard AVZ, а так-же с антивирусными мониторами и HIPS системами других производителей.

8.2 Управление системой

Инсталляция и включение AVZPM

Для включения AVZPM необходимо выполнить пункт меню "AVZPM/Установить драйвер расширенного мониторинга процессов". Выполнение этого пункта приводит к установке в систему драйвера мониторинга, его регистрации в реестре и загрузке. Драйвер начинает функционировать с момента загрузки, но для получения полной картины о всех запущенных процессах необходима перезагрузка.

После установки драйвера (или в случае обнаружения драйвера при последующих запусках AVZ) пункт меню "Установить драйвер расширенного мониторинга процессов" автоматически блокируется и разблокируется пункт меню "Удалить драйвер расширенного мониторинга процессов".

Деинсталляция и отключение AVZPM

Для выключения AVZPM необходимо выполнить пункт меню "AVZPM/Удалить драйвер расширенного мониторинга процессов". Выполнение этого пункта приводит к выгрузке драйвера, удалению его регистрационных ключей из реестра и файла с диска. Операция остановки мониторинга и выгрузки драйвера может занять несколько секунд. Перезагрузка после удаления драйвера не требуется.

После удаления драйвера пункт меню "Удалить драйвер расширенного мониторинга процессов" автоматически блокируется.

Управление из скриптов

Для управления подсистемой AVZ PM из скриптов предусмотрены функции [SetAVZPMStatus](#)^[117] и [GetAVZPMStatus](#)^[118]

Top Level Intro

This page is printed before a new
top-level chapter starts

Part

IX

9 Подсистема Boot Cleaner

9.1 О технологии

Технология Boot Cleaner основана на KernelMode Boot драйвере, который выполняет заданную последовательность операций в момент загрузки системы. После выполнения заданных операций драйвер автоматически самоуничтожается. Основное назначение - борьба с трудноудаляемыми вредоносными программами, пересоздающими свои ключи реестра и файлы, или блокирующие доступ к ним. Драйвер системы Boot Cleaner размещен в AV базе и обновляется в ходе автоматического обновления.

Назначение:

- Удаление файлов
- Удаление ключей реестра
- Удаление драйверов и служб

Перечисленные операции выполняются в ходе загрузки системы.

Boot Cleaner является альтернативой отложенному удалению по ряду причин:

1. В Boot Cleaner предусмотрена возможность ведения протоколов. В протоколе отмечаются все выполняемые операции и фиксируется код статуса (0 - успешно, >0 - код ошибки)
2. Выполнение сценария Boot Cleaner происходит на раннем этапе загрузки системы, что повышает его эффективность
3. Boot Cleaner может удалять ключи реестра (и в частности драйверы и службы), в то время как отложенное удаление позволяет оперировать только с файлами.

Совместимость

Драйвер Boot Cleaner может применяться совместно с антивирусом, системами AVZ Guard и AVZ PM, а также с антивирусными мониторами и HIPS системами других производителей.

Управление системой

Управление системой производится средствами скриптового языка AVZ.

Набор команд и примеры подробно описаны в разделе [Управление Boot Cleaner](#)^[119]

Протоколы

В процессе работы BootCleaner может формировать текстовые протоколы. Имя и местоположение протокола задается пользователем при помощи команды. Рассмотрим пример скрипта (на момент выполнения скрипта в системе установлен драйвер PE386 и в папке Windows существует файл trojan1.exe):

begin

```
// Постановка задания
BC_DeleteSvc('PE386');
BC_DeleteFile('%Windir%\trojan1.exe');
BC_DeleteFile('%Windir%\trojan2.exe');
BC_LogFile(GetAVZDirectory + 'boot_clr.log');
// Активация
BC_Activate;
// Перезагрузка
RebootWindows(true);
```

end.

После перезагрузки будет сформирован протокол:

```
-- AVZ Boot cleaner log --
DeleteFile \??\C:\WINDOWS\trojan1.exe - succeeded
DeleteFile \??\C:\WINDOWS\trojan2.exe - failed (0xC0000034)
```

```
Delete File \??\C:\WINDOWS\system32\lzx32.sys - succeeded
Delete Service & File PE386 - succeeded
-- End --
```

Структура протокола достаточно проста - в каждой строке указывается операция, имя обрабатываемого в ходе операции объекта и статус. Возможно два статуса:

- succeeded - операция выполнена успешно
- failed (0xNNNNNNN) - в ходе выполнения операции возникли ошибки, в скобках указывается код ошибки.

Boot Cleaner позволяет удалять службы и драйверы, содержащие в имени символы с кодом 0 и любые Unicode символы. При формировании протокола символ с кодом 0 заменяется на "*", а не имеющие аналогов в ANSI Unicode символы заменяются на знаки "?"

Удаление файлов, ключей реестра, драйверов и служб при помощи Boot Cleaner может нанести существенный вред системе, поэтому применяйте Boot Cleaner только если Вы уверены в правильности своих действий !

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



X

10 Ревизор

10.1 О технологии

Встроенный в AVZ ревизор диска работает по классической схеме и предполагает выполнение двух операций

- Изучение имеющихся на диске файлов и построение базы данных, содержащих информацию о этих файлах.
- Сравнение текущего состояния диска с базой данных. Подобное сравнение позволяет обнаружить, какие файлы/папки были созданы, изменены или удалены с момента создания базы
- Сравнение может вестись в стандартном режиме (сравниваются размер файла и его контрольная сумма) и скоростном режиме (сравнивается только размер). Скоростной режим удобен для оперативной проверки диска.

Особенности:

Ревизор AVZ обладает несколькими особенностями, отличающими его от аналогов:

1. База может размещаться на любом носителе и может храниться где угодно (например, на Flash диске или в сетевой папке). Для достижения такой мобильности в базе хранятся все настройки, необходимые для проведения сравнения.
2. Для одного компьютера можно создать неограниченное количество баз, сформировав несколько "снимков" в разные моменты времени или с разными настройками
3. После формирования база сжимается для экономии места. База с результатами анализа папки Windows системы Windows XP занимает на диске около 60 кб, в реальной системе - 50-150 кб (в среднем получается 10 кб базы на 1000 файлов). Небольшой объем базы позволяет хранить ее на Flash диске
4. Встроенный ревизор AVZ связан с базой безопасных файлов, что позволяет исключать из протокола сравнения файлы, опознанные по базе безопасных AVZ или по каталогу безопасности Microsoft
5. Ревизор защищен встроенным антируткитом AVZ

Назначение:

Основным назначением ревизора является:

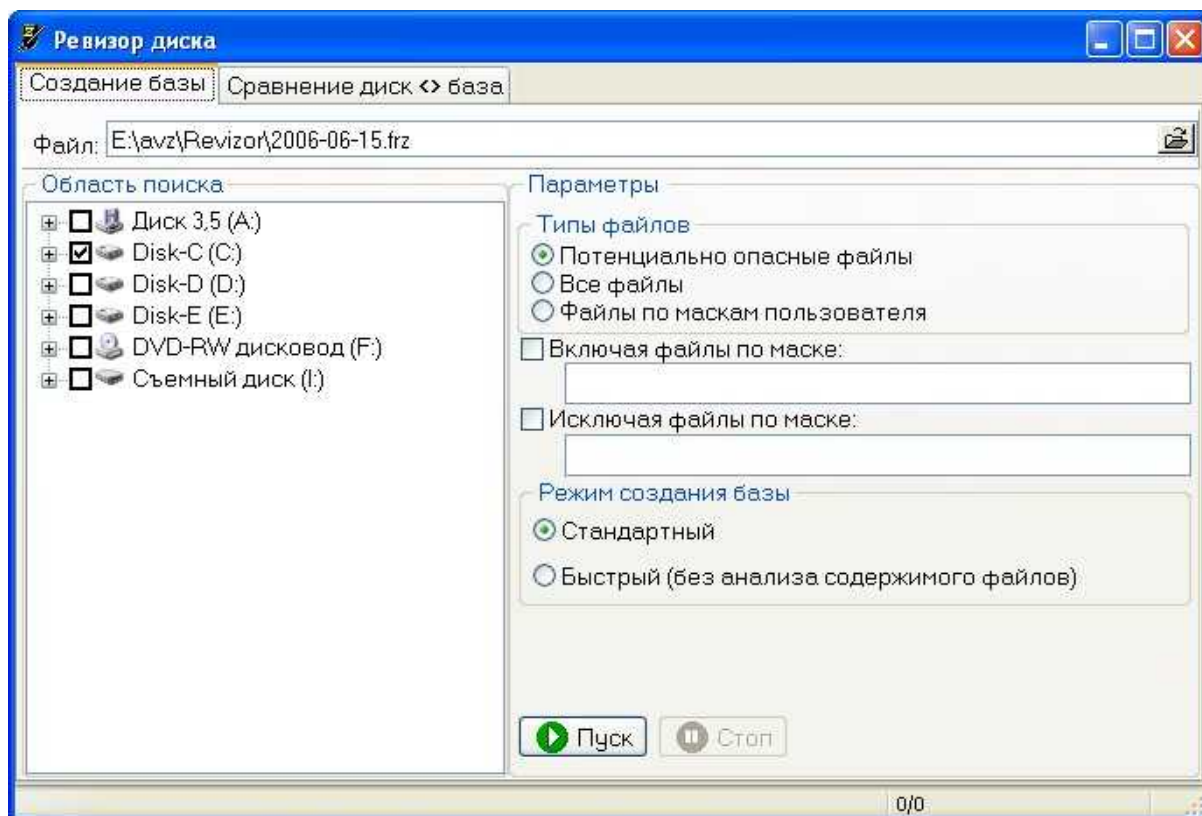
1. Поиск модификаций на диске защищаемого компьютера. Позволяет установить, какие файлы были созданы, изменены или удалены с момент построения базы. Кроме того, регистрируется создание и удаление каталогов.
2. Поиск файлов, маскируемых руткитом. Для выполнения данной операции необходимо
 - 2.1 создать базу ревизора, включив в нее системную папку и папки, в которых предполагается наличие руткитов. Построение базы можно вести в быстром режиме для экономии времени)
 - 2.2 выполнить сканирование компьютера с включенным противодействием руткитам
 - 2.3 не выходя из AVZ произвести сравнение текущего состояния с базой, построенной на шаге 2.1.
3. Сравнение содержимого некоторой папки на двух идентичных компьютерах

Области применения:

1. Периодический контроль состояния ПК. Для этого необходимо создать базу для системного диска и хранить ее в надежном месте. Далее с некоторой периодичностью можно сравнивать текущее состояние диска с базой
2. Поиск руткитов
3. Отслеживание изменений на ПК пользователей в корпоративной сети. В этом случае базы ревизора можно хранить централизованно на сервере или ПК администратора

10.2 Создание базы

Для создания базы необходимо вызвать окно ревизора (меню Файл/Ревизор) и перейти на закладку "Создание базы".



Древовидный список папок "Область поиска" позволяет указать, какие папки должны быть исследованы в ходе создания базы. По умолчанию отмечена системная папка. Можно отметить несколько папок или весь диск (или соответственно несколько дисков) целиком. При этом следует учитывать, что размер файла и время анализа напрямую зависят от количества анализируемых файлов.

В поле "Файл" необходимо задать имя файла базы данных. По умолчанию этот путь указывает на папку Revizor в рабочем каталоге AVZ и имя формируется из текущей даты, однако его можно изменить.

Переключатель "Режим создания базы".

Очень важный переключатель, влияющий на формирования базы. Поддерживается два режима:

- **Стандартный.** Этот режим включен по умолчанию, в нем для каждого анализируемого файла производится расчет контрольной суммы. Эта достаточно длительная операция, однако наличие контрольной суммы позволяет регистрировать изменение содержимого файла. Кроме контрольной суммы в базе сохраняется размер файла, его атрибуты и дата/время
- **Быстрый.** В быстром режиме фиксируется только размер файла, его атрибуты и дата/время, а контрольная сумма не вычисляется. Это на 2-3 порядка ускоряет процесс создания базы, но теряется возможность отслеживания изменений файла, не приводящих к изменению его размера. Рекомендуется только для экспресс диагностики

Переключатель "Типы файлов".

По умолчанию выбран пункт "Потенциально опасные файлы" - в этом случае ревизор анализирует только файлы с определенными расширениями (EXE, DLL, OCX, SYS, CAB, INF ...) - т.е. исполняемые файлы или файлы, которые могут содержать данные для установки вредоносных программ или опасные скрипты.

Выбор пункта "Все файлы" приводит к анализу абсолютно всех файлов, независимо от их расширения. В этом режиме блокируется переключатель "Включая файлы с расширением", т.к. в данном случае он теряет смысл. Включать данный режим без необходимости не рекомендуется, так как это замедлит создание базы и приведет к увеличению ее размера. Выбор пункта "Файлы по маскам пользователя" приводит к тому, что поиск ведется только по маскам, заданным пользователем в полях "Включая файлы по маске" и "Исключая файлы по маске".

Переключатель "Включая файлы с расширениями"

Переключатель "Включая файлы по маске:" позволяет включить в поиск файлы, соответствующие заданной пользователем маске. Этот переключатель теряет смысл в режиме "Все файлы" и при выборе этого режима автоматически выключается и становится недоступным.

Переключатель "Исключая файлы с расширениями"

Переключатель "Исключая файлы по маске:" позволяет исключить из проверки файлы, соответствующие заданной пользователем маске. Исключение файлов по маске может использоваться совместно с включением по маске или режимов "Все файлы". Указание одинаковых элементов в списке включения и исключения не является ошибкой, исключение имеет приоритет над включением - например, при указании включения в сканирование файлов *.vlg и исключения *.vlg приоритетно исключение. Более того, исключение по более общей маске доминирует над включением, например при включении trojan*.exe и исключении *.exe файл с именем trojan.exe будет исключен из проверки

На заметку: Поля с маской могут включать несколько масок, разделенных запятой, пробелом или символом ";". В маске могут присутствовать символы "?" (любой символ в позиции знака ?) и * (любые символы).

. - поиск всех файлов

*.exe - поиск файлов с расширением exe

dialer.exe - поиск файла с конкретным именем "dialer.exe"

dialer.exe, *.dll, trojan*.sys - поиск файлов с именем dialer.exe или файлов с любым именем и расширением DLL или файлов с именем Trojan<любые символы>.sys

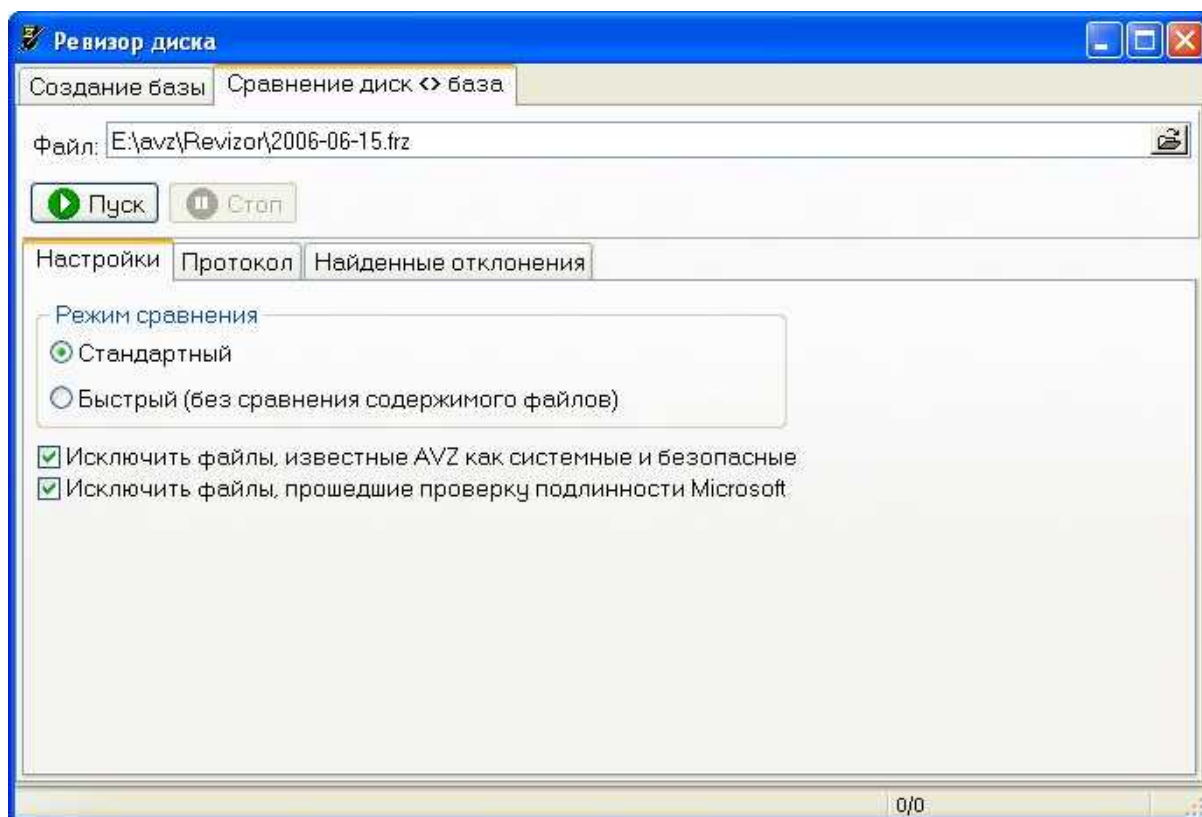
*.exe, *.dll, *.sys, *.ocx - поиск файлов с указанными расширениями

Для создания базы необходимо нажать кнопку "Пуск", для досрочной аварийной остановки - кнопку "Стоп" (в случае досрочной остановки файл не создается).

Как уже отмечалось в описании технологии, для одного компьютера можно создать неограниченное количество баз (для различных папок и дисков, а разных режимах и т.п.). Естественно, что при создании нескольких баз придется дать им различные имена.

10.3 Сравнение

Для сравнения текущего состояния диска с базой необходимо вызвать окно ревизора (меню Файл/Ревизор) и перейти на закладку "Сравнение диск <> база".



В **поле "Файл"** необходимо задать имя существующей базы данных ревизора. Нажатие кнопки **"Пуск"** запускает сравнение, нажатие кнопки **"Стоп"** - прерывает его (при этом протокол не очищается).

Закладка "Настройки"

Содержит настройки сравнения. В настройках нет возможности задать маски или выбрать каталоги, так как эти настройки уже были заданы в момент создания базы ревизора и сохраняются в базе данных.

Переключатель "Режим сравнения".

Очень важный переключатель, влияющий на процесс сравнения состояния диска с базой. Поддерживается два режима:

- **Стандартный.** Этот режим включен по умолчанию, в нем для каждого анализируемого файла производится расчет контрольной суммы и сравнение файла ведется по его размеру и содержимому. Эта достаточно длительная операция, однако наличие контрольной суммы позволяет регистрировать изменение содержимого файла. Кроме контрольной суммы в базе сохраняется размер файла, его атрибуты и дата/время
- **Быстрый.** В быстром режиме фиксируются сравниваются только размеры файла. Рекомендуется только для экспресс диагностики

На заметку

Важно отметить, что если база создавалась в стандартном режиме, то сравнение возможно в стандартном и быстром режимах. Если база создавалась в быстром режиме, то сравнение возможно **только в быстром режиме**, если в момент загрузки базы после нажатия кнопки "Пуск" обнаружится, что она создана в быстром режиме, то режим сравнения автоматически переключится в положение "Быстрый" независимо от выбор пользователя.

Переключатели "**Исключить файлы, известные AVZ как системные и безопасные**" и "**Исключить файлы, прошедшие проверку подлинности Microsoft**" позволяют уменьшить размер протокола с результатами сравнения путем выполнения соответствующей проверки для созданных и измененных файлов. В некоторых случаях (например, после установки обновлений и сервиспаков) включение данных переключателей существенно уменьшает размер протокола.

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



XI

11 Категории вредоносных программ

11.1 Вирусы

В эту категорию попадают все вредоносные программы, не относящиеся к остальным категориям, в частности:

- Собственно вирусы, т.е. вредоносные программы, обладающие способностью заражать файлы, т.е. приписывать к ним свой исполняемый код и модифицировать зараженную программу таким образом, чтобы при запуске зараженной программы машинный код вируса получал управление;
- сетевые и почтовые черви (I-Worm, Net-Worm, Worm, Email-Worm),
- троянские программы (Trojan, Trojan-Spy, Trojan-Downloader, Trojan-Dropper),
- утилиты скрытного удаленного управления (Backdoor).

11.2 AdWare

В эту категорию включены программы и модули, предназначенные для несанкционированной загрузки и отображения на компьютере пользователя рекламной информации. AdWare могут быть самостоятельными программами или различными модулями расширения браузера.

11.3 Spy или SpyWare

Как следует из названия, SpyWare - это программа-шпион. Как правило, эта категория очень тесно переплетается с AdWare (по многим классификациям AdWare считается подклассом SpyWare). Чисто формально это так и есть - AdWare в процессе своей работы "посещают" сайты, с которых они закачивают рекламную информацию. В протоколах этих сайтов отмечается периодичность обмена, фиксируется IP адрес компьютера пользователя. При использовании в AdWare стандартных API функций для загрузки файлов из Интернет в заголовке запроса будет фигурировать версия браузера пользователя и прочие параметры.

Задачей SpyWare является сбор информации о пользователе и скрытная передача этой информации на сайт разработчиков. Стоит сразу отметить, что в отличие от троянских программ SpyWare не передает пароли, номера кредитных карт и иную информацию, на основании которой можно в последствии нанести серьезный вред пользователю. Как правило, SpyWare передает данные о конфигурации компьютера, установленном программном обеспечении, посещаемых пользователем URL и вводимых поисковых запросах и т.п. Эта информация в большинстве случаев используется для маркетинговых исследований и передаче пользователю целевой рекламной информации (т.е. рекламы, которая по тематике связана с его кругом интересов).

Многие SpyWare представляют собой панели, расширяющие возможности браузера. Подобное решение очень удобно для шпионажа за работой пользователя в Интернет - SpyWare может отслеживать посещаемые URL и вмешиваться в работу браузера. При этом он сам не является отдельной задачей Windows, что затрудняет его обнаружение. Кроме того, обмен подобного SpyWare информацией с Интернет ведется из контекста браузера, и большинство Firewall не блокируют его.

Еще одной неприятной особенностью многих SpyWare и AdWare является способность периодической проверки своих обновлений на сайте разработчика. При обнаружении обновления производится его скрытная загрузка (на что затрачивается трафик) и установка.

11.4 PornWare и Dialer

В категорию PornWare включено все, имеющее отношение к порносайтам. В первую очередь это Dialer (или в просторечии порнозвонилка) - особая программа для автоматического дозвона на порносайты и (или) перенастройки имеющихся соединений удаленного доступа для набора номера модемного пула владельцев порносайта вместо номера телефона местного провайдера.

Известно множество случаев, когда в результате работы Dialer пользователям приходили внушительные счета за пользование платными услугами или междугородные (международные) звонки.

Кроме того, в эту категорию включаются разнообразные панели для браузера и прочие программы, устанавливающиеся при посещении порносайтов.

11.5 Hijacker

Hijacker - это программа, которая выполняет несанкционированную пользователем перенастройку системы, чаще всего браузера. Задачей такой перенастройки является принудительная смена стартовой страницы, страницы поиска, соответствия префиксов протоколов и т.п. Наиболее знаменитым Hijacker вероятнее всего является Sexque. Кроме того, в категорию Hijacker принято включать ключи реестра, переадресующие браузер на посторонние WEB страницы.

11.6 RiskWare

RiskWare - это программное обеспечение, которое не является вирусом или вредоносной программой, но применение которого может нанести вред пользователю (его частной информации, документам и т.п.) или операционной системе.

Типичным примером являются многие клавиатурные шпионы (Keylogger) - это зачастую коммерческие продукты, снабженные инсталлятором, деинсталлятором, документацией и справочной системой. Более того, многие из них устанавливаются и конфигурируются исключительно вручную. Однако работа таких программ может принести существенный вред пользователю - он может быть куплен у производителя злоумышленником, установлен на Ваш компьютер и настроен на работу в скрытом режиме. Опираясь на полученную с помощью клавиатурного шпиона информацию, злоумышленник может нанести очень существенный вред.

Второй типичный пример - утилита Remote Admin, позволяющая удаленно управлять компьютером. С одной стороны это полезная программа, с другой - установка этой программы на Ваш компьютер, произведенная злоумышленником, может привести к утечке информации - полезная программа превращается в backdoor.

Вывод - уничтожать такие программы в большинстве случаев не следует. Необходимо разобраться, откуда программы категории RiskWare появились на RiskWare компьютере и нужны ли они для работы.

При формировании названия известных вирусов и вредоносных программ для вирусов за основу берется система имен лаборатории Касперского (причина кроется в толковой классификации и наличии энциклопедии компьютерных вирусов).

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



XII

12 Дополнительная информация

12.1 Что такое RootKit

Термин RootKit исторически пришел из мира Unix, и под этим термином понимается набор утилит, которые хакер устанавливает на взломанном им компьютере после получения первоначального доступа. Этот набор, как правило, включает в себя разнообразные утилиты для заметания следов вторжения в систему, хакерский инструментарий (снифферы, сканеры) и троянские программы, замещающие основные утилиты Unix. RootKit позволяет хакеру закрепиться во взломанной системе и скрыть следы своей деятельности.

В системе NT/W2K/XP RootKit принято считать программу, которая внедряется в систему и перехватывает системные функции (API). Перехват и модификация низкоуровневых API функций в первую очередь позволяет такой программе достаточно качественно маскировать свое присутствие в системе. Кроме того, как правило, RootKit может маскировать присутствие в системе любых описанных в его конфигурации процессов, папок и файлов на диске, ключей в реестре. Многие RootKit устанавливают в систему свои драйверы и сервисы (они естественно также являются "невидимыми").

Самостоятельно обнаружить запущенный RootKit крайне сложно - он не виден в стандартном диспетчере процессов, его ключи реестра не отображаются в редакторе реестра Regedit, файлы невидны в Explorer и других программах просмотра диска. Для обнаружения RootKit применяются специальные методики, причем наибольшую сложность обычно составляет не обнаружение RootKit, а корректное восстановление пораженных им функций в памяти без перезагрузки компьютера для проведения поиска и уничтожения RootKit.

Если Вы не уверены в том, что перехват функций произведен полезной программой, то можно попробовать включить противодействие RootKit. В результате AVZ попытается восстановить исходную работу перехваченных функций, противодействие распространяется только на процесс AVZ и не должен сказываться на работе системы в целом.

Антивирус AVZ может противодействовать распространенным типам RootKit. Однако стоит отметить, что процесс борьбы с запущенным RootKit в памяти может привести к зависанию программы AVZ, других программ и системы в целом. Поэтому противодействие RootKit нужно включать, закрыв другие приложения и завершив работу антивирусного монитора и Firewall (последнее обязательно, т.к. мониторы антивирусов и Firewall часто перехватывают API функции).

Подробнее про RootKit и методики перехвата API функций можно прочитать в моей статье "RootKit - принципы и механизмы работы", опубликованной в журнале КомпьютерПресс № 5.2005 или в разделе ["Сетевая и информационная безопасность"](#) моего сайта.

12.2 Эвристический анализатор AVZ

Антивирусная программа производит поиск вирусов и вредоносных объектов на основании сравнения исследуемой программы со своей базой данных с описаниями вирусов. При обнаружении соответствия антивирус может производить лечение найденного вируса, причем правила и методики лечения обычно хранятся в той же базе данных.

Однако эта база данных становится уязвимым местом антивируса - он может обнаруживать только вирусы, описанные в его базе данных. Частично устранить эту проблему позволяет эвристический анализатор - специальная подсистема антивируса, которая пытается обнаружить новые разновидности вирусов, не описанные в базе данных. Кроме вирусов эвристический анализатор AVZ пытается обнаружить шпионское ПО, Hijacker и троянские программы.

Работа эвристического анализатора основана на поиске характерных для вирусов и

шпионских программ особенностей (фрагментов программного кода, определенных ключей реестра, файлов и процессов). Кроме того, эвристический анализатор пытается оценить степень похожести исследуемого объекта на известные вирусы.

Для поиска шпионского ПО, RootKit и Hijacker наиболее эффективен эвристический анализ не отдельно взятых файлов на диске, а всей системы в целом. При этом анализируется совокупность данных в реестре, файлов на диске, процессов и библиотек в памяти, прослушиваемых TCP и UDP портов, активных сервисов и загруженных драйверов.

Особенностью эвристического анализа является достаточно высокий процент ошибок - эвристика может сообщить об обнаружении подозрительных объектов, но эта информация нуждается в проверке специалистами-вирусологами. В результате проверки объект признается вредоносным и включается в базы или фиксируется ложное срабатывание и в алгоритмы эвристического анализатора вводится поправка.

В большинстве антивирусов (в том числе и в AVZ) имеется возможность регулировки чувствительности эвристического анализатора. При этом всегда возникает противоречие - чем выше чувствительность, тем выше вероятность обнаружения эвристикой неизвестного вредоносного объекта. Но при увеличении чувствительности возрастает вероятность ложных срабатываний, поэтому нужно искать некую "золотую середину". Эвристический анализатор имеет несколько ступеней чувствительности и два особых режима:

- блокировка работы эвристического анализатора. При этом анализатор полностью выключается из работы. В AVZ кроме регулировки уровня чувствительности эвристического анализатора есть возможность включать и выключать эвристический анализ системы;
- "параноидальный" режим - в этом режиме включается максимально возможная чувствительность и предупреждения выводятся при малейшем подозрении. Этот режим естественно не приемлем из-за очень высокого количества ложных срабатываний, но он иногда полезен.

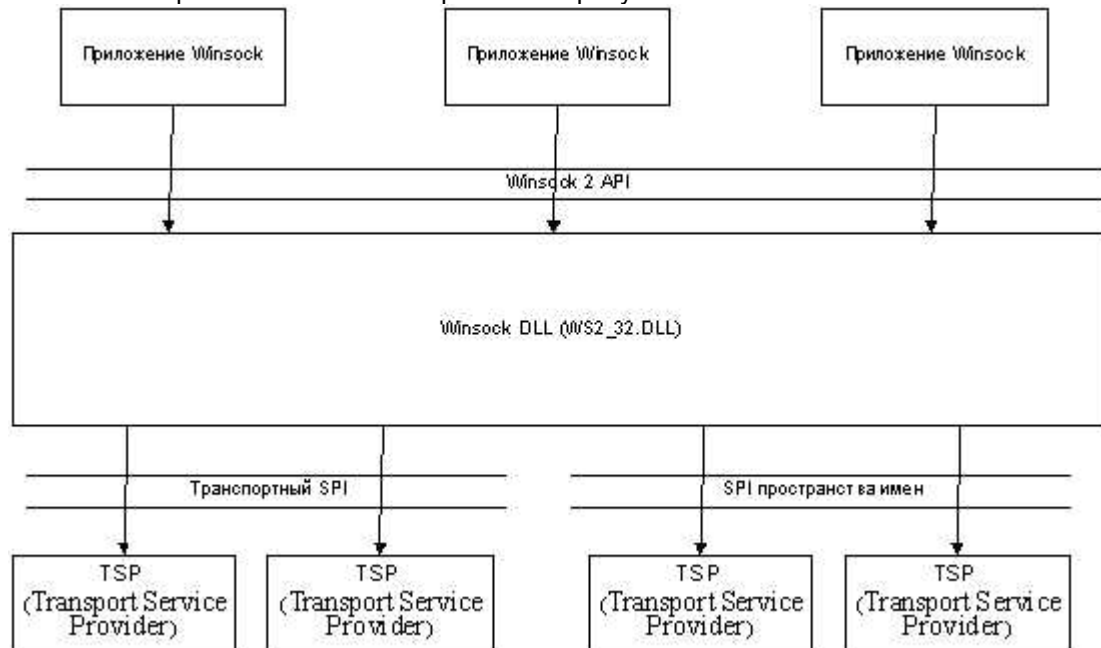
Основные сообщения эвристического анализатора AVZ приведены в следующем списке:

- **"Имя файла >>> подозрение на имя_вируса (краткие данные об объекте)"** Подобное сообщение выдается при обнаружении объекта, который по мнению AVZ похож на известный вредоносный объект. Данные в скобках позволяют разработчику найти в базе антивируса запись, которая привела к выдаче данного сообщения;
- **"Имя файла >>> PE файл с нестандартным расширением"** - это означает, что обнаружен программный файл, но вместо типичного расширения EXE, DLL, SYS он имеет другое, нестандартное расширение. Это не опасно, но многие вирусы маскируют свои PE файлы, давая им расширения PIF, COM. Данное сообщение выводится в любом уровне эвристики для PE файлов с расширением PIF, COM, для остальных - только при максимальном уровне эвристики;
- **"Имя файла >>> В имени файла больше 5 пробелов"** - множество пробелов в имени файла - это редкость, однако многие вирусы применяют пробелы для маскировки реального расширения, создавая файлы с именами типа "photo.jpeg.exe";
- **"Имя файла >>> Обнаружена маскировка расширения"** - аналогично предыдущему сообщению, но выдается при обнаружении более 15 пробелов в имени;
- **"Имя файла >>> файл не имеет видимого имени"** - выдается для файлов, не имеющих имени (т.е. имя файла имеет вид ".exe" или ".pif");
- **"Процесс Имя файла может работать с сетью"** - выводится для процессов, которые используют библиотеки типа wininet.dll, rasapi32.dll, ws2_32.dll - т.е. системные библиотеки, содержащие функции для работы с сетью или управления процессом набора номера и установления соединения. Данная проверка производится только при максимальном уровне эвристики. Факт использования сетевых библиотек естественно не является признаком вредоносности программы, но обратить внимание на непонятные процессы в этом списке стоит;

После сообщения может выводиться цифра, которая представляет собой степень опасности в процентах. На файлы, для которых выдана степень опасности более 30, следует обратить особое внимание.

12.3 Local Service Provider (LSP/SPI)

Мне часто задавали вопросы, связанные с функционированием сети, понятиями SPI, LSP, NSP ... поэтому я решил создать этот раздел справочной системы и подробно расписать базовую теорию и то, как вредоносные программы могут повредить работу Winsock. Условно схема работы Winsock отображена на рисунке:



Winsock 2 имеет как бы две "зеркальные стороны" - Winsock API и Winsock SPI. Winsock API предназначен для реализации архитектуры открытых систем Windows (WOSA), обладающей стандартными API интерфейсами между Winsock и использующими его приложениями. SPI - это интерфейс между Winsock и поставщиками служб Winsock.

Поставщики услуг SPI бывают двух типов:

- Поставщик пространства имен (Namespace Provider, NSP). Задача поставщика пространства имен - независимое от протокола разрешение имен, т.е. преобразование дружественного имени в зависимый от протокола адрес. Классическим примером может являться DNS, который преобразует дружественное для пользователя имя `www.z-oleg.com` в IP адрес сервера, на котором размещен данный сайт;
- Поставщик транспорта (Transport Service Provider, TSP). TSP - это службы, предоставляющие функции установления связи, передачи данных, управления потоком, обработки ошибок и т.п. Поставщики транспортной службы бывают двух видов - базовые и многоуровневые. Базовые (Base) поставщики реализуют конкретные детали сетевого транспортного протокола (типа TCP/IP), включая базовые сетевые функции протокола, такие как отправка и получение данных по сети. Многоуровневые (Layered) поставщики реализуют только высокоуровневые функции связи нуждаются в базовом поставщике для фактического обмена по сети;

Как видно из рисунка, поставщиков NSP и TSP может быть несколько (если точнее - то практически неограниченно много). Еще более интересно то, что список используемых LSP и TSP хранится в специальной базе данных (которая размещена в реестре). Более того, в

Winsock API есть функции, которые позволяют программе не только получать списки LSP и TSP, но и регистрировать своих поставщиков.

С точки зрения физической реализации провайдер SPI является обычной DLL, разработанной по определенному стандарту - в частности, эта DLL обязана экспортировать ряд функций для взаимодействия с Winsock.

С одной стороны, хранение списка поставщиков и их параметров в базе данных и наличие функции для управления этой базой является очень грамотным шагом - любое приложение может установить свои провайдеры SPI, расширяя возможности Winsock. При этом прикладной программе совершенно не обязательно знать с тем, какие конкретно провайдеры SPI и как именно обеспечивают разрешение имен и обмен по сети. Получается универсальная система, и многие программы используют эти возможности - при помощи AVZ нетрудно исследовать настройки SPI и посмотреть, какие провайдеры там зарегистрированы - в списке провайдеров можно обнаружить Firewall, антивирусы, разнообразные антишпионские программы, утилиты для борьбы с рекламой, учета трафика и т.п.

Однако теперь давайте посмотрим на SPI глазами разработчика вирусов и SpyWare/AdWare программ. Получается, что установив 2-3 провайдера можно взять под контроль весь обмен компьютера с сетью. При помощи своего поставщика пространства имен вредоносная программа может отслеживать запросы на разрешение имени (собственно для шпионажа) и вмешиваться в этот процесс. Упрощенно говоря, к примеру вы запрашиваете адрес сайта www.yandex.ru, а "вражеский" NSP выдаст вместо этого IP адрес скажем адрес www.никому.не.известный поисковик.com Аналогичные "безобразия" можно устраивать на уровне поставщика транспорта - SpyWare чаще всего устанавливают свои Layered Service Providers - LSP. Получается, что открытость и универсальность SPI Winsock становится его уязвимым местом - создатели вредоносного ПО знают это и активно используют. Примеров очень много - Spy.NewDotNet (not-a-virus:AdWare.NewDotNet), Spy.SAHAgent, Spy.WebHancer, Trojan.Riler, Adware.SpywareNuker, Backdoor.Kika ... - список можно продолжать достаточно долго. Кроме вмешательства в работу Winsock выполнение backdoor-компоненты в виде SPI имеет множество преимуществ - скрытый запуск, отсутствие видимого процесса.

Возможность установки вредоносной в качестве SPI провайдера - это еще только половина проблемы. Оказывается, что у Winsock есть очень неприятная особенность - любое повреждение базы данных в реестре с описанием поставщиков SPI приводит к сбоям в работе Winsock (конкретный вид сбоев зависит от характера повреждения, однако все ошибки для пользователя обычно сводятся к двум - компьютер перестает "видеть Интернет" и возникают проблемы при работе с локальной сетью - если проанализировать приведенную в начале данного раздела схему, то такие ошибки понятны и закономерны). Причин повреждения настроек SPI может быть очень много, наиболее характерные:

- Удаление DLL, являющейся провайдером SPI без отмены ее регистрации в базе провайдеров SPI (DLL может быть удалена вручную, антивирусным сканером, анти-шпионской программой, штатным деинсталлятором - способ удаления не важен);
- Повреждение базы в реестре в ходе инсталляции/деинсталляции SPI провайдеров, либо в результате ручной "чистки" реестра.

При обнаружении ошибки Winsock в частности в вся система Windows в целом не принимают никаких мер по ее ликвидации и не выдают никаких диагностических сообщений. Важно отметить, что переустановка системы "поверх" имеющейся не приводит к исправлению настройки SPI.

Как видно из приведенного мной описания, термин LSP встретился пару раз - я вообще заострил на нем внимание из-за утилиты LSP Fix и кочующего по описаниям троянских и SpyWare программ термина "LSP" - его применение во многих случаях не совсем корректно.

Причем тенденции "исправления" терминологии постепенно появляются - в последних описаниях вирусов они начинают давать более корректные и конкретные термины типа "...Registers a Windows sockets SPI hook in the LSP chain named...". Поэтому "лечить" и восстанавливать необходимо именно настройки SPI, а не LSP ...

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



XIII

13 Параметры командной строки

AVZ поддерживает параметры командной строки и профили (текстовые файлы, содержащие набор параметров). Профили могут применяться для хранения настроек и для обхода ограничений командной строки (в частности, могут возникнуть проблемы при использовании в командной строке большого количества параметров или применения имени папок с пробелами).

Основные параметры командной строки (в списке возможных значений параметра жирным шрифтом выделено значение по умолчанию).

Значения параметра описываются как
{}-обязательный строковый параметр,
[опция|опция] - одна из перечисленных опций

Все настройки, полученные через параметры командной строки, дублируются элементами интерфейса. Это позволяет производить настройки AVZ через командную строку без запуска сканирования. Наиболее удобно в данном плане применение профилей.

Любой параметр может быть настроен скриптом управления, причем в ходе работы скрипта допускается многократная перенастройка - подробности см. в описании команды [SetupAVZ](#)

13.1 Основные параметры

SCAN={каталог} Добавляет каталог или диск в список сканируемых. Путь должен начинаться с буквы диска, например c:\windows. В командной строке можно указать неограниченное количество параметров SCAN - обрабатываются все параметры, что позволяет задать сканирование нескольких каталогов. Указание несуществующего каталога не является ошибкой - он просто игнорируется. Повторное указание одного и того-же каталога или указание родительского каталога и его вложенных каталогов также не является ошибочным AVZ <суммирует> все указанные каталоги с учетом их вложенности. Пример:
SCAN=c:\windows

SCANDRIVE= [HDD|FDD|CDROM] Добавляет все диски заданного типа в список сканируемых. Поддерживается три типа дисков: HDD - отмечаются все жесткие диски, FDD-отмечаются все дисководы и Flash диски, CDROM- отмечаются все CD/DVD диски и диски, созданные эмуляторами CD дисков. Допустимо указание нескольких вариантов через запятую или знак + (без пробелов до и после запятой или знака + !), например,
SCANDRIVE=HDD+CDROM

SCANFILE={имя файла} Сканирование отдельного файла. Подобных параметров может быть несколько, сканирование файлов производится в порядке следования ключей. Указание несуществующего файла или повтор не является ошибкой. Пример:
SCANFILE=c:\windows\trojan.exe

NOSCAN={каталог} Исключение заданного каталога из списка проверяемых. Путь должен начинаться с буквы диска, например c:\windows. В командной строке можно указать неограниченное количество параметров NOSCAN - обрабатываются все параметры, что позволяет задать сканирование нескольких каталогов. Все параметры NOSCAN анализируются после построения списка сканируемых каталогов, основанного на параметрах SCAN. Указание несуществующего каталога не является ошибкой - он просто игнорируется. Повторное указание одного и того-же каталога или указание родительского каталога и его вложенных каталогов также не является ошибочным AVZ "суммирует" все указанные каталоги с учетом их вложенности.
Пример: **SCAN=c:\ NOSCAN=c:\aids** - будет проведено сканирование всего диска C:\ за исключением папки c:\aids

ScanAllFiles=[Y|N] Сканировать все файлы (независимо от расширения). Аналогично установке переключателя <Типы файлов> в положение <Все файлы>
ScanFilesMode=[0|1|2] Задаёт режим сканирования. 0 - сканирование потенциально-опасных файлов, 1 - сканирование всех файлов, 2 - сканирование файлов по маске. В режиме 2 необходимо задать маски ключами IncludeFiles и ExcludeFiles.

IncludeFiles={маска} Задаёт маску (или набор масок) файлов, которые необходимо проверять. Аналогично заполнению поля <Включая файлы по маске:> и включению одноименного переключателя

ExcludeFiles={маска} Исключить файлы с именами/расширениями по маске. Аналогично заполнению поля <Исключая файлы по маске:> и включению одноименного переключателя

WinTrustLevel=[0|1|2] Режим проверки файла по каталогу безопасности Microsoft (0-отключена, 1-проверка по каталогу, 2-проверка по каталогу плюс комплексная проверка ЭЦП самого файла). По умолчанию установлен режим 1, включение режима 2 повышает надёжность проверки, но увеличивает время проверки файла примерно в 2-3 раза.

ScanProcess=[Y|N] Проверять процессы и DLL, загруженные в память, по умолчанию включена

ScanSystem=[Y|N] Эвристическая проверка системы при помощи микропрограмм эвристики, по умолчанию включена

ScanSystemIPU=[Y|N] Эвристическая проверка системы при помощи микропрограмм ИПУ - искателя потенциальных уязвимостей, по умолчанию включена

RepGoodFiles = [Y|N] Включать в отчет информацию о файлах, которые по мнению AVZ являются "чистыми"

RepGoodCheck = [Y|N] Проверка "чистых" файлов по базе безопасных и базе ЭЦП Microsoft в выводе результатов проверки в протокол. Имеет смысл только при RepGoodFiles = Y

CheckArchives=[Y|N] Проверять архивы и составные файлы

UseInfected=[Y|N] Копировать удаляемые файлы в Infected

UseQuarantine=[Y|N] Копировать подозрительные файлы в карантин

EvLevel=[0|1|2|3] Уровень чувствительности эвристического анализатора (0-выключен, 3-максимальный уровень)

ExtEvCheck=[Y|N] Расширенная эвристическая проверка. Имеет смысл только при уровне эвристической 3.

Антируткит

RootKitDetect=[Y|N] Включение детектора RootKit и перехватчиков API

AntiRootKitSystem=[Y|N] Включение системы противодействия RootKit (на всех доступных уровнях). Включение противодействия RootKit автоматически включает их детектирование.

AntiRootKitSystemUser=[Y|N] Включение системы противодействия RootKit только UserMode.

AntiRootKitSystemKernel=[Y|N] Включение системы противодействия RootKit только KernelMode.

Параметры AntiRootKitSystemUser и AntiRootKitSystemKernel имеют приоритет над AntiRootKitSystem. Например, указание AntiRootKitSystem=Y AntiRootKitSystemKernel=N приведет к тому, что будет включена блокировка только руткитов UserMode (первый параметр включит все, а второй - отключит проверку KernelMode)

CheckLSP=[Y|N] Проверять настройки [SPI/LSP](#)⁴⁹. По умолчанию включено
AutoRepairLSP=[Y|N] Автоматически исправлять ошибки в настройке SPI/LSP. Работает только при условии, что CheckLSP=Y. По умолчанию отключено

KeyloggerSearch=[Y|N] Поиск клавиатурных шпионов и троянских DLL. По умолчанию включено

SearchTrojanPorts=[Y|N] Поиск портов, применяемых троянскими программами. По умолчанию включено

Profile={имя профиля} Загрузить профиль, хранящийся в указанном имени файла. Если не указан полный путь, то профиль ищется в текущей папке. Можно указать несколько параметров Profile (в этом случае загружаются все профили в порядке их упоминания в командной строке). Профиль в свою очередь также может содержать параметры Profile, что позволяет профилям ссылаться друг на друга. При загрузке проводится контроль - каждый профиль загружается только один раз, что позволяет корректно обрабатывать ситуации с перекрестными ссылками профилей

DelVir=[Y|N] Удаление/лечение найденных вирусов

ModeVirus=[0|1|2] Режим для вирусов (0-только отчет, 1- удалять или лечить, 2-спросить пользователя)

ModeAdvWare=[0|1|2] Режим для AdvWare (0-только отчет, 1- удалять, 2-спросить пользователя)

ModeSpy=[0|1|2] Режим для Spy и SpyWare (0-только отчет, 1- удалять, 2-спросить пользователя)

ModePornWare=[0|1|2] Режим для PornWare и Dialer (0-только отчет, 1- удалять, 2-спросить пользователя)

ModeRiskWare=[0|1|2] Режим для RiskWare (0-только отчет, 1- удалять, 2-спросить пользователя)

ExtFileDelete=[Y|N] Эвристическое удаление файлов. Подразумевает анализ системы, поиск и удаление ссылок на каждый стертый файл

Run=[Y|N] Автоматический запуск сканирования. Выполняется после выполнения всех остальных параметров

Обработка параметров без знака "="

Все параметры AVZ имеют вид *Имя=Значение*. Параметры, не содержащие знак равенства рассматриваются как имена файлов и папок, которые требуется просканировать. Для них выполняется две проверки:

1. Параметр рассматривается как имя каталога. Если каталог с таким именем существует, то он отмечается на проверку (в данном случае это эквивалентно ключу SCAN=<каталог>)
2. Если на шаге 1 каталог с указанным именем не найден, то значение параметра рассматривается как имя файла. Если файл с таким именем существует, то он отмечается на проверку (в данном случае это эквивалентно ключу SCANFILE={имя файла})

13.2 Специализированные ключи

Специализированные ключи не дублируются визуальными элементами настройки и предназначены в основном для системных администраторов.

Script=<имя скрипта> - загрузка и выполнение указанного скрипта. Данный ключ обрабатывается последним, независимо от его положения в командной строке

HiddenMode=[0|1|2|3] - режим запуска графической оболочки AVZ:

0 - Стандартный режим, окно видимо и доступно пользователю

1 - Окно AVZ невидимо, в трее отображается иконка. Пользователь может развернуть окно нажатием мышью на иконку. Пользователю доступно связанное с иконкой меню, позволяющее остановить и запустить сканирование.

2 - Окно AVZ невидимо, в трее отображается иконка, однако меню иконки заблокировано и пользователь не может развернуть окно AVZ

3 - Окно AVZ невидимо, иконка в трее не отображается.

Настройка режима работы полезна в случае запуска AVZ для проверки рабочих мест пользователей из logon-скрипта или при помощи автозапуска.

DetectMailBomb=[Y|N] - детектор "почтовых бомб" в системе распаковки архивов. Почтовой бомбой считается файл, размером более 10 Мб со степенью сжатия более 100.

Unpack_Archives=[Y|N] - распаковка проверяемых архивов и составных файлов типа MHT и почты в папку Unpacked в рабочей директории AVZ. Файлы группируются по типу архива и его имени.


Priority=[-1|0|1] - настройка приоритета процесса AVZ. -1 - пониженный приоритет, 0 - стандартный, 1 - повышенный. Пониженный приоритет полезно задавать в случае применения AVZ в качестве средства фоновой сканирования.

SleepScanTime=N, где N - время в миллисекундах. Задержка не заданное кол-во миллисекунд выдается после сканирования каждого файла, применяется для замедления процесса сканирования с целью минимизации нагрузки на диски и процессор, полезно для фоновой проверки папок на сервере. Например, если задать значение 100, то в результат в 1 секунду будет проверяться 10 файлов.

ScanAVZFolders=[Y|N] разрешает/запрещает сканировать папку AVZ и все уровнями глубже. По умолчанию параметр равен N, т.е. все файлы внутри рабочей папки AVZ не сканируются, что в частности защищает папку Infected от повторной проверки и лечения. Однако это не всегда удобно.

NQ=[Y|N] - сетевой режим карантина. В сетевом режиме карантина в имя папки карантина файла кроме даты включается сетевое имя ПК, это сделано для запуска AVZ из сетевой папки на сервере - в такой ситуации у каждого пользователя получается индивидуальный карантин и папка Infected, а администратор может легко ассоциировать попавшие в карантин файлы с компьютером.

WebServerMode=[Y|N] - режим запуска на WEB сервере для on-line проверки. Приводит к отключению сканирования памяти, расширенной проверки системы, поиска LSP ошибок, руткитов ... Назначение ключа - запуск AVZ в максимально облегченном режиме, что полезно при использовании AVZ в качестве средства on-line проверки файлов. Особенность в том, что по мере появления новых видов анализа памяти и системы они будут блокироваться данным ключом.

AG=[Y|N] - включение [AVZGuard](#)  в момент запуска AVZ. Этот ключ допустим только во командной строке, в скриптах он не поддерживается. Обработка ключа AG производится в момент запуска AVZ, до его инициализации и создания рабочих окон.

MiniLog=[Y|N] - включение режима сокращенного протоколирования. В этом режиме в протокол выводится только значимая информация (предупреждения, подозрения, данные о ошибках, текстовый вывод скриптов), вывод остальной информации подавляется (в частности, не выводится информация о загруженных базах, шапки с копирайтами, технические данные антивируса и т.п.). Режим полезен для администраторов, применяющих AVZ для проверки ПК в автоматическом режиме - сокращенный протокол гораздо меньше по размеру и его проще анализировать вручную или автоматически.

SpoolLog=<имя файла> - Дублирование протокола в указанный текстовый файл. Текстовый файл создается при его отсутствии и дополняется в случае наличия. Запись ведется без кеширования, открытие/закрытие файла производится в ходе записи каждой строки - это позволяет применять подобный протокол для диагностики ошибок в случае аварийного завершения AVZ.

QuarantineBaseFolder=<имя папки> - задает базовый каталог для папок Quarantine и Infected. По умолчанию данные папки расположены в рабочем каталоге AVZ. Данная опция полезна в случае запуска AVZ с BootCD или иного ReadOnly носителя, например из сетевой папки.

TempFolder=<имя папки> - Задает каталог для временных файлов. Если каталог с указанным именем не существует, то он автоматически создается. Данная опция позволяет переместить каталог с временными файлами AVZ в папку, проверка которой запрещена в настройках используемого антивирусного монитора. Кроме того, при помощи данного ключа можно переместить папку для временных файлов на виртуальный диск, что приведет к повышению скорости проверки архивов.

13.3 Коды возврата

Для взаимодействия с другими приложениями AVZ устанавливает коды возврата:

0 - в процессе сканирования не обнаружены подозрительные или вредоносные объекты

1 - обнаружены вредоносные объекты

2 - вредоносные объекты не обнаружены, но есть подозрения файлового сканера или эвристических анализаторов

13.4 Примеры

Пример конфигурации

Командная строка: avz.exe ScanDrive=HDD profile=exclude.txt DelVir=Y Run=Y

Файл exclude.txt:

NoScan=c:\VirusTest

NoScan=c:\Virus

NoScan=d:\aids

В данном примере при запуске AVZ автоматически отмечаются все HDD, производится исключение определенных папок (для параметров исключения применен профиль) включается режим лечения и после всех настроек производится запуск сканирования.

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



14 Часто задаваемые вопросы (FAQ)

В данном разделе документации я постараюсь собрать ответы на наиболее типовые и часто возникающие вопросы. Рекомендуется изучить FAQ перед обращением с вопросами по программе.

14.1 Что делать, если AVZ обнаружил подозрение на вирус или вредоносную программу ?

Утилита AVZ задумана как утилита, оснащенная массой различных проверок и анализаторов, порой параноидальных. Это сделано специально, т.к. AVZ часто применяется для анализа ПК, проверка которых другими средствами ничего не дала.

Поэтому ложные срабатывания возможны, и в этом случае в протоколе для объекта дается формулировка "Подозрение на ..." (на месте троеточия может выводиться категория вредоносной программы и уточняющие данные)

В случае обнаружения подозрительных объектов следует придерживаться следующей методике:

1. **Ни в коем случае не следует уничтожать подозрительные файлы.** То, что файл заподозрен анализатором, еще не означает, что он опасен. Необходимо поместить подозрительные файлы в карантин и выслать мне на адрес newvirus@z-oleg.com, (карантином пользоваться необязательно - можно вручную поместить файл в архив с паролем **virus**). При создании архива вручную очень желательно задать пароль, иначе письмо может быть заблокировано Вашим почтовым сервером;
2. В письме необходимо кратко изложить суть проблемы, какие есть подозрения. Очень желательно приложить протокол AVZ
3. Дождаться ответа с результатами анализа.

Подробнее о обращении в техническую поддержку можно почитать в разделе "[Техническая поддержка](#)"^[17]

14.2 Что делать, если AVZ выдал подозрение на keylogger ?

При обнаружении подозрений на Keylogger или троянскую DLL следует действовать по схеме, описанной в совете "[Что делать, если AVZ обнаружил подозрение на вирус ?](#)"^[99]. Кроме того, стоит заметить, что подозрение на клавиатурный перехватчик совершенно не означает, что обнаружен клавиатурный шпион. Известны сотни программ, которые устанавливают свои перехватчики клавиатуры и мыши для решения вполне мирных задач. Наиболее типичным примером является словарь Lingvo, который устанавливает свой перехватчик LvHook.dll для отслеживания нажатий на "горячие клавиши" - с одной стороны это перехватчик и анализатор AVZ на него реагирует, с другой - он совершенно безопасен. После анализа я вношу безопасные перехватчики в базу безопасных объектов и срабатывания на них прекращаются. По текущей статистике соотношение "опасный"/"безопасный" находится примерно на уровне 1/10, т.е. примерно на 10 безопасных модулей обнаруживается один вредоносный.

14.3 Что делать, если AVZ выдал подозрение на RootKit ?

В случае, если выдано подозрение на некоторый процесс или файл, то следует действовать по общей схеме ("[Что делать, если AVZ обнаружил подозрение на вирус ?](#)"^[99]), но перед отправкой файлов желательно прислать протокол работы AVZ - часто по протоколу можно многое объяснить.

Следует понимать, что обнаружение перехвата тех или иных функций еще не означает, что в систему внедрился RootKit - возможно, перехват функций применяется антивирусным монитором, Firewall или иными утилитами. Например, весьма полезная и удобная утилита RegMon устанавливает свой драйвер для перехвата 13-ти функций KiST для слежения за

операциями с реестром. По моей статистике примерно в одном случае из 50-80 перехват произведен вредоносной программой, в остальных случаях как правило это антивирусы и Firewall.

В случае обнаружения подозрения на маскировку процесса однозначно следует разобраться, что это за процесс (программа). Обычные программы (не говоря уже о компонентах системы) не прибегают к маскировке своих процессов и файлов, поэтому маскирующийся процесс однозначно подозрителен. Однако однозначно судить о вредоносности нельзя - например, известен ряд безопасных программ, применяющих маскировку процесса и файлов для их защиты от обнаружения и удаления троянскими программами.

14.4 Что такое 'Порт TCP 5000' и как с ним бороться ?

AVZ часто находит открытый порт TCP 5000 и сообщает об этом. Этот порт принадлежит системной службе, называемой "Служба обнаружения SSDP". Если данная служба не применяется, то рекомендуется ее отключить (в данной службе в свое время был обнаружен ряд уязвимостей).

Для отключения необходимо:

1. Нажать правую клавишу на ярлыке "Мой компьютер", в появившемся меню вызвать пункт "Управление"
2. В открывшейся консоли "Управление компьютером" раскрыть "Службы и приложения", там - "Службы"
3. В списке, который появится в правой стороне окна, следует найти службу с именем "Служба обнаружения SSDP"
4. Двойной клик левой клавишей по строке данной службы откроет окно свойств, там необходимо выбрать тип запуска "Отключено", затем нажать кнопку "Стоп", затем - "ОК"

После отключения службы обнаружения SSDP порт 5000 не должен прослушиваться - если прослушивание продолжается, то есть подозрение, что это Backdoor.

14.5 Что делать, если после распаковки AVZ не может найти файл *.avz ?

Если после распаковки AVZ сообщает о том, что не найден некий файл с расширением AVZ (чаще всего это файл main.avz), то необходимо проверить правильность распаковки. Дело в том, что базы утилиты AVZ хранятся в папке Base. Некоторые архиваторы некорректно распаковывают размещенный на сайте архив, в результате папка Base не извлекается из архива.

Рекомендации - распаковать архив другим архиватором и убедиться, что в результате распаковки извлечена папка Base. Подробнее по структуре папок можно почитать в разделе [Структура папок программы](#) ^[17]

14.6 AVZ удалил вредоносные программы, но стартовая страница и прочие настройки браузера по-прежнему повреждены

Это типовая ситуация - AVZ не может точно знать, какие именно настройки повредила вредоносная программа. Для восстановления настроек необходимо воспользоваться опцией "[Восстановлением системы](#)" ^[59]. Если после восстановления настройки браузера по-прежнему самопроизвольно меняются, то это первый признак того, что на компьютере есть вредоносные программы. В таком случае стоит провести [Исследование системы](#) ^[57] и проанализировать полученный протокол.

14.7 Во время сканирования диска мой антивирусный монитор сообщил, что файл avz*.tmp заражен вирусом или является вредоносной программой

Подобные сообщения могут выдаваться антивирусными мониторами, при этом зараженный файл обнаруживается в папке TEMP и имеет имя вида avz_XXXX_Y.tmp. Это временные файлы, которые создаются AVZ в ходе проверки архивов, писем электронной почты, MHT файлов и т.п., причем XXXX - это PID процесса AVZ, а Y - уровень вложенности архива. Есть другой вариант имени файла - avz_XXXX_raw.tmp - подобные файлы создаются в ходе анализа объектов, доступ к которым блокирован путем их монопольного открытия.

Причина срабатывания заключается в том, что применяемый антивирусный сканер и монитор почему-то не находят вирусы/трояны в каком-либо архиве. Можно выделить несколько типовых причин:

1. В настройках антивируса и антивирусного монитора отключена проверка архивов
2. Антивирус не умеет проверять данный тип архива
3. Архив находится в папке, проверка которой запрещена в настройке антивируса и его монитора
4. Когда архив попал на компьютер, в базе антивируса не было сигнатур содержащихся там вредоносных программ - поэтому монитор пропустил данный архив и позволил его сохранение.

Соответственно в момент извлечения AVZ-ом файла из архива антивирусный монитор проверяет его и может обнаружить вредоносную программу. В этом случае срабатывание вполне объяснимо и дальнейшая реакция может быть любой, в частности

1. Можно разрешить антивирусному монитору "вылечить" файл,
2. Можно дать монитору команду проверку файла

В любом случае AVZ рассчитан на возможность неожиданного удаления временного файла или блокировки доступа к нему со стороны антивирусного монитора, поэтому сбоя в его работе не последует.

14.8 Что делать, если выдается сообщение "Ошибка загрузки драйвера - проверка прервана"

Сообщение "Ошибка загрузки драйвера - проверка прервана" может выводиться в протокол в случае ошибки загрузки драйвера avz. Причин возникновения ошибки несколько, наиболее распространенные:

1. Нехватка прав. Для загрузки драйвера необходимо запускать AVZ из под учетной записи администратора или пользователя, имеющего право устанавливать и загружать драйвера
2. Наличие запущенных систем проактивной защиты, для которых AVZ не является доверенным процессом. Эти системы могут блокировать загрузки и (или) установку драйвера
3. Блокировка загрузки драйвера со стороны вредоносных программ

14.9 Что делать, если в ходе обновления баз возникает ошибка

Возможно несколько вариантов решения проблемы:

1. Меню "Файл/Стандартные скрипты" - выполнить скрипт "Обновление баз с автоматической настройкой". Данный скрипт пробует различные типовые комбинации настроек - в случае обнаружения подходящей конфигурации скрипт обновляет базу.
2. Можно попробовать вручную выбрать другой источник обновления и (или) изменить настройки, например попробовать прямое соединение или соединение через прокси-сервер (в случае, если применяется прокси-сервер и известен его адрес и порт).

Кроме того, необходимо убедиться, что:

1. Настройки Firewall позволяют AVZ обращаться в Интернет. Загрузка баз ведется с указанных сайтов в настройке сайтов, порт 80, протокол - HTTP.

2. Имеется устойчивое соединение с Интернет.

Если обновить базы в автоматическом режиме не удастся, то можно загрузить архив с базам вручную на странице загрузки AVZ. При этом объем загружаемой информации будет больше ввиду того, что автоматическое обновление загружает только обновленные или поврежденные файлы.

14.10 Что делать, если в ходе загрузки архива с AVZ возникают ошибки

В данном случае можно выделить две типовых ситуации:

1. Архив с AVZ не загружается. В данном случае точно проблему установить сложно, загрузка архива ведется по прямой ссылке, по протоколу HTTP, редиректы или системы типа "анти-ссылка" у меня на сайте не применяются. Возможно, проблемы с загрузкой связаны с высокой нагрузкой на сервер - после выхода новых версий AVZ в течении как минимум недели возникает ситуация DDoS из-за большого количества параллельных загрузок. В этом случае рекомендуется использовать указанные на странице загрузки AVZ зеркала.

2. Архив загружается, но его распаковка показывает, что там старая версия. В данном случае проблема связана с некорректной работой применяемого для доступа в Интернет прокси сервера, который кеширует старую версию архива. В теории прокси-сервер должен проверить, обновился ли файла - и в случае обновления загрузить актуальную версию файла.

14.11 В ходе проверки AVZ в активное окно вводится текстовая строка с текстом "test"

В ходе проверки компьютера AVZ в активное окно может быть введена длинная строка (типовое содержимое - повторяющееся слово test). Причина - работа антикейлоггера AVZ, который использует в своей работе поведенческий анализатор. Для работы поведенческий анализатор эмулирует события типа "клавиатура" и "мышь", и изучает реакцию на них со стороны подозреваемых библиотек. Если активным является окно AVZ, то все происходит корректно, в противном случае возникает описанная выше ситуация.

14.12 Под Vista не работают некоторые функции AVZ

Под Windows Vista 64-bit не функционируют KernelMode компоненты AVZ - это связано с тем, что AVZ не поддерживает 64-bit платформы. Под Vista 32-bit все должно функционировать, но следует учесть, что под Vista приложения по умолчанию запускаются с ограниченными привилегиями. Для запуска приложения с правами администратора необходимо выделить avz.exe в проводнике, вызвано контекстное меню (правая кнопка мыши) и выбрать пункт "Запустить приложение с правами администратора". После этого будет выдан запрос системы UAC с просьбой подтвердить операцию.

14.13 Особенности применения AVZ на Windows Server 2003

В случае запуска AVZ на сервере следует соблюдать особую осторожность и включать лечение / автоисправление только тогда, когда есть полная уверенность в надобности подобного исправления !

Типовые проблемы могут быть связаны с рядом моментов:

1. В случае запуска AVZ из терминальной сессии система передает ему некорректный путь к системной папке - возвращаемый системой путь указывает в профиль пользователя. Это может привести к сбоям в ряде менеджеров, в частности в анализаторе ошибок SPI

2. На сервере могут периодически запускаться и завершаться фоновые процессы. Это может приводить к сбоям в работе антивируса, в частности он может подозревать маскировку

некоторых процессов

3. Нейтрализация перехватов в Kernel Mode, модификация настроек SPI, работа с автозапуском и т.п. оказывает глобальное воздействие на систему, поэтому необходимо выполнять данные операции с особой осторожностью

14.14 Что делать, если вредоносная программа блокирует запуск AVZ ?

В этой ситуации следует:

1. Переименовать папку AVZ, задать ей бессмысленное имя типа X54S или распространенное типа Soft, Game и т.п.
2. Переименовать avz.exe в что-то типа test.exe, game.pif, program.com
3. Запускать AVZ с ключом: avz.exe ag=y (в этом случае при запуске AVZ включается AVZGuard, подробности о ключе см. в разделе [Специализированные ключи](#) ⁽⁹⁶⁾)

Top Level Intro

This page is printed before a new
top-level chapter starts

Part



XV

15 Скрипты управления

15.1 Введение

Как известно, AVZ исходно задумывался как утилита для сисадмина. При этом проверка конкретного компьютера удобна в случае запуска AVZ вручную и работы в диалоговом режиме. Однако для оперативной проверки большого количества ПК или для выполнения их периодической проверки в ходе загрузки необходима возможность автоматизации. Первым шагом в этом направлении была поддержка ключей командной строки. Однако ключи позволяют настроить AVZ, но не дают особой гибкости в работе с ним. Поэтому начиная с версии 3.80 введена поддержка внешних скриптов управления, которые могут создаваться администратором.

Скрипт позволяет:

- Модифицировать все настройки AVZ
- Запускать проверку и лечение заданных папок и дисков
- Сохранять протоколы (причем в протокол могут вноситься поясняющие данные)
- Выполнять исследование системы
- Помещать в карантин файлы (указанных файлов или автокарантин)
- Производить поиск классов с возможностью карантина связанных с ними файлов
- Выполнять блокирование перехватчиков UserMode и KernelMode

Главной особенностью скрипта является возможность использования в ней условных операторов и циклов, объявлять переменные различных типов, проводить операции с числами и строками. Все это позволяет решать ряд сложных задач, реализация которых невозможна с помощью ключей командной строки - например, проводить избирательную настройку AVZ для каждого проверяемого ПК, проводить лечение определенных папок и т.п.

15.2 Структура скрипта

В простейшем виде скрипт управления имеет вид:

```
begin  
... команды скрипта ...  
end.
```

При необходимости объявления переменных добавляется секция объявления переменных:

```
var  
S : string; // Объявление переменной типа "строка"  
begin  
... команды скрипта ...  
end.
```

Как видно из примеров, синтаксис идентичен языку Pascal. Скрипт может содержать комментарии, которые не влияют на его выполнение и предназначены для временного отключения фрагментов скрипта или для комментирования описанных в нем операций.

```
begin  
// Это первый вид комментария  
{ А это - второй вид комментария }  
end.
```

В скрипте поддерживается несколько типов переменных, в частности string (строка), integer (

целое число со знаком), double (число с плавающей точкой).

15.3 procedure Sleep

```
procedure Sleep(AInterval : integer);
```

Выполняет приостановку скрипта (и всего AVZ в целом) на указанный интервал времени. Интервал задается в секундах.

Данная команда полезна в начале скрипта в случае запуска AVZ из автозапуска - задержка на 60-80 секунд полезна, чтобы не замедлять процесс загрузки системы.

Пример:

```
AddToLog('Выполняем задержку на 15 секунд ');  
Sleep(15);  
AddToLog('15 секунд прошли, продолжаем работу');
```

15.4 procedure ActivateWatchDog

```
procedure ActivateWatchDog(Interval : integer);
```

Активирует сторожевой таймер. Параметр Interval содержит интервал времени в секундах. Сторожевой таймер позволяет ограничить время работы AVZ и принудительно завершить его работу в случае непредвиденной остановки.

Пример:

```
ActivateWatchDog(60 * 5); // Принудительное завершение работы  
AVZ через 5 минут
```

15.5 procedure SetupAVZ

```
procedure SetupAVZ(S : String);
```

Производит настройку AVZ. В качестве параметра указываются любой параметр командной строки (только один параметр).

В пределах скрипта допускается многократный вызов SetupAVZ, каждый вызов модифицирует один из параметров. Параметры подробно описаны в разделе "[Параметры командной строки](#)"⁹³

Пример 1:

```
SetupAVZ('UseQuarantine=Y'); // Включение карантина
```

Пример 2:

```
if GetComputerName = 'PUPKIN' then  
    SetupAVZ('UseQuarantine=Y'); // Включение карантина
```


15.6 procedure RunScan

```
procedure RunScan;
```

Запуск сканирования. Выполнение скрипта продолжится после завершения сканирования. Настройки сканирования должны быть заданы до вызова RunScan.

15.7 procedure ExitAVZ

```
procedure ExitAVZ;
```

Завершение работы AVZ. Применяется в качестве последней команды в скриптах, рассчитанных на автоматическое выполнение

15.8 function GetSystemBootMode

```
function GetSystemBootMode : integer;
```

Возвращает код, соответствующий режиму, в котором загружена Windows.

Поддерживаются следующие коды:

0 - загрузка системы проведена в обычном режиме

1 - загрузка в режиме защиты от сбоев

2 - загрузка в режиме защиты от сбоев с поддержкой сети

Примеры:

```
begin
```

```
  AddToLog('Система загружена в режиме ' +  
          IntToStr(GetSystemBootMode));
```

```
end.
```

```
begin
```

```
  if GetSystemBootMode <> 0 then begin  
    AddToLog('Проверка в режиме защиты от сбоев заблокирована  
    !!')
```

```
    exit;
```

```
  end;
```

```
end.
```

15.9 function GetEnvironmentVariable

```
function GetEnvironmentVariable(AName : string):string;
```

Запрашивает значение указанной переменной окружения.

Совместимость: AVZ 4.26 и выше

Пример:

```
begin
```

```
  AddToLog('SESSIONNAME = ' + GetEnvironmentVariable('SESSIONNAME'));
```

end.

15.10 Информация о компьютере

15.10.1 function GetComputerName

function GetComputerName : **string**;

Возвращает сетевое имя компьютера. Сетевое имя всегда возвращается в верхнем регистре для упрощения сравнения и анализа.

Примеры применения:

1. AddToLog('Протокол с компьютера '+GetComputerName);
В данном примере в протокол вносится строка, содержащая текущее имя компьютера

2. **if** GetComputerName = 'PUPKIN' **then begin**
 SetupAVZ('DelVir=Y'); // Включение лечения
 SetupAVZ('UseQuarantine=Y'); // Использование
автокарантина
end;

В данном примере для компьютера с именем PUPKIN проводятся некоторые индивидуальные операции (в данном случае включается лечение и использование карантина)

3. **if** pos('ASU_', GetComputerName) = 1 **then**
 ExitAVZ;

В данном примере для компьютеров, имя которых начинается с ASU_ производится немедленный выход из AVZ

15.10.2 function GetComputerComments

function GetComputerComments : **string**;

Возвращает текстовое описание компьютера, которое указывается в поле "Описание" в настройках имени компьютера. Данная функция полезна в корпоративной сети при условии заполнения описания компьютеров данными о его местоположении и ответственном за данный компьютер пользователе.

Пример:

begin
 AddToLog('Описание компьютера = '+GetComputerComments);
end.

15.10.3 function IsNT

function IsNT : **boolean**;

Возвращает true, если установленная на компьютере версия Windows относится к платформе NT (NT, W2K, XP, W2K3, Vista) и false для платформы W9x.

Совместимость: AVZ 4.26 и выше

15.11 Взаимодействие с пользователем

15.11.1 procedure ShowMessage

```
procedure ShowMessage (AMsg : string);
```

Процедура ShowMessage выводит текстовое сообщение AMsg на экран. Выполнение скрипта останавливается до тех пор, пока пользователь не нажмет кнопку "ОК" в окне с сообщением.

15.11.2 function InputBox

```
function InputBox(ACaption, APrompt, ADefault: string): string  
;
```

Функция InputBox выводит на экран диалоговое окно с полем ввода. Заголовок окна определяется параметром ACaption, приглашение для ввода данных - APrompt. Значение ADefault вводится в строку в качестве значения по умолчанию. В окне имеется две кнопки - ОК и Отмена. Работа скрипта останавливается до нажатия на кнопку "ОК" или "Отмена" в окне ввода. При нажатии ОК функция возвращает значение, введенное пользователем. При нажатии кнопки "Отмена" возвращается значение по умолчанию, заданное параметром ADefault.

```
var  
  txt : string;  
begin  
  txt := InputBox('Заголовок окна', 'Введите строку:', 'Строка  
по умолчанию');  
  ShowMessage('Введен текст="'+txt+'");  
end.
```

15.11.3 function MessageDlg

```
function MessageDlg(Msg: string; DlgType: TMsgDlgType; Buttons: TMsgDlgType);
```

Данная функция отображает на экране стандартное диалоговое окно и возвращает код кнопки, нажатой пользователем в диалоговом окне. Применение данной функции позволяет создавать интерактивные скрипты.

Параметры:

Msg - текст сообщения или вопроса, выводимого в диалоговом окне

DlgType - тип диалогового окна. Для задания типа диалогового окна предусмотрен набор констант:

- mtWarning - предупреждение
- mtError - ошибка
- mtInformation - информация
- mtConfirmation - запрос подтверждения

С технической точки зрения тип диалогового окна определяет заголовок окна и отображаемую в окне иконку.

Buttons - код, определяющий, какой набор кнопок необходимо отобразить в диалоговом окне.

Для задания набора кнопок предусмотрен набор констант:

- mbOk - "ОК"
- mbCancel - "Отмена"
- mbYes - "Да"
- mbNo - "Нет"
- mbAbort - "Прервать",
- mbRetry - "Повторить",

- mblgnore - "Игнорировать"
 - Если требуется отобразить несколько кнопок, то код набора кнопок получается суммированием констант (например, mbYes+mbNo).
- HelpCtx - код раздела справочной системы, в пользовательских скриптах должен быть равен нулю

Функция возвращает код нажатой кнопки:

- mbOk - код 1
- mbCancel - код 2
- mbYes - код 6
- mbNo - код 7
- mbAbort - код 3,
- mbRetry - код 4
- mblgnore - код 5

Примеры:

```
var
  Res : integer;
begin
  Res := MessageDLG('Выполнить операцию ?', mtConfirmation, mbYes+mbNo,
    AddToLog('Код результата = ' + Inttostr(Res)));
end.
```

Пример анализа кода:

```
var
  FileName : string;
begin
  FileName := 'c:\trojan.exe';
  if MessageDLG('Удалить файл "' + FileName + '" ?', mtConfirmation, mbYes+mbNo) then
    DeleteFile(FileName);
end.
```

15.12 Автоматическое обновление

15.12.1 function ExecuteAVUpdate

```
function ExecuteAVUpdate : boolean;
```

Выполнение обновления баз утилиты AVZ через Интернет. Параметры для функции не требуются, обновление производится с одного из серверов, содержащих обновление. Сервер выбирается случайным образом

Пример:

```
begin
  if ExecuteAVUpdate then
    AddToLog('Обновление AV баз успешно выполнено');
end.
```

15.12.2 function ExecuteAVUpdateEx

```
function ExecuteAVUpdateEx(AServerURL : string; AConnectMode  
: byte; ProxyServer, ProxyUser, ProxyPass : string) : boolean;
```

Выполнение обновления баз утилиты AVZ через Интернет. Аналогична ExecuteAVUpdate, но проводит обновление согласно заданным параметрам с указанного URL, что в частности позволяет обновлять базы с сервера в локальной сети.

Параметры:

AServerURL - URL сервера, с которого проводится обновление. Если вместо URL указать пустую строку, то в качестве источника обновления случайным образом берется один из стандартных URL (случайный выбор позволяет распределять нагрузку между источниками обновления).

AConnectMode - код режима соединения:

0 - соединение в соответствии с настройками Internet Explorer. В этом режиме параметры ProxyServer, ProxyUser, ProxyPass игнорируются;

1 - прямое соединение. Оптимально для компьютера, имеющего прямой выход в Интернет. В этом режиме параметры ProxyServer, ProxyUser, ProxyPass игнорируются;

2 - соединение через прокси сервер, не требующий авторизации. В этом режиме в обязательном порядке должен быть указан ProxyServer, через который будет вестись работа, параметры ProxyUser и ProxyPass игнорируются;

3 - соединение через прокси сервер, требующий авторизации. Режим аналогичен режиму 2, но обязательно заполняются параметры ProxyUser, ProxyPass.

4 - соединение через прокси сервер, требующий авторизации по протоколу NTLM. Режим аналогичен режиму 3.

ProxyServer - имя или IP адрес прокси-сервера, имеет смысл только в режимах 2 и 3.

ProxyUser - имя пользователя для авторизации на прокси сервере, имеет смысл только в режиме 3

ProxyPass - пароль пользователя для авторизации на прокси сервере, имеет смысл только в режиме 3

Примеры:

```
begin  
  if ExecuteAVUpdateEx('http:\\my_server.com\\avz_av_update\\',  
0, '', '', '') then  
    AddToLog('Обновление AV баз (по настройкам IE) успешно  
выполнено');  
end.
```

```
begin  
  if ExecuteAVUpdateEx('http:\\my_server.com\\avz_av_update\\',  
1, '', '', '') then  
    AddToLog('Обновление AV баз (прямое соединение) успешно  
выполнено');  
end.
```

```

begin
  if ExecuteAVUpdateEx('http:\\my_server.com\\avz_av_update\\',
2, 'myproxy','','') then
    AddToLog('Обновление AV баз (Прoxy сервер без авторизации)
успешно выполнено');
  end.

begin
  if ExecuteAVUpdateEx('http:\\my_server.com\\avz_av_update\\',
3, 'myproxy','Pupkin','TopSecretPasswd') then
    AddToLog('Обновление AV баз (Прoxy сервер с авторизацией)
успешно выполнено');
  end.

```

15.13 Работа с протоколом

15.13.1 procedure AddToLog

```
procedure AddToLog(S : String);
```

Выполняет добавление в протокол указанной строки. Данная функция полезна для регистрации некоторой информации, которую разработчик скрипта желает внести в протокол.

Пример:

```
AddToLog('Протокол с компьютера '+GetComputerName);
```

15.13.2 procedure SaveLog

```
procedure SaveLog(S : string);
```

Сохранение протокола в файл. Файл может размещаться в сети. Если указанный в имени файла протокола путь не существует, то перед сохранением файла протокола делается попытка создания этой папки. Если по указанному пути уже существует файл с указанным именем, то он затирается сохраняемым протоколом без вывода запросов или предупреждений.

Примеры:

```
SaveLog('c:\\avz_log.txt');
SaveLog('\\\\my_server\\avz_logs\\avz_'+GetComputerName+'.txt');
```

15.13.3 procedure SaveCSVLog

```
procedure SaveCSVLog(S : string);
```

Сохраняет протокол в формате CSV. Протокол в формате CSV содержит три поля, разделенных символом ";" - имя файла, код события и текстовое примечание.

Поддерживаются следующие коды событий:

- 1 - вредоносная программа
- 2 - подозрение файлового сканера на вредоносную программу
- 3 - подозрение эвристика
- 4 - подозрение антируткита

5 - подозрение антикейлоггера

Пример:

```
SaveCSVLog('c:\avz_log.csv');
```

15.13.4 function AddLineToTxtFile

```
function AddLineToTxtFile(AFileName, S : string) : boolean;
```

Добавляет строку S к текстовому файлу с именем AFileName. Файл создается в случае его отсутствия и открывается в случае наличия, закрытие файла производится сразу после добавления строки. Данная функция удобна для формирования сводных протоколов и ведения единого протокола с данными для администратора.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

Функция возвращает true в случае успешного добавления строки к файлу и false в случае ошибки.

Пример:

1. Простейший пример:

```
begin
```

```
    AddLineToTxtFile('c:\log.txt', 'Сообщение');
```

```
end.
```

2. Пример генерации протокола, строки которого содержат дату, время и имя компьютера:

```
begin
```

```
    AddLineToTxtFile('c:\log.txt',  
                    DateTimeToStr(Now) + ' ' + GetComputerName + ' :
```

```
    '+'
```

```
        'Сообщение');
```

```
end.
```

15.14 Карантин и папка Infected

15.14.1 procedure ExecuteAutoQuarantine

```
procedure ExecuteAutoQuarantine;
```

Выполнение автоматического карантина файлов, которые не опознаны по базе безопасных. В такой карантин попадают все загруженные процессы/DLL/драйверы, элементы автозапуска.

Пример:

```
begin
```

```
    ClearQuarantine;
```

```
end.
```

15.14.2 function CreateQurantineArchive

```
function CreateQurantineArchive(AFileName : string) :  
boolean;
```


Функция создает архив с файлами, помещенными сегодня в папку карантина (т.е. берутся все файлы из папки Infected, соответствующей текущей дате). Архив имеет ZIP формат, пароль - virus. Если указанная в параметре AFileName папка не существует, то перед сохранением архива делается попытка создать ее.

Пример:

```
begin
    // Очистка карантина
    ClearQuarantine;
    // Автокарантин
    ExecuteAutoQuarantine;
    // Создание архива с файлами, помещенными в карантин
    CreateQuarantineArchive (GetAVZDirectory+'quarantine.zip');
end.
```

15.14.3 function CreateInfectedArchive

```
function CreateInfectedArchive (AFileName : string) : boolean;
```

Функция создает архив с файлами, помещенными сегодня в Infected (т.е. берутся все файлы из папки карантина, соответствующей текущей дате). Архив имеет ZIP формат, пароль - virus. Если указанная в параметре AFileName папка не существует, то перед сохранением архива делается попытка создать ее.

Пример:

```
begin
    // Создание архива с содержимым папки Infected
    CreateInfectedArchive (GetAVZDirectory+'infected.zip');
    // Автокарантин
    ExecuteAutoQuarantine;
end.
```

15.14.4 function QuarantineFile

```
function QuarantineFile (AFileName, AMsg : string):boolean;
```

Копирование указанного файла с именем AFileName в карантин. При помещении в карантин в описание файла вносится текст из параметра AMsg. Если файл уже есть в карантине или он опознан по базе безопасных, то копирование в карантин блокируется.

Если файл найден и успешно скопирован в карантин, то функция возвращает true.

Если файл не найден или помещение в карантин заблокировано по причине наличия файла в базе безопасных, то возвращается False.

В имени файла допустимы [макросы](#)^[150], в частности:

%WinDir% - полный путь в папке Windows

%System32% - полный путь к папке System (в Win9x) или System32 (в NT/W2K/XP ...)

На заметку:

Если функции QuarantineFile не указано полное имя файла, то делается попытка поиска файла, в ходе которого кроме штатного поиска перебираются типовые варианты местоположения файла, в частности проверяются варианты "имя файла + пути к системным папкам". Это

делается для того, чтобы повысить вероятность карантина нужных файлов для исследования

Пример:

```
QuarantineFile('%System32%\vbsys2.dll',  
               'Подозрительное имя, возможно Trojan.Agent');  
QuarantineFile('%WinDir%\vbsys2.dll',  
               'Подозрительное имя, возможно Trojan.Agent');
```

В имени файла допускаются маски, содержащие "*" и "?", например:

```
QuarantineFile('%System32%\vbsys*.dll',  
               'Подозрительное имя, возможно Trojan.Agent');
```

При указании маски производится поиск подходящих файлов, и все найденные файлы добавляются в карантин.

В случае необходимости папка Quarantine может быть перемещена в любой каталог с помощью ключа командной строки [QuarantineBaseFolder](#)^[96]. В этом случае настройка папки карантина должна производиться до операций с карантинном.

Пример:

begin

// Настройка местоположения папки карантина

```
SetupAVZ('QuarantineBaseFolder=c:\avz4\');
```

// Карантин файла

```
QuarantineFile('%System32%\vbsys2.dll',  
               'Подозрительное имя, возможно Trojan.Agent');
```

end.

15.14.5 function ClearQuarantine

```
function ClearQuarantine:boolean;
```

Выполняет очистку карантина. Чистится только папка карантина за текущую дату.

Пример:

begin

// Очистка карантина

```
ClearQuarantine;
```

end.

15.15 Функции запроса информации о AVZ

15.15.1 function GetAVZDirectory

```
function GetAVZDirectory : string;
```

Возвращает полный путь (с завершающим '\' на конце) к папке, из которой запущен AVZ.

Пример:

```
begin
  // Создание архива с файлами, помещенными в карантин
  CreateQuarantineArchive(GetAVZDirectory+'quarantine.zip');
end.
```

В данном примере файл с помещенными в карантин файлами создается в папке AVZ.

15.15.2 function GetScanPath

```
function GetScanPath:string
```

Возвращает список сканируемых папок. Если задано сканирование нескольких папок, то они разделяются точкой с запятой.

Пример:

```
var
  s : string;
begin
  setupavz('scan=c:\Program Files');
  setupavz('scan=c:\windows');
  addtolog('Сканируются: ' + GetScanPath);
  // Выделение первого сканируемого каталога
  s := GetScanPath;
  if pos(';', s) > 0 then
    s := copy(s, 1, pos(';', s) - 1);
  addtolog('Первая сканируемая папка = ' + s);
end.
```

15.15.3 function GetAVZVersion

```
function GetAVZVersion : double;
```

Возвращает версию AVZ в виде числа с плавающей запятой. Следует помнить, что условия "=" к переменным с плавающей запятой применять не рекомендуется.

Пример:

```
begin
  if GetAVZVersion < 4.20 then begin
    AddToLog('Для работы скрипта необходима версия AVZ не менее 4.20');
    AddToLog('Текущая версия - '+FormatFloat('#0.00', GetAVZVersion));
    exit;
  end;
end.
```

15.15.4 function GetAVZVersionTxt

```
function GetAVZVersionTxt : string;
```

Получение версии AVZ в текстовом виде. Возвращается строка, выводимая в копирайты протоколов (она содержит версию и дату ее выпуска).

15.15.5 function GetAVZSvcName

```
function GetAVZSvcName(ID : integer) : string;
```

Возвращает имя службы AVZ с указанным идентификатором.

Совместимость: AVZ 4.26 и выше

15.16 Интерфейс AVZ

15.16.1 procedure UnLockInterface

```
procedure UnLockInterface;
```

Разблокирует интерфейс программы AVZ для пользователя. Вызов для незаблокированного состояния интерфейса и повторные вызовы не являются ошибкой

15.16.2 procedure LockInterface

```
procedure LockInterface;
```

Заблокировать интерфейс программы AVZ для пользователя. Повторные вызовы не являются ошибкой.

15.16.3 procedure SetStatusBarText

```
procedure SetStatusBarText (AMsg : string);
```

Выводит заданный текст в строке статуса. Может применяться для вывода информации о ходе работы скрипта, особенно в случае его длительного выполнения.

15.17 Управление AVZPM

15.17.1 function SetAVZPMStatus

```
function SetAVZPMStatus(ANewStatus : boolean) : boolean;
```

Включает или выключает AVZPM. ANewStatus - новый статус системы AVZPM. Если ANewStatus = true, то система AVZPM активируется, если false - выключается. Попытка повторного включения или повторного выключения не является ошибкой. Активация предполагает установку драйвера на диск, регистрацию драйвера в реестре и загрузку драйвера. Удаление предполагает удаление ключа реестра и файла драйвера с диска. Возвращаемое значение: возвращается признак успешности выполнения операции, true - операция выполнена успешно, false - ошибка.

Пример:

```
if SetAVZPMStatus(True) then
  AddToLog('AVZ PM успешно установлен и активирован')
else
  AddToLog('AVZ PM - ошибка активации');
```

На заметку:

Помните, что драйвер AVZ PM автоматически загружается при старте системы, поэтому после активации AVZ PM и перезагрузки драйвер загрузится автоматически.

15.17.2 function GetAVZPMStatus

```
function GetAVZPMStatus : boolean;
```

Возвращает текущий статус AVZPM - true=включен, false=выключен.

Пример:

```
if GetAVZPMStatus then
  AddToLog('AVZ PM активен')
else
  AddToLog('AVZ PM не активен');
```

15.18 Управление AVZGuard**15.18.1 function SetAVZGuardStatus**

```
function SetAVZGuardStatus(ANewStatus : boolean) : boolean;
```

Включает или выключает AVZGuard. ANewStatus - новый статус системы AVZGuard. Если ANewStatus = true, то система AVZGuard активируется, если false - выключается. Попытка повторного включения или повторного выключения не является ошибкой. Возвращаемое значение: возвращается признак успешности выполнения операции, true - операция выполнена успешно, false - ошибка.

Пример:

```
if SetAVZGuardStatus(True) then
  AddToLog('AVZGuard успешно активирован')
else
  AddToLog('AVZGuard - ошибка активации');
```

На заметку:

Помните, что если завершить работу AVZ при включенном AVZGuard, то контроль над ним будет потерян и потребуется перезагрузка компьютера.

15.18.2 function GetAVZGuardStatus

```
function GetAVZGuardStatus : boolean;
```

Возвращает текущий статус AVZGuard - true=включен, false=выключен.

Пример:

```
if GetAVZGuardStatus then
  AddToLog('AVZGuard активен')
else
  AddToLog('AVZGuard не активен');
```

15.19 Управление Boot Cleaner

15.19.1 function BC_Activate

```
function BC_Activate : boolean;
```

Производит активацию системы, установку драйвера и его настройку. Перед вызовом BC_Activate необходимо указать удаляемые объекты при помощи команд BC_DeleteFile, BC_DeleteReg, BC_DeleteSvc. Функция возвращает true в случае успешной установки и конфигурирования драйвера и false в случае ошибки.

Пример:

```
begin
  // Добавление в сценарий команды удаления драйвера PE386
  BC_DeleteSvc('PE386');
  // Активация драйвера
  BC_Activate;
  // Перезагрузка
  RebootWindows(true);
end.
```

15.19.2 function BC_DeActivate

```
function BC_DeActivate : boolean;
```

Производит удаление драйвера с диска и всех принадлежащих драйверу ключей и параметров из реестра. Возвращает true в случае успешного удаления драйвера.

См. также: [BC_Activate](#)^[119], [BC_Execute](#)^[119]

15.19.3 function BC_Execute

```
function BC_Execute : boolean;
```

Устанавливает драйвер и настраивает его, после чего производит принудительную загрузку драйвера, что приводит к выполнению поставленного драйверу задания немедленно, без перезагрузки. Может применяться для удаления объектов из режима ядра без перезагрузки, а также для отладочных целей. После успешного выполнения операций драйвер самоуничтожается, поэтому вызов BC_DeActivate необязателен.

Вызов BC_Execute не влияет на подготовленные другими командами настройки, поэтому возможен повторный вызов BC_Execute или BC_Activate без перенастройки.

Примеры:

begin

// Попытка удаления драйвера PE386 из режима ядра без перезагрузки

BC_DeleteSvc('PE386');

BC_Execute;

end.

begin

// Удаления драйвера PE386

BC_DeleteSvc('PE386');

BC_Activate;

end.

См. также: [BC_Activate](#)^[119]

15.19.4 function BC_Clear

function BC_Clear : boolean;

Выполняет очистку существующих настроек. В момент старта скрипта данная операция выполняется автоматически.

15.19.5 function BC_LogFile

function BC_LogFile(AFileName : **string**) : boolean;

Данная функция позволяет включить протоколирование и задать имя и местоположение файла протокола. Задание вместо имени протокола пустой строки отключает протоколирование. По умолчанию протоколирование отключено. Задание недопустимого имени файла или корректного имени на недоступном в момент загрузки системы диске не является ошибкой - в случае невозможности создания файла протоколирование отключится, но драйвер выполнить все предписанные в настройках операции.

Пример:

begin

// Добавление в сценарий команды удаления драйвера PE386

BC_DeleteSvc('PE386');

// Настройка протокола

BC_LogFile(GetAVZDirectory + 'boot_clr.log');

// Активация драйвера

BC_Activate;

// Перезагрузка

RebootWindows(true);

end.

15.19.6 function BC_QrFile

function BC_QrFile(AFileName : **string**) : boolean;

Добавляет команду карантина указанного файла в сценарий BootCleaner. В имени файлы допустимы [макросы](#)^[150]. Карантин файлов посредством BootCleaner полезен в случае, если вредоносная программа блокирует доступ к своим файлам при помощи руткит-механизмов, которые не удастся нейтрализовать встроенным антируткитом AVZ. Помещенные в карантин файлы доступны через строенные средства просмотра карантина AVZ, но файлы именуются как bcqrXXXXX.dat и bcqrXXXXX.ini

На заметку:

Команда карантина файла имеет приоритет над прочими командами.

begin

```
// Добавление в сценарий команды карантина файла
BC_QrFile('%WinDir%\trojan.exe');
// Настройка протокола
BC_LogFile(GetAVZDirectory + 'boot_clr.log');
// Активация
BC_Activate;
end.
```

15.19.7 function BC_QrSvc

function BC_QrSvc (ASvcName : **string**) : boolean;

Добавляет команду карантина указанной службы или драйвера. Помещенные в карантин файлы доступны через строенные средства просмотра карантина AVZ, но файлы именуются как bcqrXXXXX.dat и bcqrXXXXX.ini

begin

```
// Добавление в сценарий команды карантина файла
BC_QrSvc('PE386');
// Настройка протокола
BC_LogFile(GetAVZDirectory + 'boot_clr.log');
// Активация
BC_Activate;
end.
```

15.19.8 function BC_DeleteFile

function BC_DeleteFile (AFileName : **string**) : boolean;

Добавляет команду удаления файла с именем AFileName в сценарий BootCleaner. В имени файлы допустимы [макросы](#)^[150]

begin

```
// Добавление в сценарий команды удаления файла
BC_DeleteFile('%WinDir%\trojan.exe');
// Настройка протокола
BC_LogFile(GetAVZDirectory + 'boot_clr.log');
// Активация
BC_Activate;
end.
```

15.19.9 function BC_CopyFile

```
function BC_CopyFile (AFromName, AToName : string) : boolean;
```

Добавляет команду копирования файла AFromName -> AToName в сценарий BootCleaner. В именах файлов допустимы [макросы](#) ^[150]

Совместимость: AVZ 4.26 и выше

15.19.1function BC_DeleteReg

```
function BC_DeleteReg (AKeyName : string) : boolean;
```

Добавляет команду удаления ключа реестра с именем AKeyName в сценарий BootCleaner. Имя ключа реестра должно начинаться с имени раздела, допустимы следующие имена разделов: HKLM, HKCU, HKEY_LOCAL_MACHINE, HKEY_CURRENT_USER.

15.19.1function BC_DeleteSvcReg

```
function BC_DeleteSvcReg (ASvcName : string) : boolean;
```

Добавляет команду удаления ключа реестра для драйвера или службы с именем ASvcName в сценарий BootCleaner. Важно отметить, что исполняемый файл на диске при это не удаляется.

И имени службы/драйвера допустимы Unicode символы, в том числе символы с кодом 0. Для указания Unicode символов применяется специальная форма записи "\uXXXX", где XXXX - код символа. Например: **'Super\u0000Puper\u0000Driver'**

15.19.1function BC_DeleteSvc

```
function BC_DeleteSvc (ASvcName : string) : boolean;
```

Добавляет команду удаления драйвера или службы с именем ASvcName в сценарий BootCleaner. Удаление предполагает поиск и удаление исполняемого файла указанной службы с последующим удалением соответствующего ей ключа реестра.

И имени службы/драйвера допустимы Unicode символы, в том числе символы с кодом 0. Для указания Unicode символов применяется специальная форма записи "\uXXXX", где XXXX - код символа. Например: **'Super\u0000Puper\u0000Driver'**

15.19.1function BC_DisableSvc

```
function BC_DisableSvc (ASvcName : string) : boolean;
```

Добавляет команду отключения драйвера или службы с именем ASvcName в сценарий BootCleaner. Отключение предполагает установку статуса 4 (отключен) в соответствующем ключе реестра. Остальные параметры в реестре и файл на диске при этом не затрагиваются.

И имени службы/драйвера допустимы Unicode символы, в том числе символы с кодом 0. Для указания Unicode символов применяется специальная форма записи "\uXXXX", где XXXX - код символа. Например: **'Super\u0000Puper\u0000Driver'**

Пример:

```
begin  
  // Добавление в сценарий команды отключения драйвера "PE386"  
  BC_DisableSvc('PE386');  
  // Настройка протокола  
  BC_LogFile(GetAVZDirectory + 'boot_clr.log');  
  // Активация  
  BC_Activate;  
  // Перезагрузка  
  RebootWindows(true);  
end.
```

15.19.14function BC_ImportDeletedList

```
function BC_ImportDeletedList : boolean;
```

Выполняет импорт списка удаленных файлов в настройки Boot Cleaner. Список удаленных файлов ведется автоматически и пополняется при каждом вызове функции [DeleteFile](#)^[148]. Кроме того, возможно ручное добавление файлов в список удаленных при помощи SysCleanAddFile. Список удаленных файлов применяется для эвристической чистки системы, которая активируется командой ExecuteSysClean.

На заметку:

Вызов ExecuteSysClean очищает список удаленных файлов, поэтому необходимо использовать функцию ImportDeletedList пред вызовом ExecuteSysClean.

Пример:

```
begin  
  // Удаление файлов  
  DeleteFile('%WinDir%\Trojan1.exe');  
  DeleteFile('%WinDir%\Trojan2.exe');  
  DeleteFile('%WinDir%\Trojan3.exe');  
  // Импорт списка удаленных файлов в настройки Boot Cleaner  
  BC_ImportDeletedList;  
  // Чистка ссылок на удаленные файлы  
  ExecuteSysClean;  
  // Активация драйвера Boot Cleaner  
  BC_LogFile(GetAVZDirectory + 'boot_clr.log');  
  BC_Activate;  
  // Перезагрузка  
  RebootWindows(true);  
end.
```

15.19.15function BC_ImportQuarantineList

```
function BC_ImportQuarantineList : boolean;
```

Выполняет импорт списка файлов, для которых делались попытки карантина, в настройки Boot Cleaner. Список файлов ведется автоматически и пополняется при каждом вызове функции [QuarantineFile](#)^[114]. Для каждого файла из данного списка будет выполнена операция [BC_QrFile](#)

 120.

На заметку:

Если функции [QuarantineFile](#)^[114] не указано имя файла, то она добавляет в список карантина всевозможные варианты имени файла - результат его поиска разными методами, имя файла + пути к системным папкам. Это делается для того, чтобы повысить вероятность карантина нужных файлов для исследования

На заметку:

Boot Cleaner не применяет базу безопасных файлов AVZ, поэтому в карантин может быть добавлен абсолютно любой файл - блокировка добавления безопасных файлов на него не действует.

Пример:

begin

```
// Карантин файлов
QuarantineFile('%WinDir%\Trojan1.exe', '');
QuarantineFile('%WinDir%\Trojan2.exe', '');
QuarantineFile('%WinDir%\Trojan3.exe', '');
// Импорт списка карантина в настройки Boot Cleaner
BC_ImportQuarantineList;
// Активация драйвера Boot Cleaner
BC_LogFile(GetAVZDirectory + 'boot_clr.log');
BC_Activate;
// Перезагрузка
RebootWindows(true);
```

end.

15.19.1 function BC_ImportALL

function BC_ImportALL : boolean;

Импортирует в настройки Boot Cleaner всю информацию об операциях, выполненных до этого в скрипте при помощи [DeleteFile](#)^[148] и [QuarantineFile](#)^[114]. Применяется для упрощения интеграции Boot Cleaner с типовыми скриптами.

Пример:

begin

```
// Карантин файлов
QuarantineFile('%WinDir%\Trojan1.exe', '');
QuarantineFile('%WinDir%\Trojan2.exe', '');
// Удаление файлов
DeleteFile('%WinDir%\Trojan3.exe', '');
// Импорт списка карантина и удаленных файлов в настройки Boot Cleaner
BC_ImportALL;
// Активация драйвера Boot Cleaner
BC_LogFile(GetAVZDirectory + 'boot_clr.log');
BC_Activate;
// Перезагрузка
RebootWindows(true);
```

end.

15.19.1 Типовые примеры

Пример 1. Необходимо удалить в ходе перезагрузки файл c:\trojan.exe без протоколирования.
Решение:

```
begin  
  // Добавление в сценарий команды удаления драйвера PE386  
  BC_DeleteFile('c:\trojan.exe');  
  // Активация драйвера  
  BC_Activate;  
  // Перезагрузка  
  RebootWindows(true);  
end.
```

Пример 2. Необходимо выполнить попытку удаления файла c:\trojan.exe из ядра без перезагрузки с протоколированием. Протокол работы должен быть размещен в каталоге AVZ, имя файла протокола - boot_clr.log

```
begin  
  // Добавление в сценарий команды удаления драйвера PE386  
  BC_DeleteFile('c:\trojan.exe');  
  // Настройка протокола  
  BC_LogFile(GetAVZDirectory + 'boot_clr.log');  
  // Запуск  
  BC_Execute;  
end.
```

Пример 3. Необходимо удалить в ходе перезагрузки драйвер PE386. Протокол работы должен быть размещен в каталоге AVZ, имя файла протокола - boot_clr.log
Решение:

```
begin  
  // Добавление в сценарий команды удаления драйвера PE386  
  BC_DeleteSvc('PE386');  
  // Настройка протокола  
  BC_LogFile(GetAVZDirectory + 'boot_clr.log');  
  // Активация драйвера  
  BC_Activate;  
  // Перезагрузка  
  RebootWindows(true);  
end.
```

Пример 4. Необходимо удалить в ходе перезагрузки драйвер PE386, создав предварительно его копию в карантине. Протокол работы должен быть размещен в каталоге AVZ, имя файла протокола - boot_clr.log
Решение:

```
begin  
  // Добавление в сценарий команды карантина драйвера PE386  
  BC_QrSvc('PE386');
```

```
// Добавление в сценарий команды удаления драйвера PE386
BC_DeleteSvc('PE386');
// Настройка протокола
BC_LogFile(GetAVZDirectory + 'boot_clr.log');
// Активация драйвера
BC_Activate;
// Перезагрузка
RebootWindows(true);
end.
```

15.20 Эвристическая проверка системы

15.20.1 procedure SearchRootkit

```
procedure SearchRootkit(UserModeLock, KernelModeLock :
boolean);
```

UserModeLock - управляет блокировкой руткитов UserMode (false - только анализ, true - противодействие)

KernelModeLock - управляет блокировкой руткитов KernelMode (false - только анализ, true - противодействие)

15.20.2 procedure SearchKeylogger

```
procedure SearchKeylogger;
```

Осуществляет поиск кейлоггеров и троянских DLL. Эта операция производится в ходе сканирования, а данная функция позволяет выполнить ее по мере надобности индивидуально, в любом месте скрипта. Результаты анализа помещаются в протокол.

Совместимость:

Версия 4.26 и выше

15.20.3 procedure ExecuteSysChkEV

```
procedure ExecuteSysChkEV;
```

Выполняет эвристическую проверку системы (ЭПС). По умолчанию эвристическая проверка производится в ходе сканирования компьютера при условии, что установлен переключатель "Эвристическая проверка системы" на главном окне. Результаты проверки заносятся в протокол.

Совместимость:

Версия 4.26 и выше

15.20.4 procedure ExecuteSysChkIPU

```
procedure ExecuteSysChkIPU;
```

Выполняет поиск потенциальных уязвимостей (ИПУ). По умолчанию поиск потенциальных уязвимостей ведется в ходе сканирования компьютера при условии, что установлен переключатель "Поиск потенциальных уязвимостей" на главном окне. Результаты проверки заносятся в протокол.

Совместимость:

Версия 4.26 и выше

15.20.5 procedure ExecuteSysCheck

```
procedure ExecuteSysCheck(AFileName : string);
```

Выполнение исследования системы с сохранением HTML протокола в указанный файл (одновременно с созданием HTML протокола создается содержащий его zip одноименный файл).

Пример:

```
begin
```

```
    // Выполнить исследование системы
```

```
    ExecuteSysCheck(GetAVZDirectory + 'virusinfo_syscheck.htm');
```

```
end.
```

15.20.6 procedure ExecuteSysCheckEx

```
procedure ExecuteSysCheckEx(AFileName : string; ACheckMode : dword;  
                           AScanAllSvc : boolean; ARepParams : dword);
```

Расширенный аналог [ExecuteSysCheck](#)^[127]. Содержит ряд параметров, позволяющих выполнить тонкую настройку исследования системы и формирования протокола.

Параметры:

AFileName - имя файла. Имя файла должно быть с расширением HTM, имена файлов XML и ZIP формируются из него путем замены расширения

ACheckMode - битовая маска, задающая режим проверки. Каждый бит соответствует определенной проверке (бит (вес)):

- 0 (1) - процессы
- 1 (2) - библиотеки процессов
- 2 (4) - Службы и драйверы
- 3 (8) - Модули пространства ядра
- 4 (16) - Автозапуск
- 5 (32) - SPI/LSP провайдеры
- 6 (64) - Открытые порты TCP/UDP
- 7 (128) - Расширения IE
- 8 (256) - Расширения Explorer
- 9 (512) - Расширения системы печати
- 10 (1024) - Задания планировщика
- 11 (2048) - Файл Hosts
- 12 (4096) - Обработчики протоколов
- 13 (8192) - Downloaded Program Files
- 14 (16384) - Апплеты панели управления (CPL)
- 15 (32768) - Active Setup

AScanAllSvc - режим сканирования служб и драйверов. Если данный параметр равен True, то в протокол помещаются все службы и файлы, если false - только активные

ARepParams - битовая маска с параметрами, управляющими формированием отчета

- 0 (1) - режим фильтрации чистых файлов. Если данный бит установлен, то чистые объекты не помещаются в протокол
- 1 (2) - если бит установлен, то в XML отчет помещаются данные о всех файлах

- 2 (4) - если бит установлен, то в XML отчет помещаются данные только о файлах, которые не опознаны по базе безопасных
- 3 (8) - после формирования архива удалить HTML файл
- 4 (16) - после формирования архива удалить XML файл
- 5 (32) - не создавать ZIP архив

Совместимость:

Версия 4.28 и выше

Пример:

begin

```
ExecuteSysCheckEX('C:\avz_rep.htm', $FFFFFFFF, true, 1+2+16+32);  
end.
```

В данном примере выполняются все проверки (\$FFFFFFFF - все биты "1"), изучаются активные и неактивные службы и драйверы, из HTML протокола исключаются данные о чистых файлах, в XML отчет включаются данные о всех файлах, файлы HTML и XML после формирования архива удаляются с диска

15.20.7 function ExecuteWizard

```
function ExecuteWizard(ADBName : string; ACheckLevel, AFixLevel : integer;  
                        AUseBackup : boolean) : integer;
```

Выполняет проверку системы с помощью мастера поиска и устранения проблем. Параметры позволяют задать уровень опасности проблем и уровень опасности для автоматического устранения. Функция ведет базу отката, которая может использоваться для отката сделанных изменений.

Параметры:

ADBName - имя базы. Допустим следующие имена:

TSW - база мастера поиска и устранения проблем

BT - база проверки настроек безопасности браузеров и системы в целом

PRT - база мастера для чистки из системы данных, влияющих на приватность (протоколы, журналы, кукизы и т.п.)

ACheckLevel - порог срабатывания. Степень тяжести проблемы отсчитывается от 1 до 3 (1-незначительные проблемы и ошибки, 2-проблемы средней тяжести, 3-опасные ошибки и проблемы). Указание значения за пределами 1..3 приведет к ошибке и функция не отработает

AFixLevel - порог срабатывания автоматического исправления проблем. Степень тяжести проблем сравнивается с данным порогом и если степень тяжести больше или равна порогу, то производится автоматическое исправление проблемы. Допустимые значения 1..3, указание значения -1 отключает систему автоматического исправления и функция ExecuteWizard работает в режиме сбора информации

AUseBackup - если данный параметр равен true, то произведенные функцией изменения записываются в базу отката

Возвращаемое значение: функция возвращает -1 в случае ошибки. Если функция выполнена успешно, то она возвращает количество найденных проблем, уровень которых больше или равен порогу ACheckLevel

Совместимость:

Версия 4.28 и выше

Примеры:

```
var
  X : integer;
begin
  X := ExecuteWizard('TSW', 2, -1, false);
  AddToLog('Количество найденных проблем = '+inttostr(X));
end.
```

Данный скрипт загружает базу TSW (мастер поиска и устранения проблем), порог срабатывания равен 2, автоматическое исправление отключено.

```
var
  X : integer;
begin
  X := ExecuteWizard('TSW', 2, 3, true);
  AddToLog('Количество найденных проблем = '+inttostr(X));
end.
```

Данный скрипт загружает базу TSW (мастер поиска и устранения проблем), порог срабатывания равен 2, порог для автоматического исправления равен 3 (исправляются только тяжелые ошибки), вносимые изменения вносятся в базу отката.

15.21 Эвристическая чистка системы

15.21.1 function ExecuteSysClean

```
function ExecuteSysClean : boolean;
```

Выполняет эвристическую чистку системы. Эвристическая чистка состоит в анализе реестра и поиске ссылок на удаленные файлы. Список файлов по умолчанию формируется автоматически - он пополняется при каждом вызове функции [DeleteFile](#)^[148]. После выполнения эвристической чистки системы этот список очищается автоматически.

Вызов данной функции приводит к созданию в протоколе отметки "Автоматическая чистка следов удаленных в ходе лечения программ". Если в ходе чистки были обнаружены и удалены какие-либо элементы реестра, то в протоколе делаются соответствующие отметки.

Пример:

```
begin
  DeleteFile('%WinDir%\trojan.exe');
  DeleteFile('C:\worm.exe');
  // Чистка ссылок на удаленные файлы
  ExecuteSysClean;
end.
```

15.21.2 procedure SysCleanAddFile

```
procedure SysCleanAddFile(AFileName : string);
```

Добавляет файл с именем AFileName в список, используемый функцией эвристической чистки системы [ExecuteSysClean](#)^[129]. Добавление файла в список при помощи SysCleanAddFile позволяет автоматизированно удалить следы файлов, стерты ранее вручную или при помощи другой антивирусной программы.

```
begin
```

```
  // Удаление файла
```

```
  DeleteFile('%WinDir%\trojan.exe');
```

```
  // Добавление файла в список очистки вручную
```

```
  SysCleanAddFile('C:\worm.exe');
```

```
  // Чистка ссылок на удаленные файлы
```

```
  ExecuteSysClean;
```

```
end.
```

15.21.3 procedure SysCleanDelFilesList

```
procedure SysCleanDelFilesList;
```

Очищает список файлов, ссылки на которые будет искать и удалять подсистема эвристической чистки системы. Очистка пустого списка (и, как следствие, повторные вызовы функции) не являются ошибкой.

На заметку:

В момент запуска скрипта список файлов пустой и его дополнительная очистка вызовом SysCleanDelFilesList не требуется.

Пример:

```
begin
```

```
  // Удаление файла
```

```
  DeleteFile('%WinDir%\trojan.exe');
```

```
  // Добавление файла в список очистки вручную
```

```
  SysCleanAddFile('C:\worm.exe');
```

```
  // Чистка ссылок на удаленные файлы
```

```
  ExecuteSysClean;
```

```
end.
```

15.22 Восстановление системы и стандартные скрипты

15.22.1 function ExecuteRepair

```
function ExecuteRepair(ID : integer) : boolean;
```

Выполняет микропрограмму восстановления системы с указанным номером. Номера микропрограмм можно посмотреть в диалоговом окне "Восстановление системы". Функция возвращает true в случае успешного выполнения указанной микропрограммы и false в случае

ошибки.

Пример:

```
begin  
  ExecuteRepair(3);  
  ExecuteRepair(5);  
end.
```

15.22.2 function ExecuteStdScr

```
function ExecuteStdScr(ID : integer) : boolean;
```

Выполняет скрипт из раздела "Стандартные скрипты" с указанным номером. Номера скриптов можно посмотреть в диалоговом окне "Стандартные скрипты". Функция возвращает true в случае успешного выполнения указанной микропрограммы и false в случае ошибки.

Пример:

```
begin  
  ExecuteStdScr(3);  
end.
```

15.23 Завершение работы и перезагрузка

15.23.1 procedure ShutdownWindows

```
procedure ShutdownWindows(AForce : boolean);
```

Завершение работы системы. Флаг AForce управляет режимом завершения работы - если он равен true, то завершение работы производится в ускоренном (форсированном) режиме, без рассылки приложениям сообщений о том, что система завершает работу. Завершение работы в формированном режиме может привести к потере открытых документов и некорректному завершению работы программ.

15.23.2 procedure RebootWindows

```
procedure RebootWindows(AForce : boolean);
```

Перезагрузка системы. Флаг AForce управляет режимом перезагрузки - если он равен true, то завершение работы производится в ускоренном (форсированном) режиме, без рассылки приложениям сообщений о том, что система завершает работу.

Пример:

```
begin  
  RebootWindows(false);  
end.
```

15.24 Использование AV анализатора в скрипте

15.24.1 function CheckFile

```
function CheckFile(AFileName : string) : integer;
```

Выполняет проверку указанного файла при помощи AV базы и базы безопасных AVZ.

Результат анализа определяется по коду возврата:

- 1 - файл не найден или к нему нет доступа
- 0 - файл не значится в базе безопасных и не опознан как вредоносный
- 1 - файл опознан как вредоносный объект
- 2 - файл не значится в базе безопасных и подозревается как вредоносный объект
- 3 - файл опознан по базе безопасных AVZ или базе Microsoft

Если функция возвращает код 1 или 2, то при помощи [GetLastCheckTxt](#)^[132] можно получить уточняющую текстовую информацию (имя malware и прочую информацию).

15.24.2 function GetLastCheckTxt

```
function GetLastCheckTxt : string;
```

Возвращает текстовое описание объекта, детектированного файловым сканером. Имеет смысл только после вызова [CheckFile](#)^[132], возвратившего код 1 или 2. Во всех остальных случаях функция возвращает пустую строку.

15.25 Работа с CLSID и BHO

15.25.1 function CLSIDExists

```
function CLSIDExists(CLSID : string) : boolean;
```

Проверяет, существует ли указанный CLSID в реестре. CLSID рекомендуется передавать без фигурных скобок.

Если CLSID обнаружен в реестре, то возвращается true. Иначе - false.

Пример:

```
if CLSIDExists('B5F3970B-745E-46AC-B890-E08F69777D80') then  
  AddToLog('CLSID найден !!');
```

15.25.2 function CLSIDFileExists

```
function CLSIDFileExists(CLSID : string) : boolean;
```

Возвращает true, если указанный CLSID есть в реестре, по его анализу удалось найти связанный с CLSID файл и этот файл существует на диске. Возвращает false в любой другой ситуации (нет CLSID, запись CLSID в реестре).

Пример:

```
if CLSIDFileExists('B5F3970B-745E-46AC-B890-E08F69777D80')
then
  AddToLog(CLSIDFileName('B5F3970B-745E-46AC-B890-E08F69777D80'
) +
          ' >> ' +
          'подозрение на AdvWare.ToolBar.SearchIt');
```

15.25.3 function CLSIDFileName

```
function CLSIDFileName(CLSID : string) : string;
```

Возвращает имя файла, соответствующего указанному CLSID или пустую строку, если CLSID не найден в реестре или его анализ не позволил найти имя файла. Функция не проверяет, существует файл на диске или нет.

Пример:

```
var
  S : string;
begin
  S := 'B5F3970B-745E-46AC-B890-E08F69777D80';
  if CLSIDFileExists(S) then
    AddToLog(CLSIDFileName(S) +
            ' >> ' +
            'подозрение на AdvWare.ToolBar.SearchIt');
end.
```

15.25.4 procedure DelCLSID

```
procedure DelCLSID(CLSID : string);
```

Удаляет из реестра регистрацию класса с заданным CLSID. Отсутствие класса с указанным CLSID не является ошибкой.

Удаление регистрации предполагает выполнение следующих операций:

- Удаление регистрации класса
- Удаление интерфейса с таким же CLSID (если он существует)
- Удаление ВХО и панелей расширения IE с данным CLSID
- Удаление записей ActiveSetup с данным CLSID
- Удаление расширений проводника с данным CLSID
- Удаление элементов SharedTaskScheduler

Обратите внимание, что CLSID должен передаваться без фигурных скобок !!

Пример:

```
begin
  DelCLSID('B5F3970B-745E-46AC-B890-E08F69777D80');
end.
```

15.25.5 function BHOExists

```
function BHOExists(BHO : string) : boolean;
```

Проверяет, существует ли BHO или иное расширение Internet Explorer с указанным CLSID. CLSID рекомендуется передавать без фигурных скобок. Если BHO обнаружен в реестре, то возвращается true. Иначе - false.

15.25.6 procedure DelBHO

```
procedure DelBHO(BHO : string);
```

Удаляет BHO с указанным CLSID. CLSID рекомендуется передавать без фигурных скобок.

15.26 Работа со службами и драйверами

15.26.1 function ServiceExists

```
function ServiceExists(AName : string) : boolean;
```

Выполняет поиск регистрационных данных в реестре. Возвращает true при условии, что в реестре зарегистрирована служба или драйвер с указанным именем AName.

Пример:

```
begin  
  if ServiceExists('PE383') then  
    AddToLog('Обнаружен драйвер руткита PE386');  
end.
```

15.26.2 function ServiceAndFileExists

```
function ServiceAndFileExists(AName : string) : boolean;
```

Выполняет поиск регистрационных данных в реестре и соответствующего файла на диске. Возвращает true при условии, что в реестре зарегистрирована служба или драйвер с указанным именем, и соответствующий ей файл найден на диске.

15.26.3 function StartService

```
function StartService(AName : string) : boolean;
```

Выполняет запуск службы или загрузку драйвера с указанным именем.

15.26.4 function StopService

```
function StopService(AName : string) : boolean;
```

Выполняет остановку службы или выгрузку драйвера с указанным именем.

15.26.5 function DeleteService

```
function DeleteService (AName : string; ADelFile : boolean =  
false) : boolean;
```

Удаляет из реестра службу или драйвер с именем AName. Функция возвращает true в случае успешного удаления службы и false в случае ошибки. Вторым параметром ADelFile (параметр необязательный и его можно не указывать) задает режим удаления. По умолчанию он равен false - в результате при удалении регистрационных данных драйвера из реестра файл на диске не удаляется. Если ADelFile=true, то после удаления данных из реестра производится попытка удаления файла драйвера с диска.

Пример:

```
begin  
DeleteService('PE386', true);  
end.
```

15.26.6 function GetServiceFile

```
function GetServiceFile (AName : string) : string;
```

Возвращает полное имя исполняемого файла указанной службы или драйвера. Если служба или драйвер с указанным именем не найдена, то возвращается пустая строка.

15.26.7 function GetServiceStart

```
function GetServiceStart (AServiceName: string): integer;
```

Возвращает код режима автозапуска службы или драйвера с именем AServiceName.

Коды автозапуска:

- 1 - ошибка определения кода автозапуска
- 1 - загрузка в ходе начальной загрузки системы
- 2 - режим AUTO - автоматическая загрузка
- 3 - MANUAL - загрузка вручную (по команде пользователя или по команде прикладной программы)
- 4 - загрузка запрещена

Совместимость: AVZ 4.26 и выше

Пример:

```
var  
ASvc : string;  
begin  
ASvc := 'RemoteRegistry';  
if GetServiceStart (ASvc) < 4 then  
AddToLog ('>> разрешена потенциально опасная служба '+  
ASvc+' ('+GetServiceName (ASvc)+' ) ');  
end.
```

15.26.8 function SetServiceStart

```
function SetServiceStart (AServiceName: string; AStart: integer): boolean;
```

Устанавливает код режима автозапуска равный AStart для службы или драйвера с именем AServiceName. Возвращает True в случае успешной установки кода режима и false в случае ошибки.

Коды автозапуска:

- 1 - загрузка в ходе начальной загрузки системы
- 2 - режим AUTO - автоматическая загрузка
- 3 - MANUAL - загрузка вручную (по команде пользователя или по команде прикладной программы)
- 4 - загрузка запрещена

Совместимость: AVZ 4.26 и выше

Пример:

begin

```
if SetServiceStart('RemoteRegistry', 4) then
  AddToLog('Служба RemoteRegistry успешно отключена');
end.
```

15.26.9 function GetServiceName

function GetServiceName(AServiceName: **string**): **string**;

Возвращает отображаемое имя службы или драйвера с именем AServiceName. Отображаемое имя является необязательным и хранится в реестре, его удобно применять при формировании протоколов в качестве уточняющей информации. В случае ошибки функция возвращает пустую строку.

Совместимость: AVZ 4.26 и выше

Пример:

var

ASvc : **string**;

begin

```
ASvc := 'RemoteRegistry';
if GetServiceStart(ASvc) < 4 then
  AddToLog('>> разрешена потенциально опасная служба ' +
    ASvc+ ' ('+GetServiceName(ASvc)+') ');
```

end.

15.27 Работа с SPI/LSP

15.27.1 procedure DelSPIByFileName

procedure DelSPIByFileName(ALibrary : **string**; NameCompare : **boolean**);

Удаляет SPI/LSP провайдер по имени библиотеки-поставщика ALibrary. Параметр NameCompare управляет алгоритмом сравнения имен - если он равен false, то производится сравнение полного имени файлов. Если NameCompare=true, то сравниваются только имена файлов, без пути. Сравнение имен файлов идет без учета регистра, имена файлов перед сравнением подвергаются нормализации.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые](http://z-oleg.com/secr)

[в именах файлов](#) 

15.27.2 procedure CheckSPI

```
procedure CheckSPI;
```

Выполняет проверку настроек SPI/LSP. В случае обнаружения ошибок информация вносится в протокол.

Совместимость: AVZ 4.26 и выше

15.27.3 procedure AutoFixSPI

```
procedure AutoFixSPI;
```

Выполняет анализ настроек SPI/LSP и автоматически исправляет все обнаруженные ошибки. Начиная с версии 4.26 перед исправлением ошибок при вызове данной функции AVZ автоматически создает резервную копию настроек в папке BackUp. Резервная копия является стандартным REG файлом.

На заметку

Принцип поиска ошибок в частности содержит удаление ссылок на несуществующие провайдеры. Поэтому при использовании AutoFixSPI в конце скрипта, удаляющего файлы, может возникнуть ситуация, когда AVZ в ходе выполнения DeleteFile не смог удалить файл и записал его на отложенное удаление. Следовательно, в ходе выполнения команды AutoFixSPI анализатор обнаружит файл и не посчитает ссылку на него ошибкой. Эту ситуацию нужно учитывать при удалении файлов, являющихся SPI провайдерами. Выходом из положения является явное удаление SPI провайдера по имени файла.

На заметку

Анализатор считает отсутствующими библиотеки провайдеров в случае, если ему не удастся найти файл на диске. Однако в некоторых случаях возможны различные сбои в работе анализатора - например в случае, если AVZ запущен из под учетной записи пользователя или из терминального сеанса на сервере.

15.28 Работа с автозапуском и Winlogon

15.28.1 procedure DelWinlogonNotifyByFileName

```
procedure DelWinlogonNotifyByFileName (AFileName : string);
```

Производит удаление из реестра регистрации элемента Winlogon, соответствующего заданному имени файла. В качестве AFileName можно передать только имя файла или полное имя файла, включая путь.

Пример:

```
begin
```

```
  DelWinlogonNotifyByFileName ('1_32bean32_1.dll');
```

```
end.
```

15.28.2 procedure DelWinlogonNotifyByKeyName

```
procedure DelWinlogonNotifyByKeyName (AKeyName : string);
```

Удаляет элемент Winlogon по известному имени ключа в реестре.

Пример:

```
begin  
  DelWinlogonNotifyByKeyName ('1_32bean32_1reg');  
end.
```

15.28.3 function DelAutorunByFileName

```
function DelAutorunByFileName (AKeyName : string) : boolean;
```

Удаляет элемент автозапуска по имени файла. В качестве AFileName можно передать только имя файла или полное имя файла, включая путь.

Совместимость: AVZ 4.26 и выше

Пример:

```
begin  
  DelAutorunByFileName ('trojan.dll');  
end.
```

15.29 Работа с реестром

15.29.1 function RegKeyExists

```
function RegKeyExists (ARoot, AName : string) : boolean;
```

Возвращает true, если в реестре существует ключ AName в разделе ARoot, и false в случае отсутствия ключа или указания недопустимых параметров.

Пример:

```
begin  
  if RegKeyExists ('HKLM', 'Software\Gator') then  
    AddToLog ('Найден ключ реестра, принадлежащий Gator');  
end.
```

15.29.2 procedure RegKeyDel

```
procedure RegKeyDel (ARoot, AName : string);
```

Удаляет указанный ключ реестра. AName - имя ключа, ARoot - имя раздела
Попытка удаления несуществующего ключа не является ошибкой.

15.29.3 procedure RegKeyCreate

```
procedure RegKeyCreate (ARoot, AName : string);
```

Создает ключ реестра с указанным именем. AName - имя ключа, ARoot - имя раздела

15.29.4 function RegKeyParamExists

```
function RegKeyParamExists (ARoot, AName, AParam : string) :  
boolean;
```

Возвращает TRUE, если в ключе с именем AName существует параметр AParam.

15.29.5 procedure RegKeyParamDel

```
procedure RegKeyParamDel (ARoot, AName, AParam : string);
```

Производит удаление параметра с именем AParam ключа AName в разделе ARoot. Указание несуществующего параметра, ключа или раздела не является ошибкой.

15.29.6 function RegKeyStrParamRead

```
function RegKeyStrParamRead (ARoot, AName, AParam : string) :  
string;
```

Производит чтение текстового параметра с именем AParam ключа AName раздела ARoot. Указание несуществующего параметра, ключа или раздела не является ошибкой - в этом случае функция возвращает пустую строку.

15.29.7 function RegKeyIntParamRead

```
function RegKeyIntParamRead (ARoot, AName, AParam : string) :  
dword;
```

Производит чтение параметра типа REG_DWORD с именем AParam ключа AName раздела ARoot. Указание несуществующего параметра, ключа или раздела не является ошибкой - в этом случае функция возвращает 0.

15.29.8 procedure RegKeyStrParamWrite

```
procedure RegKeyStrParamWrite (ARoot, AName, AParam, AValue :  
string);
```

Записывает значение AValue в параметр AParam типа REG_SZ (строка) ключа реестра AName в разделе ARoot. В случае отсутствия ключа или параметра они автоматически создаются.

15.29.9 procedure RegKeyIntParamWrite

```
procedure RegKeyIntParamWrite (ARoot, AName, AParam : string;  
AValue : dword);
```

Записывает значение AValue в параметр AParam типа REG_DWORD (числовое значение) ключа реестра AName в разделе ARoot. В случае отсутствия ключа или параметра они автоматически создаются.

15.29.1procedure RegKeyBinParamWrite

```
procedure RegKeyBinParamWrite(ARoot, AName, AParam : string;  
AValue : dword);
```

Записывает значение AValue в параметр AParam типа REG_BINARY (бинарное значение произвольной длины) ключа реестра AName в разделе ARoot. В случае отсутствия ключа или параметра они автоматически создаются.

Значение бинарного параметра передается в текстовом виде, побайтно. Каждый байт кодируется в шестнадцатеричном формате, разделителями является запятая. Предшествующие нули в числах и пробелы игнорируются.

Пример:

```
begin  
  RegKeyBinParamWrite('HKEY_LOCAL_MACHINE',  
  
  'SOFTWARE\Microsoft\Windows\CurrentVersion',  
    'Test',  
    '011,23,5, F2,0E4,1B');  
end.
```

15.29.1function RegSearch

```
function RegSearch(ARoot, AName, AValue : string) : boolean;
```

Производит сканирование ключа AName раздела реестра ARoot и осуществляет поиск образца AValue. Информация о всех найденных ключах выводится в протокол. Если AName равен пустой строке, то сканируются все ключи раздела ARoot. Сравнение с образцом идет без учета регистра, допускаются частичные совпадения. Функция полезна для поиска в реестре определенных данных, например ссылок на файл с известным именем. Поиск может занять длительное время (до нескольких минут).

Пример:

```
begin  
  RegSearch('HKLM', '', 'svchost.exe');  
end.
```

Данный пример производит поиск в реестре и выводит в протокол информацию о всех вхождениях 'svchost.exe' в реестре.

15.29.1function ExpRegKey

```
function ExpRegKey(ARoot, AName, AFileName : string) : boolean;
```

Выполняет экспорт ключа реестра. Экспортируется ключ AName раздела реестра ARoot в файл с именем AFileName. Формат файла - стандартный REG файл, рекомендуемое расширение - ".reg". Полученный в ходе экспорта ключ можно импортировать в реестра редактором реестра (меню "Файл\Импорт"). По умолчанию стандартная системная реакция на открытие REG файла

так-же сводится к попытке его импорта. Функция возвращает true в случае успешного выполнения и false в случае ошибки.

Экспорт рекомендуется в качестве средства резервного копирования в скриптах, выполняющих модификацию ключей реестра.

Совместимость: AVZ 4.26 и выше

Пример:

begin

// Экспорт ключа реестра в файл

ExpRegKey('HKEY_CURRENT_USER', 'Software\Microsoft\Windows\CurrentVersi

// Удаление ключа реестра

RegKeyDel('HKEY_CURRENT_USER', 'Software\Microsoft\Windows\CurrentVers

end.

15.29.1function ExpRegKeyEx

function ExpRegKeyEx(ARoot, AName : **string**; Lines : TStrings) : boolean

Выполняет экспорт ключа реестра. Экспортируется ключ AName раздела реестра ARoot в массив строк Lines. Функция возвращает true в случае успешного выполнения и false в случае ошибки. Данная функция по принципу работы идентична функции [ExpRegKey](#), но ее удобно

од н

Экспорт рекомендуется в качестве средства резервного копирования в скриптах, выполняющих модификацию ключей реестра.

Совместимость: AVZ 4.26 и выше

15.29.1function BackupRegKey

function BackupRegKey(ARoot, AName, ABackupName : **string**) : boolean;

Выполняет резервное копирование ключа реестра. По принципу работы данная функция аналогична функции [ExpRegKey](#)^[140] - она выполняет экспорт ключа AName раздела реестра ARoot. Отличие состоит в том, что файл с результатами экспорта создается в папке Backup AVZ, имя файлов состоит из префикса ABackupName и текущей даты-времени. Если данный файл уже существует на диске (т.е. скрипт выполнил два вызова BackupRegKey с одинаковым ABackupName в течение одной секунды, то к имени файла добавляется порядковый номер. Префикс позволяет дать смысловое имя файлу для того, чтобы упростить его поиск на диске. Допускается не указывать ABackupName - в этом случае имя файла будет содержать только дату и время. Если резервируемый ключ отсутствует в реестре, то файл на диске не создается.

Резервное копирование рекомендуется применять в скриптах, выполняющих модификацию ключей реестра.

Совместимость: AVZ 4.26 и выше

Пример:

begin

// Резервная копия ключа реестра

BackupRegKey('HKEY_CURRENT_USER', 'Software\Microsoft\Windows\CurrentVe

```
// Удаление ключа реестра  
RegKeyDel('HKEY_CURRENT_USER', 'Software\Microsoft\Windows\CurrentVers  
end.
```

15.29.1function RegKeyEnumVal

```
function RegKeyEnumVal(ARoot, AName : string; AList : TStrings);
```

Строит список вложенных ключей ключа реестра AName. Список AList должен быть создан и инициализирован перед вызовом данной функции.

Совместимость: AVZ 4.28 и выше

Пример:

```
var  
  Lines : TStrings;  
  i : integer;  
begin  
  Lines := TStringList.Create;  
  RegKeyEnumVal('HKCU', 'Software\Microsoft\Windows\CurrentVersion\Inter  
  for i:= 0 to Lines.Count-1 do  
    AddToLog(Lines[i]);  
  Lines.Free;  
end.
```

15.29.1function RegKeyEnumKey

```
function RegKeyEnumKey(ARoot, AName : string; AList : TStrings);
```

Строит список вложенных ключей ключа реестра AName. Список AList должен быть создан и инициализирован перед вызовом данной функции.

Совместимость: AVZ 4.28 и выше

Пример:

```
var  
  Lines : TStrings;  
  i : integer;  
begin  
  Lines := TStringList.Create;  
  RegKeyEnumKey('HKCU', 'Software\Microsoft\Windows\CurrentVersion\Inter  
  for i:= 0 to Lines.Count-1 do  
    AddToLog(Lines[i]);  
  Lines.Free;  
end.
```

15.29.1Имена разделов

В именовании разделов допустимы следующие строки:
'HKEY_LOCAL_MACHINE', 'HKLM' для HKEY_LOCAL_MACHINE
'HKEY_CLASSES_ROOT', 'HKCR' для HKEY_CLASSES_ROOT
'HKEY_CURRENT_USER', 'HKCU' для HKEY_CURRENT_USER

'HKEY_CURRENT_CONFIG', 'HKCC' для HKEY_CURRENT_CONFIG
'HKEY_USERS' для HKEY_USERS
'HKEY_PERFORMANCE_DATA' для HKEY_PERFORMANCE_DATA
'HKEY_DYN_DATA' - для HKEY_DYN_DATA

Обратите внимание, что ключ указывается относительно выбранного раздела, т.е. если необходимо работать с ключом HKEY_LOCAL_MACHINE\Software\TrojanKey, то имя раздела будет 'HKEY_LOCAL_MACHINE', а ключ - "Software\TrojanKey". Регистр символов в имени раздела не играет роли.

15.30 Работа с INI файлами

15.30.1 function INIStrParamRead

```
function INIStrParamRead(AFileName, ASectionName, AParamName, ADefVal
```

Осуществляет чтение параметра с именем AParamName из секции ASectionName файла INI с именем AFileName. Если такой параметр существует, то возвращается его значение, если параметр не существует - то возвращается значение по умолчанию ADefVal.

Совместимость: AVZ 4.26 и выше

15.30.2 function INIStrParamWrite

```
function INIStrParamWrite(AFileName, ASectionName, AParamName, AParamV
```

Осуществляет запись параметра с именем AParamName из секции ASectionName файла INI с именем AFileName. Если такой параметр не существует, то он автоматически, аналогично происходит с секцией и файлом. Функция возвращает true в случае успешного выполнения и false в случае ошибки.

Совместимость: AVZ 4.26 и выше

15.30.3 function INISectionExists

```
function INISectionExists(AFileName, ASectionName : string) : boolean
```

Выполняет проверку, существует ли в INI файле с именем AFileName секция с именем ASectionName. Возвращает true, если секция существует и false в случае любой ошибки или отсутствия секции.

Совместимость: AVZ 4.26 и выше

15.30.4 function INIEraseSection

```
function INIEraseSection(AFileName, ASectionName : string) : boolean
```

Выполняет удаление из INI файла с именем AFileName секции с именем ASectionName. Возвращает true, если удаление успешно выполнено и false в случае ошибки.

Совместимость: AVZ 4.26 и выше

15.30.5 function INIEraseParam

```
function INIEraseParam(AFileName, ASectionName, AParamName : string) :
```

Выполняет удаление из INI файла с именем AFileName параметра AParamName секции

ASectionName. Возвращает true, если удаление успешно выполнено и false в случае ошибки.

Совместимость: AVZ 4.26 и выше

15.31 Работа с файлом Hosts

15.31.1 function GetHostsFileName

```
function GetHostsFileName : String;
```

Возвращает полное имя файла Hosts в системе. Функция учитывает тот факт, что в NT местоположение файла Hosts может быть настроено через реестр.

15.31.2 function ClearHostsFile

```
function ClearHostsFile : boolean;
```

Выполняет очистку файла Hosts. В файле Hosts остаются все комментарии из исходного файла, все значащие строки удаляются и добавляется единственная значащая строка строка "127.0.0.1 localhost". Функция возвращает true в случае успешного запуска и False в случае ошибки

15.32 Работа с файлами и папками

15.32.1 function GetCurrentDirectory

```
function GetCurrentDirectory : string
```

Функция возвращает текущий каталог. Имя каталога нормализовано и содержит '\' на конце.

Примеры:

```
begin
```

```
  // Вывод в протокол текущего каталога
```

```
  AddToLog('Текущий каталог = '+GetCurrentDirectory);
```

```
end.
```

```
begin
```

```
  // Сохранение протокола в текущий каталог
```

```
  SaveLog(GetCurrentDirectory+'avz_log.txt');
```

```
end.
```

15.32.2 function GetSystemDisk

```
function GetSystemDisk : string;
```

Функция возвращает букву диска, на котором размещена система (папка Windows, ProgramFiles ...). Данная настройка полезна для сканирования всего диска, на котором размещены системные папки

Пример:

```
begin
  // Настройка - сканировать системный диск
  SetupAVZ ('SCAN='+GetSystemDisk+':\');
end.
```

15.32.3 function GetTempDirectoryPath

```
function GetTempDirectoryPath : String;
```

Возвращает путь к системной папке для временных файлов. Возвращаемый путь содержит слеш на конце, например "c:\windows\temp"

15.32.4 function NormalDir

```
function NormalDir(ADirName : string) : string;
```

Производит нормализацию имени папки. Нормализация предполагает:

- Удаление пробелов в начале и конце имени папки
- Производит замену "/" на "\"
- Производит удаление повторяющихся слешей (например, C:\\abc). Исключение - это парный слеш в начале сетевого пути
- При необходимости добавляет слеш в конце пути
- Замену макросов на их значения. Подробнее про макросы см. в разделе ["макросы, допустимые в именах файлов"](#)^[150]

Данная функция удобна в различных процедурах, производящих обработку каталогов.

Примеры:

```
NormalDir('c:\test') = 'c:\test\'
NormalDir('c:\\test') = 'c:\test\'
NormalDir('c:\\test\abc/') = 'c:\test\abc\'
NormalDir('%SysDisk%\test') = 'c:\test\'
```

15.32.5 function NormalFileName

```
function NormalFileName (AName : string) : string;
```

Возвращает полное нормализованное имя файла. Нормализация имени предполагает:

1. Замену [макросов](#)^[150] значениями
2. Обработку префиксов типа "\\?\\", "\\System32"
3. Поиск файла в случае, если не указан полный путь к нему

15.32.6 function ExtractFileName

```
function ExtractFileName (AName : string) : string;
```

Извлекает имя файла из полного имени. Например, если передать функции полное имя файла "c:\virus\file.exe", то функция вернет "file.exe".

В имени файла допустимо указывать макросы, подробнее см. в разделе ["макросы, допустимые в именах файлов"](#)^[150]

Пример:**begin**`AddToLog (ExtractFileName ('c:\windows\virus.exe')) ;`**end.**

В данном примере в протокол будет внесен текст "virus.exe"

15.32.7 function ExtractFilePath**function** ExtractFilePath (AName : **string**) : **string**;

Извлекает путь к файлу из полного имени файла. Например, если передать функции полное имя файла "c:\virus\file.exe", то функция вернет "c:\virus\"

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

Пример:**begin**`AddToLog (ExtractFilePath ('c:\windows\virus.exe')) ;`**end.**

В данном примере в протокол будет внесен текст "c:\windows\"

15.32.8 function ExtractFileExt**function** ExtractFileExt (AName : **string**) : **string**;

Извлекает путь к файлу из полного имени файла. Например, если передать функции полное имя файла "c:\virus\file.exe", то функция вернет ".exe". Обратите внимание, что расширение возвращается вместе с точкой перед ним.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

Пример:**begin**`AddToLog (ExtractFileExt ('c:\windows\virus.exe')) ;`**end.****15.32.9 function DirectoryExists****function** DirectoryExists (ADirName : **string**) : **boolean**;

Возвращает true, если на диске существует указанный каталог. Переданное имя каталога автоматически нормализуется, поэтому наличие/отсутствие слеша в конце пути, наличие пробелов в начале и конце имени каталога не влияет на работу функции.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

15.32.1procedure CreateDirectory

```
procedure CreateDirectory(ADirName : string);
```

Создает каталог с указанным именем на диске. Переданное в качестве параметра имя каталога автоматически нормализуется.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

15.32.1procedure DeleteDirectory

```
procedure DeleteDirectory(ADirName : string);
```

Удаляет пустой каталог с указанным именем. Переданное в качестве параметра имя каталога автоматически нормализуется.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

Отсутствие каталога или невозможность его удаления не является ошибкой, успешность операции можно проверить при помощи функции [DirectoryExists](#)^[146]. Если в каталоге есть вложенные каталоги или файлы, то его удаление при помощи DeleteDirectory невозможно - следует предварительно вложенные файлы и папки. Для рекурсивного удаления файлов и папок в данном случае можно применить функцию [DeleteFileMask](#)^[148]

15.32.1function FileExists

```
function FileExists(AFileName : string) : boolean;
```

Функция возвращает TRUE, если на диске найден указанный файл.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

15.32.1function GetFileSize

```
function GetFileSize(AFileName : string) : integer;
```

Возвращает размер файла с указанным именем в байтах. В случае ошибки (файл не найден, ошибка доступа) возвращается размер "-1".

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

Пример:

```
begin
```

```
  AddToLog(IntToStr(GetFileSize('%System32%\notepad.exe')));  
end.
```

15.32.1function CalkFileMD5

```
function CalkFileMD5 (AFileName : string) : string;
```

Возвращает MD5 сумму указанного файла или пустую строку в случае, если файл не существует или доступ к нему заблокирован.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

Совместимость: AVZ 4.28 и выше

Пример:

```
begin
```

```
AddToLog (CalkFileMD5 ('%System32%\notepad.exe')) ;  
end.
```

15.32.1function DeleteFile

```
function DeleteFile (AFileName : string) : boolean;
```

Удаляет указанный файл. Возвращает TRUE в случае успешного удаления и FALSE в случае ошибки.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

Указание несуществующего файла или пустой строки не является ошибкой.

В случае ошибки удаления файла обычным удалением променяется отложенное удаление. Имена удаляемых файлов автоматически добавляются в список файлов, который используется функцией [ExecuteSysClean](#)^[129].

Пример:

```
begin
```

```
DeleteFile ('%WinDir%\trojan.exe') ;  
end.
```

15.32.1function DeleteFileMask

```
function DeleteFileMask (ADir, AMask : string; ARecurse : boolean) : boolean;
```

Выполняет поиск и удаление файлов в папке ADir, имена которых соответствуют маске AMask. Параметр ARecurse управляет обработкой вложенных каталогов - если ARecurse = true, то будет выполнен рекурсивный обход папок начиная от заданной и в каждой из папок будет выполнен поиск и удаление файлов по маске AMask. После завершения обработки папки в данном случае производится проверка, остались ли в ней файлы - если файлов не осталось, то папка автоматически удаляется. Если ARecurse = false, то поиск и удаление файлов производится только в заданной папке.

В параметре AMask можно указать одну или несколько масок. В случае задания нескольких масок они разделяются пробелами, запятыми или точкой с запятой. Т.е. данный варианты задания маски являются эквивалентными:

```
*.exe;*.dll;*.com  
*.exe,*.dll,*.com  
*.exe *.dll *.com
```

На заметку:

Данная функция опасна и может серьезно повредить системе в случае задания некорректных параметров. Поэтому перед применением функции рекомендуется тщательно проверить ее аргументы.

Совместимость: AVZ 4.28 и выше

Пример:

```
begin  
  // Очистка папки Temp  
  DeleteFileMask('%Tmp%', '*.*', true);  
end.
```

15.32.1function CopyFile

```
function CopyFile(AFromName, AToName : string) : boolean;
```

Выполняет копирование файла с именем AFromName в файл AToName. Возвращает true, если копирование было успешным и false в случае ошибки.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

15.32.1function RenameFile

```
function RenameFile(AFromName, AToName : string) : boolean;
```

Выполняет переименование файла с именем AFromName в AToName. Возвращает true в случае успешного выполнения переименование и false в случае ошибки.

В имени файла допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

```
begin  
  RenameFile('%WinDir%\trojan.exe', '%WinDir%\trojan.bak');  
end.
```

15.32.1function GetDriveType

```
function GetDriveType(ADirName : string) : integer;
```

Определяет тип диска. Параметр ADirName должен содержать путь к корневому каталогу тома, например "C:\".

Возвращаемые значения:

0 - Ошибка, определить тип диска невозможно

1 - Указанный диск не существует (как вариант - некорректно путь к корневому каталогу тому, например "C:\Windows")

2 - Сменный диск
3 - HDD
4 - Сетевой диск
5 - CD-ROM/DVD-ROM
6 - RAM-диск

Совместимость: AVZ 4.26 и выше

Пример:

begin

```
AddToLog (Inttostr (GetDriveType ('C:\')) );
```

end.

15.32.2(Макросы, допустимые в именах файлов

%WinDir% - путь к папке Windows (без \ на конце)
%SystemRoot%, %System32% - заменяется на путь к папке System/System32 (без \ на конце)
%SysDisk% - системный диск в виде "X:" (без \)
%Tmp% - путь к системной папке для временных файлов
%PF% - путь к папке Program Files на системном диске (без \ на конце)

Начиная с версии 4.26 поддерживаются

%SYSTEMDRIVE% - синоним %SysDisk%, системный диск в виде "X:"
%Personal% - путь к папке "Мои документы" текущего пользователя
%ProfileDir% - путь к папке, в которой размещаются профили пользователей
%USERPROFILE% - путь к папке с профилем текущего пользователя (берется из одноименной переменной окружения)

На заметку:

Регистр символов макроса не учитывается, к примеру %SysDisk%, %SYSDISK% и %sysdisk% эквивалентны.

См. также [GetEnvironmentVariable](#)^[107]

15.33 Работа с текстовыми файлами и списками строк

15.33.1 Класс TStringList

Для работы с текстовыми файлами и массивами строк в скриптовом языке поддерживается класс TStringList, идентичный аналогичному классу в Delphi. Данный класс позволяет:

- Создавать списки строк и осуществлять их обработку;
- Производить поиск строк и сортировку;
- Работать с текстовыми файлами;
- Осуществлять особую обработку массивов строк вида "имя=значение";

Методы:

constructor Create;

Создает экземпляр класса. Попытка работы с экземпляром класса до его создания или после разрушения приведет к ошибке.

destructor Free;

Разрушает экземпляр класса, освобождает все занятые им ресурсы.

function Add(**const** S: **string**): Integer;

Добавляет строку к списку, возвращаемое значение - позиция элемента в списке (позиция отсчитывается от 0).

procedure Append(**const** S: **string**);

Аналогично Add, но не возвращает позицию добавленного элемента

procedure Clear;

Очищает список строк

procedure Delete(Index: Integer);

Удаляет строку с указанным индексом. Указание недопустимого индекса является ошибкой

procedure Move(CurIndex, NewIndex: Integer); **virtual**;

Перемещает элемент CurIndex в позицию NewIndex. Указание недопустимого индекса является ошибкой

function IndexOf(**const** S: **string**): Integer;

Поиск указанной строки в массиве. Если строка найдена, то возвращается ее индекс, если не найдена - значение -1.

procedure Insert(Index: Integer; **const** S: **string**);

Вставляет строку по указанному индексу (раздвигая массив). Например, Insert(0, 'First string !') вставит текст 'First string !' в начало массива строк

procedure LoadFromFile(**const** FileName: **string**);

Загружает текстовый файл с именем FileName в список строк. Имеющиеся при этом в списке данные затираются.

procedure SaveToFile(**const** FileName: **string**);

Загружает список строк в текстовый файл с именем FileName. Имеющиеся при этом в списке данные не изменяются.

Свойства

property Count: Integer;

Возвращает количество строк в списке. Допускает только чтение.

15.33.2 Примеры

Минимальный скрипт, использующий списки

```
var
  SL : TStringList;
begin
  SL := TStringList.Create;
  SL.Free;
end.
```

Вывод содержимого списка в протокол

```
// Вывод списка строк в протокол
procedure PrintStringListToLog(ASL : TStringList);
var
    i : integer;
begin
    for i := 0 to ASL.Count - 1 do
        AddToLog(ASL[i]);
    end;

var
    SL : TStringList;
begin
    SL := TStringList.Create;
    SL.Add('111111');
    SL.Add('222222');
    SL.Add('333333');
    PrintStringListToLog(SL);
    SL.Free;
end.
```

Добавление, удаление и перемещение строк

```
var
    SL : TStringList;
begin
    SL := TStringList.Create;
    // Добавление строк
    SL.Add('111111');
    SL.Add('222222');
    SL.Add('333333');
    // Удаление второй строки
    SL.Delete(1);
    // Перестановка строк
    SL.Move(0, 1);
    // Вывод результатов
    PrintStringListToLog(SL);
    SL.Free;
end.
```

Добавление, удаление и перемещение строк

```
var
    SL : TStringList;
begin
    SL := TStringList.Create;
    // Добавление строк
    SL.Add('111111');
    SL.Add('222222');
```

```
SL.Add('333333');  
// Удаление второй строки  
SL.Delete(1);  
// Перестановка строк  
SL.Move(0,1);  
// Вывод результатов  
PrintStringListToLog(SL);  
SL.Free;  
end.
```

Поиск строк

```
var  
  SL : TStringList;  
begin  
  SL := TStringList.Create;  
  // Добавление строк  
  SL.Add('111111');  
  SL.Add('222222');  
  SL.Add('333333');  
  // Удаление второй строки  
  AddToLog(IntToStr(SL.IndexOf('111')));  
  AddToLog(IntToStr(SL.IndexOf('111111')));  
  AddToLog(IntToStr(SL.IndexOf('333333')));  
  SL.Free;  
end.
```

В данном примере первая операция поиска не найдет образец, и функция вернет -1. Поиск "333333" и "111111" будет успешным и метод вернет их индексы.

Работа с текстовыми файлами

```
var  
  SL : TStringList;  
begin  
  SL := TStringList.Create;  
  // Добавление строк  
  SL.Add('111111');  
  SL.Add('222222');  
  SL.Add('333333');  
  // Сохранение в текстовый файл  
  SL.SaveToFile(GetAVZDirectory + '1.txt');  
  // Загрузка из текстового файла  
  SL.LoadFromFile(GetAVZDirectory + '1.txt');  
  SL.Free;  
end.
```

15.34 Поиск файла и папок

15.34.1 Класс TFileSearch

Для организации поиска файлов и папок на диске в скриптовом языке предусмотрен класс TFileSearch, реализующий всю необходимую для поиска функциональность.

Методы:

constructor Create(AOwner: TComponent);

Создает экземпляр класса. Единственный параметр AOwner задает владельца в данном случае не имеет смысла и должен быть равен nil

destructor Free;

Разрушает экземпляр класса, освобождает все занятые им ресурсы.

function FindFirst(AFileMask : string) : boolean;

Поиск первого файла или папки, удовлетворяющего условию поиска. Возвращает TRUE, если первый подходящий объект успешной найден. AFileMask задает маску поиска файлов, например "*. *". В маске допустимо указывать макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

function FindNext : boolean;

Поиск очередного файла или папки, удовлетворяющего условию поиска. Возвращает TRUE, если очередной подходящий объект успешной найден. Результат последней операции поиска можно так-же получить при помощи свойства Found

function FindClose : boolean;

Завершает поиск, освобождает занятые для поиска ресурсы, но не разрушает экземпляр класса. FindClose автоматически вызывается при вызове FindFirst и Free

Свойства:

property Found : boolean;

Результат последней операции FindFirst или FindNext. Наличие этого свойства упрощает организацию цикла поиска

property ISDir : boolean;

Возвращает TRUE, если текущий найденный объект является каталогом

property FileName : string;

Возвращает имя файла текущего найденного объекта (важно отметить - только имя, без пути !!)

property FileAttr : byte;

Возвращает атрибуты последнего найденного объекта

property FileTime : TDateTime;

Возвращает дату и время создания последнего найденного объекта

15.34.2 Примеры поиска

Возможно два базовых варианта организации цикла поиска файлов. Рассмотрим оба (они равнозначны по скорости работы)

Вариант 1. Цикл repeat ... until

```
var
  FS : TFileSearch;
begin
  FS := TFileSearch.Create(nil);
  if FS.FindFirst('c:\*.*') then
    repeat
      if FS.IsDir then
        AddToLog('Найден каталог: '+FS.FileName)
      else
        AddToLog('Найден файл: '+FS.FileName)
      until not(FS.FindNext);
      FS.Free;
    end.
```

Вариант 2. Цикл while

```
var
  FS : TFileSearch;
begin
  FS := TFileSearch.Create(nil);
  FS.FindFirst('c:\*.*');
  while FS.Found do begin
    if FS.IsDir then
      AddToLog('Найден каталог: '+FS.FileName)
    else
      AddToLog('Найден файл: '+FS.FileName);
    FS.FindNext;
  end;
  FS.Free;
end.
```

Пример рекурсивного обхода каталогов

```
// Обработка найденного файла
Procedure ScanFile(AFileName : string);
begin
  SetStatusBarText(AFileName);
end;

// Сканирование каталога
Procedure ScanDir(ADirName : string; AScanSubDir : boolean);
var
```

```

    FS : TFileSearch;
begin
    ADirName := NormalDir(ADirName);
    FS := TFileSearch.Create(nil);
    FS.FindFirst(ADirName + '.*.*');
    while FS.Found do begin
        if FS.IsDir and AScanSubDir and (FS.FileName <> '.') and
        (FS.FileName <> '..') then
            ScanDir(ADirName + FS.FileName, AScanSubDir)
        else
            ScanFile(ADirName + FS.FileName);
        FS.FindNext;
    end;
    FS.Free;
end;

begin
    ScanDir('c:\', true);
end.

```

15.35 Работа с процессами

15.35.1 function ExecuteFile

```

function ExecuteFile(FileName, Params : string;
                      Mode : integer; WaitTime : integer;
                      Terminate : boolean) : boolean;

```

Производит выполнение указанного файла.

FileName - имя файла. В имени файла допустимы макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150].

Params - параметры. При отсутствии параметров передается пустая строка. Параметры так-же допустимо передавать после имени запускаемого файла, отделяя их запятой

Mode - код режима отображения запускаемой программы. Поддерживаются коды:

0 - запуск приложения с флагом SW_HIDE - при этом окно приложения невидимо для пользователя

1 - стандартное отображение окна запускаемого приложения

2 - окно запускаемого приложения минимизировано

3 - окно запускаемого приложения развернуто на весь экран

WaitTime - время в миллисекундах, в течении которого производится ожидание завершения запущенной программы. Указание нулевого времени приводит к тому, что скрипт продолжает свое выполнение без ожидания завершения запущенной программы. Минимальная задержка - 100 мс, указание меньшей задержки приводит к тому, что она будет установлена на 100 мс. Ожидание завершается по достижению одного из двух событий - исчезению указанного ремени или завершению запущенной программы.

Terminate - принудительное завершение запущенной программы после истечения заданного в WaitTime периода. При WaitTime=0 данный параметр не имеет смысла и игнорируется. Логика работы: если прошло время WaitTime, Terminate=true и приложение еще работает, то оно

принудительно завершается.

На заметку:

Если на момент запуска процесса работает AVZGuard, то процесс будет запущен как доверенный процесс и действие AVZGuard на него не будет распространяться.

Примеры:

begin

```
ExecuteFile('notepad.exe', '', 1, 10000, true);
```

end.

15.35.2 function RefreshProcessList

function RefreshProcessList:boolean;

Обновляет список процессов. Эта функция должна вызываться перед началом работы со списком процессов. Функция возвращает True, если построение списка было успешным

15.35.3 function GetProcessCount

function GetProcessCount:integer;

Возвращает количество процессов в списке, построенном при помощи RefreshProcessList. Вызов GetProcessCount до успешного вызова RefreshProcessList не является ошибкой - в этом случае возвращается 0.

15.35.4 function GetProcessName

function GetProcessName(Indx : integer):string;

Возвращает имя исполняемого файла процесса с индексом Indx. Обратите внимание - индекс отсчитывается от нуля. Указание индекса вне допустимого диапазона не является ошибкой - в этом случае функция возвращает пустую строку.

15.35.5 function GetProcessPID

function GetProcessPID(Indx : integer):dword;

Возвращает PID процесса с индексом Indx. Обратите внимание - индекс отсчитывается от нуля. Указание индекса вне допустимого диапазона не является ошибкой - в этом случае функция возвращает пустую строку.

15.35.6 function TerminateProcess

function TerminateProcess(PID : dword):boolean;

Завершает процесс с указанным PID. Функция возвращает TRUE, если процесс с заданным PID найден и успешно завершён.

15.35.7 function TerminateProcessByName

```
function TerminateProcessByName (AName : string) : boolean;
```

Останавливает все процессы, имя которых содержит указанный в AName образец. Применять данную функцию необходимо с разумной осторожностью, так как задание образца ".exe" приведет к остановке всех процессов и краху системы.

```
begin
```

```
  TerminateProcessByName ('iexplore.exe');
```

```
end.
```

15.35.8 Пример

```
var
```

```
  i : integer;
```

```
begin
```

```
  // Обновление списка процессов
```

```
  RefreshProcessList;
```

```
  AddToLog ('Количество процессов = '
```

```
+IntToStr(GetProcessCount));
```

```
  // Цикл анализа полученного списка
```

```
  for i := 0 to GetProcessCount - 1 do begin
```

```
    // Вывод PID процесса и его имя
```

```
    AddToLog (IntToStr(GetProcessPID(i)) + ' ' +
```

```
GetProcessName(i));
```

```
    // Остановка процесса по условию
```

```
    if pos('trojan.exe', LowerCase(GetProcessName(i))) > 0 then
```

```
      TerminateProcess (GetProcessPID(i));
```

```
  end;
```

```
end.
```

Данный пример демонстрирует использование всех описанных выше функций для работы с процессам. Сначала при помощи функции [RefreshProcessList](#)^[157] производится обновление списка процессов. Затем производится просмотр списка в цикле с выводом информации в протокол.

Кроме того, производится завершение процессов, имя исполняемого файла которых содержит текст "trojan.exe"

15.36 Сигнатурный анализатор файла

15.36.1 function LoadFileToBuffer

```
function LoadFileToBuffer (AFile : string) : boolean;
```

Производит загрузку файла с указанным именем в буфер для последующего изучения. В случае успешного выполнения операции возвращается true, в случае ошибки - false. Повторные вызовы не являются ошибкой - перед выполнением операции выделенный при предыдущем вызове буфер автоматически очищается. В имени файла допустимы макросы, подробнее см. в разделе ["макросы, допустимые в именах файлов"](#)^[158]

См. также: [LoadFileToBufferEx](#)^[159], [FreeBuffer](#)^[159], [GetBufferSize](#)^[159], [GetBufferByte](#)^[159], [GetBufferWord](#)^[160], [GetBufferDWord](#)^[160], [GetBufferStr](#)^[160], [SearchSign](#)^[160]

15.36.2 function LoadFileToBufferEx

```
function LoadFileToBuffer(AFile : string; ARel : integer;  
ALen : integer) : boolean;
```

Производит загрузку файла с указанным именем в буфер для последующего изучения. Функция аналогична [LoadFileToBuffer](#)^[158], но позволяет осуществить частичную загрузку файла. В имени файла AFile допустимы макросы, подробнее см. в разделе "[макросы, допустимые в именах файлов](#)"^[150]

В случае успешного выполнения операции возвращается true, в случае ошибки - false. Повторные вызовы не являются ошибкой - перед выполнением операции выделенный при предыдущем вызове буфер автоматически очищается.

ARel - смещение в файле. Если ARel равно 0, то загрузка идет с начала файла. Отрицательное значение ARel рассматривается как смещение от конца файла. Указание отрицательного смещения больше размера файла приведет к тому, что файл будет загружаться с начала. Указание положительного смещения больше размера файла является ошибкой и файл при этом не загружается.

ALen - размер загружаемого буфера. Указание нулевого размера трактуется как "загрузить от указанного смещения до конца файла". Указание размера, при котором точка ARel + ALen выйдет за границу файла не является ошибкой - значение ARel автоматически исправляется таким образом, что идет загрузка от ARel и до конца файла.

См. также: [LoadFileToBuffer](#)^[158], [FreeBuffer](#)^[159], [GetBufferSize](#)^[159], [GetBufferByte](#)^[159], [GetBufferWord](#)^[160], [GetBufferDWord](#)^[160], [GetBufferStr](#)^[160], [SearchSign](#)^[160]

15.36.3 function FreeBuffer

```
function FreeBuffer : boolean;
```

Выполняет освобождение памяти, выделенной в ходе успешного вызова LoadFileToBuffer. Повторный вызов не является ошибкой - при отсутствии буфера в памяти функция считает, что очистка прошла успешно.

См. также: [LoadFileToBuffer](#)^[158], [GetBufferSize](#)^[159]

15.36.4 function GetBufferSize

```
function GetBufferSize : integer;
```

Возвращает размер буфера в байтах. Может применяться в случае организации нестандартных алгоритмов анализа буфера в цикле. В случае буфера возвращается значение 0.

См. также: [LoadFileToBuffer](#)^[158], [FreeBuffer](#)^[159]

15.36.5 function GetBufferByte

```
function GetBufferByte(ARel : integer) : byte;
```

Возвращает байт по указанному смещению. Задание недопустимого смещения не приведет к ошибке, но функция вернет значение 0.

См. также: [LoadFileToBuffer](#)^[158], [GetBufferSize](#)^[159]

15.36.6 function GetBufferWord

```
function GetBufferWord(ARel : integer) : word;
```

Возвращает word (двух-байтное целое) по указанному смещению. Задание недопустимого смещения не приведет к ошибке, но функция вернет значение 0.

См. также: [LoadFileToBuffer](#)^[158], [GetBufferSize](#)^[159]

15.36.7 function GetBufferDWord

```
function GetBufferDWord(ARel : integer) : dword;
```

Возвращает dword (четырёх-байтное целое) по указанному смещению. Задание недопустимого смещения не приведет к ошибке, но функция вернет значение 0.

См. также: [LoadFileToBuffer](#)^[158], [GetBufferSize](#)^[159]

15.36.8 function GetBufferStr

```
function GetBufferStr(ARel : integer; ALen : integer) :  
string;
```

Возвращает строку символов начиная от заданного смещения ARel. Отрицательные смещения рассматриваются как смещения относительно конца буфера. Длина строки задается параметром ALen, но реальная длина определяется автоматически исходя из следующих критериев:

- В случае выхода точки ARel + ALen за пределы буфера длина ALen автоматически уменьшается таким образом, чтобы копировались данные от указанной позиции и до конца буфера
- В случае выхода точки ARel за левую границу буфера берется ARel = 0, за правую - возвращается пустая строка
- При обнаружении в строке символа с кодом 0 строка усекается до этой позиции

См. также: [LoadFileToBuffer](#)^[158], [FreeBuffer](#)^[159], [GetBufferSize](#)^[159]

15.36.9 function SearchSign

```
function SearchSign(ASign : string; ARel : integer; ALen :  
integer) : integer;
```

Основная функция анализатора. Производит поиск сигнатуры ASign начиная со смещения ARel в зоне длиной ALen.

ARel - смещение точки, с которой начинается поиск. Смещение отсчитывается с нуля, указание отрицательного смещения рассматривается как смещение от конца файла

ALen - длина. Указание нулевой длины рассматривается как поиск от точки ARel до конца

буфера

ASign - сигнатура. Сигнатура задается текстовой строкой, описывающей правила сигнатурного поиска. Сигнатура состоит из элементов, разделенных пробелами. Пробелы в начале и конце сигнатуры игнорируются.

Поддерживаются следующие типы элементов сигнатуры:

xx - байт должен быть равен xx, где xx - значение байта в шестнадцатеричном виде
Пример: "55 AA 45 21"

!xx - байт не равен xx, где xx - значение байта в шестнадцатеричном виде
Пример: "55 AA !45 21"

? - байт может иметь любое значение (по сути это пропуск байта при анализе). Указание данного элемента в начале и конце сигнатуры не имеет смысла, хотя не является ошибкой
Пример: "55 ? AA ? ? 45 21"

?nn - пропуск при анализе nn байт (по сути условие - nn байт имеют произвольное значение), где nn - количество пропускаемых байт в десятичном виде. Пример: 55 AA ?5 45 21 - данная сигнатура аналогична сигнатуре 55 AA ? ? ? ? 45 21

***xx** - пропуск нескольких байт (от нуля до достижения границы анализируемой области) до обнаружения байта, равного xx.

~xx,yy - производит проверку условия <байт буфера> AND yy = xx. Данная операция позволяет сравнивать заданные биты в байте, а не весь байт целиком. Пробелы в данной конструкции (до и после запятой) недопустимы, т.к. пробел является разделителем элементов в сигнатуре
Пример: "55 ~0A,0F" - в данном случае расшифровка сигнатуры звучит как "первый байт равен 55, а младшая тетрада второго равна 0A"

См. также: [LoadFileToBuffer](#)^[158], [FreeBuffer](#)^[159], [GetBufferSize](#)^[159]

15.36.1(Пример реализации сигнатурного искателя

```
// Добавление сообщения в протокол
Procedure AddAlarm(AFileName, AMsg : string);
begin
  AddtoLog('>>>>> '+AFileName+' подозрение на '+AMsg);
end;

// Сканирование файла
Procedure ScanFile(AFileName : string);
begin
  SetStatusBarText(AFileName);
  LoadFileToBuffer(AFileName);
  if SearchSign('2E 61 64 2D 77 2D 61 2D 72 2D 65 2E 63 6F',-
20000, 0) >= 0 then
    AddAlarm(AFileName, 'Adware.Look2me');
    FreeBuffer;
end;

// Сканирование папки (с рекурсивным обходом)
```

```
Procedure ScanDir(ADirName : string; AScanSubDir : boolean);  
var  
    FS : TFileSearch;  
begin  
    ADirName := NormalDir(ADirName);  
    FS := TFileSearch.Create(nil);  
    FS.FindFirst(ADirName + '*.');  
    while FS.Found do begin  
        if FS.IsDir then begin  
            if AScanSubDir and (FS.FileName <> '.') and (FS.FileName  
<> '..') then  
                ScanDir(ADirName + FS.FileName, AScanSubDir)  
            end else  
                ScanFile(ADirName + FS.FileName);  
            FS.FindNext;  
        end;  
        FS.Free;  
    end;  
  
begin  
    ScanDir('c:\', true);  
end.
```

15.37 Работа с результатами исследования системы

Исследование системы собирает различную информацию об изучаемом компьютере и сохраняет ее в двух форматах - HTML (с интерактивными элементами) и XML (для автоматизированного анализа). Для решения ряда задач может возникнуть потребность в анализе собранных данных и выполнении некоторых действий на основании подобного анализа. Для решения этой задачи в скриптовой язык AVZ введен набор команд, позволяющих загрузить XML файл с данными последнего исследования из памяти или любой произвольный XML с диска для последующей обработки.

15.37.1 function SC_INIT

```
function SC_INIT(AFileName : string = '') : boolean;
```

Выполняет инициализацию системы обработки результатов исследования. В случае успешного выполнения возвращает TRUE, в случае ошибки - FALSE. Перед вызовом SC_INIT требуется выполнить исследование системы. Если в рамках скрипта имеется несколько вызовов SC_INIT, то в ходе работы данной функции будет при необходимости автоматически вызвана SC_FREE. Параметр AFileName является необязательным и по умолчанию отсутствует. Его указание задает имя XML файла, который должен быть загружен для анализа - применение данного параметра позволяет анализировать любой протокол исследования системы, а не только последний лог исследования, полученный на данном компьютере.

На заметку: Первичное назначение системы - анализ XML логов исследования системы при помощи скриптов. Однако теоретически данный набор функций позволяет скрипту загрузить и обработать любой XML файл.

Совместимость: AVZ 4.28 и выше

Пример:

```
var
  Res : boolean;
begin
  Res := SC_INIT('C:\avz4\avz_sysinfo.xml');
  if Res then
    AddToLog('Загрузка и парсинг протокола выполнены успешно')
  else
    AddToLog('Ошибка загрузки и парсинга протокола');
end.
```

В данном примере производится загрузка существующего протокола исследования системы с жестко заданным именем, результативность загрузки отображается в протоколе.

```
var
  Res : boolean;
begin
  // Исследование системы
  ExecuteSysCheck('test.htm');
  // Загрузка и анализ результатов исследования
  Res := SC_INIT;
  if Res then
    AddToLog('Парсинг протокола выполнен успешно')
  else
    AddToLog('Ошибка парсинга протокола');
end.
```

В данном примере выполняется исследование системы и затем собранные результаты загружаются в парсер для анализа, результативность загрузки отображается в протоколе.

15.37.2 function SC_FREE

```
function SC_FREE : boolean;
```

Освобождает ресурсы, выделенные в процессе работы SC_INIT. Рекомендуется вызвать данную функцию после завершения работы с данными, собранными в ходе исследования системы.

На заметку: после завершения работы скрипта AVZ в случае необходимости автоматически вызывает SC_FREE.

Совместимость: AVZ 4.28 и выше

Пример:

```
begin
  // Исследование системы
  ExecuteSysCheck('test.htm');
  // Инициализация системы обработки результатов исследования
  SC_INIT;
  ... обработка результатов исследования ...
```

```
// Освобождение ресурсов  
SC_FREE;  
end.
```

15.37.3 function SC_SelectNode

```
function SC_SelectNode(ANode : string) : boolean;
```

Выбирает один из тегов XML с заданным именем. Имя задается в виде "AVZ\имя_узла", имя не чувствительно к регистру. Например, для работы с данными о запущенных процессах требуется задать имя 'AVZ\Process', с расширениями проводника - "AVZ\ExplorerExt".
Функция возвращает TRUE, если успешно удалось найти запрошенные данные. В случае, если запрошенные данные не найдены, не инициализирован анализатор или не выполнялось исследование, то функция возвращает FALSE.

Совместимость: AVZ 4.28 и выше

Пример:

```
var  
  Res : boolean;  
begin  
  Res := SC_INIT(GetAVZDirectory + 'LOG\avz_sysinfo.xml');  
  if Res then  
    AddToLog('Загрузка и парсинг протокола выполнены успешно')  
  else begin  
    AddToLog('Ошибка загрузки и парсинга протокола');  
    exit;  
  end;  
  if SC_SelectNode('AVZ\KERNELOBJ') then  
    AddToLog('Тег KERNELOBJ найден');  
end.
```

15.37.4 function SC_GetItemsCount

```
function SC_GetItemsCount : integer;
```

Возвращает количество вложенных тегов данных в текущем теге XML базы. Если тег не выбран, то функция возвращает 0.

Совместимость: AVZ 4.28 и выше

Пример:

```
var  
  Res : boolean;  
begin  
  Res := SC_INIT(GetAVZDirectory + 'LOG\avz_sysinfo.xml');  
  if Res then  
    AddToLog('Загрузка и парсинг протокола выполнены успешно')  
  else begin  
    AddToLog('Ошибка загрузки и парсинга протокола');  
end.
```

```

    exit;
end;
if SC_SelectNode('AVZ\KERNELOBJ') then
    AddToLog('Теґ KERNELOBJ найден')
else begin
    AddToLog('Теґ KERNELOBJ не найден');
    exit;
end;
// Запрос количества вложенных тегов
AddToLog('SC_GetItemsCount = '+inttostr(SC_GetItemsCount));
end.

```

15.37.5 function SC_GetParamVal

function SC_GetParamVal(AIndx : integer; AName, ADefVal : string) : string

Возвращает значение параметра с именем AName теґа с номером AIndx. Отсчет элементов идет с 0. Указание имени несуществующего параметра или индекса за пределами диапазона не является ошибкой - в этом случае функция возвращает значение ADefVal. Значение ADefVal также возвращается в случае, если у теґа с индексом AIndx нет параметра с именем AName. Для чтения параметров текущего теґа (родительского для теґов данных) необходимо указать индекс равный -1.

Совместимость: AVZ 4.28 и выше

Пример:

```

var
    Res : boolean;
    i      : integer;
begin
    Res := SC_INIT(GetAVZDirectory + 'LOG\avz_sysinfo.xml');
    if Res then
        AddToLog('Загрузка и парсинг протокола выполнены успешно')
    else begin
        AddToLog('Ошибка загрузки и парсинга протокола');
        exit;
    end;
    if SC_SelectNode('AVZ\KERNELOBJ') then
        AddToLog('Теґ KERNELOBJ найден')
    else begin
        AddToLog('Теґ KERNELOBJ не найден');
        exit;
    end;
    // Запрос количества вложенных теґов
    AddToLog('SC_GetItemsCount = '+inttostr(SC_GetItemsCount));
    // Вывод данных в цикле
    for i := 0 to SC_GetItemsCount - 1 do
        AddToLog([''+inttostr(i)+''] File= ''+SC_GetParamVal(i, 'File', '')+'')
    end.

```

В данном примере выполняется загрузка существующего протокола исследования из папки LOG, размещенной в рабочем каталоге AVZ, и вывод в цикле элементов значения параметра File всех тегов, для которых родительским является текущий тег, выбранный при помощи SC_SelectNode.

15.37.6 function SC_GetTagName

function SC_GetTagName (AIndx : integer) : string;

Возвращает имя тега с индексом AIndx. Имя возвращается в верхнем регистре. Индекс -1 применяется для получения имени родительского тега. В случае задания некорректного индекса или в случае возникновения ошибки функция возвращает пустую строку.

Совместимость: AVZ 4.28 и выше

Пример:

```
var
  Res : boolean;
  i    : integer;
begin
  Res := SC_INIT(GetAVZDirectory + 'LOG\avz_sysinfo.xml');
  if Res then
    AddToLog('Загрузка и парсинг протокола выполнены успешно')
  else begin
    AddToLog('Ошибка загрузки и парсинга протокола');
    exit;
  end;
  if SC_SelectNode('AVZ\KERNELOBJ') then
    AddToLog('Тег KERNELOBJ найден')
  else begin
    AddToLog('Тег KERNELOBJ не найден');
    exit;
  end;
  // Запрос количества вложенных тегов
  AddToLog('SC_GetItemsCount = '+inttostr(SC_GetItemsCount));
  for i := 0 to SC_GetItemsCount - 1 do
    if SC_GetTagName(i) = 'ITEM' then
      AddToLog('['+inttostr(i)+'] File= '+SC_GetParamVal(i, 'File', ''));
    end;
  end;
```

15.37.7 function SC_SearchItem

function SC_SearchItem (AName, AVal : string; AFullComp : boolean) : int

Поиск элемента по его значению. Выполняет поиск в текущем теге базы XML, обрабатываются значения параметров с именем AName, AVal - сравнение не чувствительно к регистру. AFullComp - параметр, управляющий режимом сравнения. Если он равен TRUE, то выполняется полное сравнение образца и значения параметра с именем AName, если он равен FALSE, то ищется вхождение AVal в значениях параметра с именем AName.

В случае успешного поиска функция возвращает индекс первой найденной записи, в случае отсутствия совпадений или ошибки - значение "-1".

На заметку: Данная функция предназначена для простейшего поиска. Для более сложного поиска (по нескольким параметрам и т.п.) следует организовать цикл от 0 до SC_GetItemsCount-1 и в нем проверять любые условия, получая значения параметров через SC_GetItem. Однако в случае, если поиск возможен через SC_SearchItem, то следует применять именно его, т.к. поиск через SC_SearchItem работает на порядок быстрее, чем аналогичный ему перебор элементов в цикле.

Совместимость: AVZ 4.28 и выше

Пример:

```
var
  Res : boolean;
  i    : integer;
begin
  Res := SC_INIT(GetAVZDirectory + 'LOG\avz_sysinfo.xml');
  if Res then
    AddToLog('Загрузка и парсинг протокола выполнены успешно')
  else begin
    AddToLog('Ошибка загрузки и парсинга протокола');
    exit;
  end;
  if SC_SelectNode('AVZ\KERNELOBJ') then
    AddToLog('Тег KERNELOBJ найден')
  else begin
    AddToLog('Тег KERNELOBJ не найден');
    exit;
  end;
  // Запрос количества вложенных тегов
  AddToLog('SC_GetItemsCount = '+inttostr(SC_GetItemsCount));
  // Поиск тега
  i := SC_SearchItem('File', 'AnyDVD', false);
  if i >= 0 then
    AddToLog('Найден драйвер AnyDVD - ['+inttostr(i)+'] File= '"+SC_GetP
end.
```

15.38 Функции запроса статистики

15.38.1 function GetSuspCount

```
function GetSuspCount:integer;
```

Возвращает количество подозрительных файлов, обнаруженных в ходе последнего сканирования.

Пример:

```
begin
  // Использовать карантин
  SetupAVZ('UseQuarantine=Y');
```

```
// Запуск сканирования
RunScan;
// Что-то подозрительное ??
if GetSuspCount > 0 then begin
    // Выполнение исследования системы
    ExecuteSysCheck(GetAVZDirectory + 'syscheck.htm');
    // Сохранение архива с файлами карантина
    CreateQuarantineArchive(GetAVZDirectory+'quarantine.zip');
end;
end;
```

15.38.2 function GetDetectedCount

```
function GetDetectedCount:integer;
```

Возвращает количество детектированных вредоносных файлов, обнаруженных в ходе последнего сканирования.

15.39 Обработка параметров командной строки

15.39.1 function GetParamCount

```
function GetParamCount:integer;
```

Возвращает количество параметров, переданных пользователем в командной строке

15.39.2 function GetParamStr

```
function GetParamStr (AParamId:integer) : string;
```

Возвращает параметр с указанным номером или пустую строку, если номер параметра вне диапазона. Параметры нумеруются с 1, запрос параметра 0 возвращает полное имя исполняемого файла AVZ.

15.39.3 function GetParamByName

```
function GetParamByName (AParamName:string) : string;
```

Возвращает значение параметра, найденного по имени. Для использования данной функции необходимо передавать параметры в формате имя=значение. Если параметр не найден, то возвращается пустая строка. Если яреди параметров встречается несколько параметров с одинаковым именем, то функция возвращает значение первого из них.

Совместимость: AVZ 4.26 и выше

Пример:

```
begin
    AddToLog (GetParamByName ('test')) ;
end.
```

На заметку:

Данную функцию удобно применять для создания условных секций кода в скриптах - в этом случае можно не модифицируя код скрипта изменять его поведение.

15.39.4 Пример обработки параметров

Пример скрипта, выводящего в протокол все параметры командной строки AVZ:

```
var
  i : integer;
begin
  AddToLog('ParamCount = ' + IntToStr(GetParamCount));
  for i := 1 to GetParamCount do
    AddToLog(' Param[' + IntToStr(i) + '] = "' + GetParamStr(i)
+ '"');
end.
```

15.40 Работа со строками**15.40.1 function StringReplace**

```
function StringReplace(S, OldPattern, NewPattern: string);
```

Возвращает строку S, в которой все вхождения OldPattern заменены на NewPattern. Сравнение при поиске OldPattern ведется без учета регистра символов.

Пример:

```
begin
  AddToLog(StringReplace('У попа была кошка', 'кошка', 'собака'
));
end.
```

15.40.2 function Pos

```
function Pos(SubStr, S: String): Integer;
```

Возвращает позицию подстроки substr в строке S. Поиск подстроки ведется с учетом регистра символов. Если поиск подстроки успешен, то возвращается номер первого символа первого вхождения SubStr в строке S. Если поиск неуспешен, то функция возвращает 0.

```
begin
  AddToLog(IntToStr(Pos('кошка', 'У попа была собака')));
  AddToLog(IntToStr(Pos('собака', 'У попа была собака')));
end.
```

15.40.3 procedure Insert

```
procedure Insert(S: String; var S2: String; pos: Integer);
```

Вставляет строку S в позицию pos строки S2.

Пример:

```
var  
  S : string;  
begin  
  S := 'У попа собака';  
  Insert('была ', S, 8);  
  AddToLog(S);  
end.
```

15.40.4 procedure Delete

```
procedure Delete(var s: String; from, count: Integer);
```

Удаляет из строки S несколько символов начиная с позиции from, количество удаляемых символов задается параметром count.

```
var  
  S : string;  
begin  
  S := 'У попа была кошка или собака';  
  Delete(S, 12, 10);  
  AddToLog(S);  
end.
```

15.40.5 function Copy

```
function Copy(s: String; from, count: Integer): String;
```

Возвращает фрагмент строки S длиной count символов начиная с позиции from.

Пример:

```
begin  
  AddToLog(Copy('У попа была кошка или собака', 13, 5));  
end.
```

15.40.6 function Length

```
function Length(s: string): Integer;
```

Возвращает длину строки s.

Пример:

```
begin  
  AddToLog(IntToStr(Length('У попа была собака')));
```

end.

15.40.7 function Trim

```
function Trim(s: String): String;
```

Возвращает строку s, в которой удалены все пробелы и спецсимволы в начале и конце.

15.40.8 function UpperCase

```
function UpperCase(s: String): String;
```

Возвращает строку s, приведенную к верхнему регистру.

15.40.9 function LowerCase

```
function LowerCase(s: String): String;
```

Возвращает строку s, приведенную к нижнему регистру.

15.40.10 function Chr

```
function Chr(CharCode: Integer): Char;
```

Возвращает символ с ASCII кодом CharCode

15.40.11 function Ord

```
function Ord(ch: Char): Integer;
```

Возвращает ASCII код заданного символа

15.40.12 Преобразования в строку

15.40.12.1 function IntToStr

```
function IntToStr(i: Integer): String;
```

Преобразует целое число в строку.

15.40.12.2 function FloatToStr

```
function FloatToStr(e: Extended): String;
```

Преобразует число с плавающей точкой в строку.

15.40.12.3 function DateToStr

```
function DateToStr(e: Extended): String;
```

Преобразует дату в строку. При форматировании даты используется стандартная системная форматная маска.

Примеры:

```
begin
  AddToLog ('Дата завершения сканирования: '+DateToStr (Now));
end.

begin

SaveLog (GetAVZDirectory+'\LOG\' +GetComputerName+'_'+DateToStr (
Now)+'_log.txt');
end.
```

15.40.12.function TimeToStr

```
function TimeToStr (e: Extended): String;
```

Преобразует время в строку. При форматировании времени используется стандартная системная форматная маска.

```
begin
  AddToLog ('Время завершения сканирования: '+TimeToStr (Now));
end.
```

15.40.12.function DateTimeToStr

```
function DateTimeToStr (e: Extended): String;
```

Преобразует дату/время в строку. При форматировании даты и времени используется стандартная системная форматная маска.

```
begin
  AddToLog ('Сканирование завершено: '+DateTimeToStr (Now));
end.
```

15.41 Передача оповещения администратору

15.41.1 function SendNetMessage

```
function SendNetMessage (AHost, AFromSt, AToSt, AMessageSt:
string) : boolean;
```

Отправляет сообщение по сети, аналог NET SEND.

Параметры:

- AHost - имя хоста, которому передается сообщение
- AFromSt - поле "От кого"
- AToSt - поле "Кому"
- AMessageSt - текст сообщения.

Значения параметров AFromSt, AToSt, AMessageSt могут содержать русские буквы, но в этом случае не гарантируется корректное отображение сообщения.

```
begin
  // Запуск сканирования
  RunScan;
```

```

// Что-то подозрительное ??
if (GetSuspCount > 0) or (GetDetectedCount > 0) then begin
    SendNetMessage('Monitoring', 'AVZ', 'Admin',
        'Report from computer "' + GetComputerName +
        '" '+
        'SuspCount = ' + InttoStr(GetSuspCount) + ',
        '
        'DetectedCount = ' +
        InttoStr(GetDetectedCount)
        );
    end;
end.

```

15.41.2 function SendSyslogMessage

```

function SendSyslogMessage(AHost, AMessageSt: string) :
boolean;

```

Отправляет сообщение стандартной службе Syslog (порт 514 UDP) на указанном сервере, удобно для регистрации различных событий

```

begin
    // Запуск сканирования
    RunScan;
    // Что-то подозрительное ??
    if (GetSuspCount > 0) or (GetDetectedCount > 0) then begin
        SendSysLogMessage('172.20.97.28',
            'Report from computer "' + GetComputerName +
            '" '+
            'SuspCount = ' + InttoStr(GetSuspCount) + ',
            '
            'DetectedCount = ' +
            InttoStr(GetDetectedCount)
            );
        end;
    end.

```

15.41.3 function SendEmailMessage

```

function SendEmailMessage(AServer, AFrom, ARecipients,
ASubject, AMessageSt: string; AIdentification : boolean;
ALogin, APasswd : string; AAttachFile1, AAttachFile2,
AAttachFile3 : string) : boolean;

```

Данная функция производит отправку письма под управлением скрипта.

Параметры функции:

AServer - имя или IP адрес почтового сервера, через который производится отправка письма

AFrom - значение поля "От кого"

ARecipients - список получателей

ASubject - заголовок письма

AMessageSt - текст сообщения, может содержать символы перевода строки

AIdentification - режим аутентификации. Если передается false, то аутентификация на сервере не производится и значение параметров ALogin и APasswd игнорируются. Если передается true, то аутентификация производится и параметры ALogin и APasswd должны быть заполнены.

ALogin - имя пользователя для авторизации на почтовом сервере

APasswd - пароль пользователя для авторизации на почтовом сервере

AAttachFile1, AAttachFile2, AAttachFile3 - имена файлов, которые должны быть приложены к письму. Передача пустого имени означает, что вложение не требуется.

Пример:

begin

```
// Использовать карантин
SetupAVZ ('UseQuarantine=Y');
// Запуск сканирования
RunScan;
// Что-то подозрительное ??
if (GetSuspCount > 0) or (GetDetectedCount > 0) then begin
    // Выполнение исследования системы
    ExecuteSysCheck(GetAVZDirectory + 'syscheck.htm');
    // Сохранение архива с файлами карантина
    CreateQuarantineArchive(GetAVZDirectory+'quarantine.zip');
    // Отправка письма
    SendEmailMessage('mail.my_mail_server.ru', 'AVZ',
'my_mail@mail.ru',
        'AVZ email alert',
        'Report from computer "' + GetComputerName +
'" '+#13 +
        'SuspCount = ' + InttoStr(GetSuspCount) + #13
+
        'DetectedCount = ' +
InttoStr(GetDetectedCount) ,
        false, '', '',
        GetAVZDirectory + 'syscheck.zip',
        GetAVZDirectory + 'quarantine.zip',
        ''
    );
end;
end.
```

15.42 Типовые примеры

15.42.1 Параметры запуска

1. Запуск AVZ и выполнение скрипта Test1.txt

avz.exe Script=Test1.txt

2. Запуск AVZ в невидимом для пользователя режиме и выполнение скрипта Test2.txt


```
avz.exe HiddenMode[96]=3 Script=Test2.txt
```

3. Запуск AVZ в недоступном для пользователя режиме (отображается ярлык с трее) и выполнение скрипта Test3.txt, режим карантина - сетевой

```
avz.exe nw=Y HiddenMode=2 Script=Test3.txt
```

15.42.2 Сканирование компьютера

```
begin
  // если сканирование не завершилось за 10 минут, прервать его
  ActivateWatchDog(10*60);
  // Настройка AVZ
  SetupAVZ('UseQuarantine=Y'); // Включить карантин
  SetupAVZ('Priority=-1');      // Пониженный приоритет
  // Запуск сканирования
  RunScan;
  // Завершение работы AVZ
  ExitAVZ;
end.
```

Данный пример демонстрирует простейший скрипт, выполняющий настройку AVZ при помощи функции [SetupAVZ](#)^[106] и запуск сканирования при помощи [RunScan](#)^[107].

15.42.3 Блокировка сканирования по условию

```
begin
  // Не проверять компьютеры, имя которых начинается с 'ASU'
  if pos('ASU', GetComputerName) = 1 then
    ExitAVZ;
  // если сканирование не завершилось за 10 минут, прервать его
  ActivateWatchDog(10*60);
  // Настройка AVZ
  SetupAVZ('UseQuarantine=Y'); // Включить карантин
  SetupAVZ('Priority=-1');      // Пониженный приоритет
  // Запуск сканирования
  RunScan;
  // Завершение работы AVZ
  ExitAVZ;
end.
```

В данном примере для блокировки применяется имя компьютера, получаемое при помощи [GetComputerName](#)^[108]

15.42.4 Сканирование и автокарантин

```
begin
  ActivateWatchDog(5*60);
  // Настройка AVZ
  SetupAVZ('UseQuarantine=Y'); // Включить карантин
  SetupAVZ('Priority=-1');      // Пониженный приоритет
  SetupAVZ('nw=Y');            // Сетевой режим
```

```

// Запуск сканирования
RunScan;
// Добавление данных о имени ПК
AddToLog('-----');
AddToLog('Протокол с компьютера '+GetComputerName);
// Сохранение протокола
SaveLog('AVZ_LOG\' +GetComputerName+'_log.txt');
// Автокарантин
ExecuteAutoQuarantine;
// Завершение работы AVZ
ExitAVZ;
end.

```

Для работы данного примера необходимо создать в рабочем каталоге AVZ папку AVZ_LOG. Ключ NW=Y переключает AVZ в сетевой режим работы - это проявляется в том, что при помещении файла в карантин в папке Quarantine и Infected создается новый уровень - имя компьютера. Это позволяет избежать наложения помещаемых в карантин файлов при запуске AVZ на множестве ПК

15.42.5 Лечение заданных папок по условию

```

begin
  ActivateWatchDog(5*60);
  // Настройка AVZ
  SetupAVZ('UseQuarantine=Y'); // Включить карантин
  SetupAVZ('Priority=-1');     // Пониженный приоритет
  if GetComputerName = 'PUPKIN' then begin
    SetupAVZ('Scan=C:\');      // Проверять диск C:\
    SetupAVZ('Scan=D:\Virus'); // Проверять указанную папку
    SetupAVZ('DelVir=Y');      // Включить лечение
  end;
  // Запуск сканирования
  RunScan;
  // Добавление данных о имени ПК
  AddToLog('-----');
  AddToLog('Протокол с компьютера '+GetComputerName);
  // Сохранение протокола
  SaveLog('AVZ_LOG\' +GetComputerName+'_log.txt');
  // Завершение работы AVZ
  ExitAVZ;
end.

```

15.42.6 Автокарантин и сбор подозрительных файлов с ПК

```

begin
  // ***** Настройка AVZ *****
  SetupAVZ('UseQuarantine=Y'); // Включить карантин
  подозрительных
  SetupAVZ('Priority=-1');     // Пониженный приоритет

```

```
SetupAVZ('EvLevel=3');           // Эвристика на максимум
SetupAVZ('ExtEvCheck=Y');        // Расширенный анализ включен
// Активирование сторожевого таймера на 15 минут
ActivateWatchDog(60 * 15);
// Запуск сканирования
RunScan;
// Добавление данных о имени ПК
AddToLog('-----');
AddToLog('Протокол с компьютера '+GetComputerName);
// Автокарантин
ExecuteAutoQuarantine;
// Генерация архива с собранными файлами
CreateQurantineArchive('c:\AVZ_Qurantine.zip');
// Сохранение протокола
SaveLog('c:\AVZ_Qurantine.log');
// Завершение работы AVZ
ExitAVZ;
end.
```

Для запуска скрипта удобно создать BAT файл CollectFiles.bat, содержащий команду следующего вида:

avz.exe HiddenMode=1 Script=CollectFiles.avz

Предполагается, что скрипт сохранен в файле CollectFiles.avz, этот файл вместе с CollectFiles.bat лежат в том-же каталоге, что и avz.exe

15.42.7 Поиск файла на диске

Задача - произвести поиск на диске файла с заданным именем. Данная задача решается при помощи следующего скрипта:

```
// Сканирование папки (с рекурсивным обходом)
Procedure ScanDir(ADirName : string; AScanSubDir : boolean);
var
  FS : TFileSearch;
begin
  ADirName := NormalDir(ADirName);
  FS := TFileSearch.Create(nil);
  FS.FindFirst(ADirName + '.*.*');
  while FS.Found do begin
    SetStatusBarText(ADirName + FS.FileName);
    if FS.IsDir then begin
      if AScanSubDir and (FS.FileName <> '.') and (FS.FileName
<> '..') then
        ScanDir(ADirName + FS.FileName, AScanSubDir)
      end else
        if LowerCase(FS.FileName) = 'trojan.dll' then
          AddToLog('Найден файл '+ADirName + FS.FileName);
        FS.FindNext;
      end;
```

```

    FS.Free;
end;

begin
    ScanDir('c:', true);
end.

```

В данном примере функция ScanDir применяется для рекурсивного обхода каталогов. Функция получает два параметра – имя каталога для сканирования и параметр AScanSubDir, указывающий, следует ли сканировать вложенные каталоги.

15.42.8 Заготовка скрипта для сканирования сети

Заготовка типового скрипта для сканирования множества ПК в сети:

```

begin
    // Проверка - блокировки запуска
    if pos('первые буквы имени', GetComputerName) = 1 then
ExitAVZ;
    if GetComputerName = 'имя1' then ExitAVZ;
    if GetComputerName = 'имя2' then ExitAVZ;
    // Пауза на 50 сек, чтобы не мешать автозагрузке
    Sleep(50);
    // Настройка AVZ
    SetupAVZ('UseQuarantine=Y'); // Включить карантин
    SetupAVZ('nw=Y'); // Сетевой режим карантина
    SetupAVZ('Priority=-1'); // Пониженный приоритет
    SetupAVZ('EvLevel=3'); // Эвристика на максимум
    SetupAVZ('ExtEvCheck=Y'); // Расширенный анализ включен
    SetupAVZ('DelVir=Y'); // Включить лечение
    // Активирование сторожевого таймера на 15 минут
    ActivateWatchDog(60 * 15);
    // Запуск сканирования
    RunScan;
    // Добавление данных о имени ПК
    AddToLog('-----');
    AddToLog('Протокол с компьютера '+GetComputerName);
    AddToLog('Путь к AVZ = ' + GetAVZDirectory);
    // Автокарантин
    ExecuteAutoQuarantine;
    // Сохранение протокола
    SaveLog(GetAVZDirectory+'\LOG\'+GetComputerName+'_ log.txt'
);
    // Завершение работы AVZ
    ExitAVZ;
end.

```

Параметры запуска AVZ в этом случае:

\\my_server\AVZ\lavz.exe Priority=-1 nw=Y nq=Y HiddenMode=2 Script=\\my_server\AVZ\

netscan.avz

при этом предполагается, что AVZ находится в папке \\my_server\AVZ, а скрипт сохранен в файле netscan.avz.

Некоторые комментарии по скрипту и параметрам запуска:

- Priority=-1 - запускает AVZ с пониженным приоритетом, чтобы он не мешал работе компьютера. Этот ключ дублируется в скрипте и командной строке и в принципе не обязателен в случае сканирования современных ПК
- pw=Y - аналогично, дублируется в командной строке и скрипте (дублирование это не обязательно, достаточно в одном месте). Этот ключ переключает карантин в "сетевой режим" - в нем в папке Quarantine появляется еще один уровень - по сетевому имени компьютера
- HiddenMode=2 - запуск AVZ в скрытом режиме - в тее на время сканирования видна иконка AVZ, но доступ пользователя к органам управления AVZ-ом блокирован. Папку Quarantine необходимо предварительно создать и выдать пользователям права на запись в нее
- [SaveLog](#)^[112](GetAVZDirectory+"\LOG\"+GetComputerName+"_ log.txt"); - в данном случае сохранение протокола идет в рабочем каталоге AVZ, причем протокол содержит в имени имя компьютера. Папку LOG необходимо предварительно создать и выдать пользователям права на запись в нее
- Вызов [ExecuteAutoQuarantine](#)^[113] в скрипте для обычного сканирования следует закомментировать - эта функция собирает все файлы, находящиеся в памяти или автозапуске и не опознанные как безопасные и системные. Сбор таких файлов удобен для их последующей проверки несколькими антивирусами
- Проверки вида "if pos('первые буквы имени', [GetComputerName](#)^[108]) = 1 then ExitAVZ;" и "if GetComputerName = 'имя2' then ExitAVZ;" естественно не обязательны и позволяют исключать из сканирования компьютеры с заданными сетевыми именами. Это удобно для отключения ежедневного сканирования компьютеров администраторов сети и IT-специалистов (но при этом AVZ у них находится в автозапуске и в случае чего можно убрать фильтр и включить их ПК в сканирование).

15.42.9 Обновление баз с протоколированием

Задача: выполнить обновление базы в "тихом" режиме (без отображения GUI) с протоколированием успешности операций. Это в частности удобно в случае размещения AVZ на сервере - в этом случае процедуру обновления баз можно включить в планировщик и в результате на сервере будет находиться AVZ с актуальной базой.

Скрипт имеет вид:

```
var
  S : string;
begin
  // Обновление баз
  if ExecuteAVUpdate then S := 'Обновление прошло успешно'
  else S := 'Ошибка обновления баз AVZ';
  // Протоколирование
  AddLineToTxtFile(GetAVZDirectory + 'avz_upd.log',
    DateTimeToStr(Now) + ' ' + S);
  // Завершение работы AVZ
  ExitAVZ;
end.
```

Запуск AVZ в данном случае должен производиться с параметрами:

avz.exe HiddenMode=1 script=update.txt

В данном случае предполагается, что скрипт сохранен в папке AVZ в файле с именем update.txt. Параметр [HiddenMode](#)^[96]=1 предписывает AVZ запускаться свернутым в трей. В качестве усовершенствования в данном скрипте можно применить функцию [ExecuteAVUpdateEx](#)^[117]

15.42.1(Пример удаления троянской программы с известным именем

Постановка задачи:

Администратору известно, что на компьютере пользователя имеется троянская программа и ее исполняемый файл с именем c:\windows\system32\trojan.dll, причем данная программа агрессивно защищается от удаления.

Решение задачи:

```
begin
  // **** Шаг 1. Активируем AVZGuard
  if SetAVZGuardStatus(True) then
    AddToLog('AVZGuard успешно активирован')
  else
    AddToLog('AVZGuard - ошибка активации');
  // **** Шаг 2. Удаляем файл
  DeleteFile('c:\windows\system32\trojan.dll');
  // *** Шаг 3. Чистка ссылок на удаленный файл
  ExecuteSysClean;
  // *** Шаг 4. Сохранение протокола
  SaveLog('del_trojan.log');
  // *** Шаг 5. Перезагрузка системы
  RebootWindows(true);
end.
```

Приведенный выше скрипт может быть усовершенствован - можно задействовать систему [Boot Cleaner](#)^[75], что существенно повысит вероятность удаления файла:

```
begin
  // **** Шаг 1. Активируем AVZGuard
  if SetAVZGuardStatus(True) then
    AddToLog('AVZGuard успешно активирован')
  else
    AddToLog('AVZGuard - ошибка активации');
  // **** Шаг 2. Удаляем файл
  DeleteFile('c:\windows\system32\trojan.dll');
  // **** Шаг 3. Импорт списка удаленных файлов в настройки
  Boot Cleaner
  BC_ImportDeletedList;
  // **** Шаг 4. Чистка ссылок на удаленный файл
  ExecuteSysClean;
  // **** Шаг 5. Сохранение протокола
  SaveLog('del_trojan.log');
  // **** Шаг 6. Активация драйвера Boot Cleaner
  BC_LogFile(GetAVZDirectory + 'boot_clr.log');
  BC_Activate;
  // **** Шаг 7. Перезагрузка системы
```

```
RebootWindows(true);  
end.
```

В описанных примерах работа скрипт завершается командой [RebootWindows](#)^[13], что приводит к автоматической перезагрузке компьютера с активной системой [AVZ Guard](#)^[68].

15.42.1 Поиск подозрительных объектов по именам

Поиск файлов по именам малоэффективен для борьбы с современными вредоносными программами, но он может пригодиться в качестве одной из форм своеобразной эвристики. Достоинство метода - высокая скорость поиска и простота реализации. Рассмотрим несколько вариантов реализации этого поиска при помощи скриптов AVZ

Пример 1. Простейший поиск по именам

```
// Поиск файла с указанным именем  
function CheckByName(Fname : string) : boolean;  
begin  
    Result := FileExists(FName) ;  
    if Result then  
        AddToLog('Файл '+FName+' имеет подозрительное имя');  
    end;  
  
begin  
    // Проверка файлов  
    CheckByName('%WinDir%\cservv32.exe');  
    CheckByName('%WinDir%\services.exe');  
    CheckByName('%WinDir%\lsass.exe');  
end.
```

В данном примере объявляется функция CheckByName, задачей которой является проверка наличия файла с указанным именем на диске. В случае обнаружения подобного файла функция возвращает true и делает отметку в протоколе при помощи функции [AddToLog](#)^[12]. Недостатком функции является то, что список файлов как таковой отсутствует - вместо него идут повторяющиеся вызовы CheckByName. В более сложном примере можно усовершенствовать поиск, разместив список имен в отдельном файле, который будет выступать в роли базы данных.

Пример 2. Поиск по именам с использованием загружаемого списка файлов

```
// Поиск файла с указанным именем  
function CheckByName(Fname : string) : boolean;  
begin  
    Result := FileExists(FName) ;  
    if Result then  
        AddToLog('Файл '+FName+' имеет подозрительное имя');  
    end;  
  
var  
    SuspNames : TStringList; // Список имен подозрительных  
    файлов  
    i : integer;
```

```

begin
  // Проверка файлов по обновляемой базе данных
  if FileExists(GetAVZDirectory + 'files.db') then begin
    SuspNames := TStringList.Create;
    SuspNames.LoadFromFile('files.db');
    AddToLog('База имен загружена - количество записей = '
+inttostr(SuspNames.Count));
    // Цикл поиска
    for i := 0 to SuspNames.Count - 1 do
      CheckByName(SuspNames[i]);
    end else
      AddToLog('Ошибка загрузки списка имен файлов');
  end.

```

В данном примере используется уже знакомая по примеру 1 функция CheckByName, которая вызывается в цикле. Для работы со списком файлов применяется класс [TStringList](#)^[15]. Предполагается, что список файлов хранится в текстовом файле files.db, который размещается в каталоге AVZ (путь к этому каталогу определяется при помощи [GetAVZDirectory](#)^[115]).

Пример 3. Поиск по именам с использованием загружаемого списка файлов и AV баз AVZ

```

// Поиск файла с указанным именем
function CheckByName(Fname : string) : boolean;
var
  S : string;
begin
  Result := FileExists(FName) ;
  if Result then begin
    S := '';
    case CheckFile(FName) of
      -1 : S := ', доступ к файлу блокируется';
      1 : S := ', опознан как Malware ('+GetLastCheckTxt+')';
      2 : S := ', подозревается файловым сканером ('
+GetLastCheckTxt+')';
      3 : exit; // Безопасные файлы игнорируем
    end;
    AddToLog('Файл '+NormalFileName(FName)+' имеет
подозрительное имя'+S);
  end;
end;

var
  SuspNames : TStringList; // Список имен подозрительных
  // файлов
  i : integer;
begin
  // Проверка файлов по обновляемой базе данных
  if FileExists(GetAVZDirectory + 'files.db') then begin
    SuspNames := TStringList.Create;

```



```

SuspNames.LoadFromFile('files.db');
AddToLog('База имен загружена - количество записей = '
+inttostr(SuspNames.Count));
// Цикл поиска
for i := 0 to SuspNames.Count - 1 do
  CheckByName(SuspNames[i]);
end else
  AddToLog('Ошибка загрузки списка имен файлов');
end.

```

В данном примере появляется еще одно усовершенствование - функция CheckByName проверяет обнаруженный по имени файла по базам AVZ. Для этих целей применяется функция [CheckFile](#)^[132] (она поддерживается начиная с AVZ 4.23). Анализ результатов сканирования файла позволяет исключить из протокола безопасные файлы и файлы, прошедшие контроль по каталогу Microsoft. Функция CheckFile приемляется совместно с GetLastCheckTxt для получения имени зловреда в случае его детектирования или подозрения. Еще одной особенностью является применение функции [NormalFileName](#)^[145] для нормализации имени файла (которая, в частности, предполагает замену применяемых в AVZ макросов). Дополнительную информацию о поиске файлов можно посмотреть в разделе ["Поиск файлов и папок"](#)^[154]

15.42.1 Блокировка повторного запуска скрипта в течение дня

В случае применения AVZ для автоматического сканирования компьютеров в сети может возникнуть задача блокировки повторного запуска одного и того-же скрипта в течении дня. Это полезно, например, для блокировки повторных запусков AVZ после каждой перезагрузки компьютера или для однократного обновления баз в течении дня. Решение данной операции достаточно простое и сводится в установки в реестре отметки, содержащей время последнего сканирования.

```

var
  LastScanDate, CurrentDate : string;
begin
  LastScanDate := RegKeyStrParamRead('HKEY_CURRENT_USER', 'Software\AVZ');
  CurrentDate := DateToStr(Now);
  if LastScanDate = CurrentDate then begin
    AddToLog('Блокировка повторного запуска. Скрипт уже запускался '+Last
    ExitAVZ;
    exit;
  end;
  RegKeyStrParamWrite('HKEY_CURRENT_USER', 'Software\AVZ', 'LastScanDate
  // продолжение скрипта ....
end.

```

Принцип работы данного скрипта основан на том, что из реестра считывается параметр с именем 'LastScanDate', хранящийся в ключе 'HKEY_CURRENT_USER\Software\AVZ' (естественно, что ключ и имя параметры можно задать любые). Если считанное значение содержит текущую дату, то работа скрипта прерывается и производится выход из AVZ. В противном случае производится запись в реестр отметки с текущей датой.

15.42.1 Сканирование и отправка результатов по почте

Постановка задачи:

Необходимо выполнить сканирование диска C: компьютера, после чего отправить на указанный

адрес письмо, содержащее протокол сканирования, результаты исследования системы и файлы, помещенные в карантин.

begin

```
// Разрешить использовать карантин
SetupAVZ('UseQuarantine=Y');
// Сканировать диск C
SetupAVZ('Scan=C:\');
// Запуск сканирования
RunScan;
// Выполнение исследования системы
ExecuteSysCheck(GetAVZDirectory + 'syscheck.htm');
// Сохранение архива с файлами карантина
CreateQurantineArchive(GetAVZDirectory+'quarantine.zip');
// Отправка письма
SendMessage('mail.my_mail_server.ru', 'AVZ', 'my_mail@mail.ru',
            'AVZ email alert',
            'Report from computer "' + GetComputerName + '" '+#13
            'SuspCount = ' + InttoStr(GetSuspCount) + #13 +
            'DetectedCount = ' + InttoStr(GetDetectedCount) ,
            false, '', '',
            GetAVZDirectory + 'syscheck.zip',
            GetAVZDirectory + 'quarantine.zip',
            ''
            );
// Выход из AVZ
ExitAVZ;
end.
```

В данном примере исследование системы производится после сканирования, поэтому в протокол исследования включается протокол сканирования.

Index

- * -

*.avz 16

- 2 -

2000 Professional 16

- 5 -

5000 100

- A -

ActivateWatchDog 106
Active Setup 54
AddLineToTxtFile 113, 179
AddToLog 112
AdvWare 14
AdWare 14, 84
AS IS 16
AutoFixSPI 137
AVZ 14, 16
AVZ не может найти файл *.avz 100
avz*.tmp 101
AVZ.SYS 16
AVZGuard 68, 70
AVZGuard и антируткит 70
avzNNNN.dta 34
avzNNNN.ini 34
AVZPM 72, 117, 118

- B -

Backdoor 14, 84
BC_Activate 119
BC_Clear 120
BC_DeActivate 119
BC_ImportALL 124
BC_ImportDeletedList 123
BC_ImportQuarantineList 123
BC_LogFile 120

BHO 134
BHOExists 134
Boot Cleaner 75

- C -

Chr 171
ClearQuarantine 115
CLSID 132, 133
CLSIDExists 132
CLSIDFileExists 132
CLSIDFileName 133
Cookie 42
CopyFile 149
CPL 52
CreateDirectory 147
CreateInfectedArchive 114
CreateQuarantineArchive 113

- D -

DateTimeToStr 172
DateToStr 171
DeleteFile 148
DelSPIByFileName 136
Dialer 14, 85
DirectoryExists 146
DLL 53

- E -

Email-Worm 84
ExecuteAutoQuarantine 113
ExecuteAVUpdate 110, 179
ExecuteAVUpdateEx 111, 179
ExecuteFile 156
ExecuteRepair 130
ExecuteStdScr 131
ExitAVZ 107

- F -

FAQ 99
FileExists 147
FloatToStr 171
FreeBuffer 159

- G -

GetAVZGuardStatus 118
GetAVZPMStatus 118
GetAVZVersion 116
GetAVZVersionTxt 117
GetBufferByte 159
GetBufferDWord 160
GetBufferSize 159
GetBufferStr 160
GetComputerComments 108
GetComputerName 108
GetProcessCount 157
GetProcessName 157
GetProcessPID 157
GetScanPath 116
GetSuspCount 167
GetSystemBootMode 107
GetSystemDisk 144

- H -

Hijacker 85
Hosts 55

- I -

Infected 34, 114
INI 143
INIEraseParam 143
INIEraseSection 143
INISectionExists 143
InputBox 109
IntToStr 171
I-Worm 84

- K -

KernelMode 126
keylogger 99

- L -

Length 170
LoadFileToBuffer 158

LoadFileToBufferEx 159
Local Service Provider 89
LowerCase 171
LSP 89

- M -

MD5 45
MiniLog 29

- N -

Namespace Provider 89
NET SEND 172
Net-Worm 84
NormalFileName 145
NSP 89

- O -

Ord 171

- P -

PID процесса 157
Porn-Dialer 14
PornWare 85
PornWare и Dialer 85
Pos 169

- Q -

QuarantineFile 114

- R -

RefreshProcessList 157
REG_DWORD 139
REG_SZ 139
RenameFile 149
RiskWare 85
Rootkit 87, 99, 126
RunScan 107

- S -

SaveCSVLog 112
SaveLog 112
SendEmailMessage 173
SendNetMessage 172
SendSyslogMessage 173
SetAVZGuardStatus 118
SetStatusBarText 117
SetupAVZ 106
ShowMessage 109
Sleep 106
SPI 89
SPI/LSP 49
Spy 14, 84
Spy и SpyWare 84
SpyWare 14, 84
SSDP 100
StringReplace 169

- T -

Task Scheduler 53
TerminateProcess 157
TerminateProcessByName 158
TFileSearch 154
TimeToStr 172
Transport Service Provider 89
Trim 171
Trojan 84
Trojan.Dialer 14
TrojanDownloader 84
TrojanDropper 84
TrojanSpy 84
TSP 89
TStringList 150, 151

- U -

UpperCase 171
URL сервера
 с которого проводится обновление 111
UserMode 126

- V -

Vista 102

- W -

W9x 16
Windows NT 16
Worm 84

- X -

XP Home Edition 16
XP Professional 16

- Z -

автокарантин 35, 175
Автокарантин и сбор подозрительных файлов 176
адрес прокси-сервера 111
Активные соединения по протоколу TCP/IP 50
архив 113, 114
байт по указанному смещению 159
Блокировка 175
Блокировка повторного запуска 183
блокируется запуск AVZ 103
букву диска 144
в реестре 39
Восстановление системы 59
временный файл 145
Вывод списка строк в протокол 151
Выполнение автоматического карантина 113
Выполнение обновления баз утилиты 111
выполнение указанного файла 156
Выполняет приостановку скрипта 106
главное меню 22
Главное окно программы 22
дата модификации 40
дата создания 40
деинсталляция 16
Деинсталляция и отключение AVZPM 72
декомпилировать 16
диалоговое окно с полем ввода 109
дизассемблировать 16
Диспетчер процессов 45

- Добавление
 - удаление и перемещение строк 151
- Добавление в карантин 36
- Добавление в карантин по списку 36
- добавление в протокол 112
- драйвер 46
- Заблокировать 117
- Завершает процесс 157
- Завершение работы AVZ. 107
- загружена Windows 107
- загрузка файла в буфер 158, 159
- запись 139
- запись параметра 143
- Запуск сканирования 107
- зарегистрированных в качестве обработчиков 54
- защищается от удаления 180
- имя 142
- имя исполняемого файла процесса 157
- имя каталога 145
- имя файла 150, 156
- инсталляции 16
- Инсталляция и включение AVZPM 72
- интерфейс 117
- Искать 39, 40
- исследование 57, 127
- исследование системы 57, 127
- как есть 16
- карантин 34, 113, 114
- каталог 144
- ключ 138, 139
- количество детектированных вредоносных файлов 168
- количество параметров 168
- количество подозрительных файлов 167
- количество процессов 157
- Командная строка 93, 97
- комментарии 105
- копирование файла 149
- Лечение 176
- макрос 150
- Менеджер 54
- Менеджер SPI/LSP 49
- Менеджер Winsock SPI 49
- Менеджер автозапуска 50
- Менеджер апплетов панели управления 52
- Менеджер внедренных dll 53
- Менеджер планировщика заданий 53
- Менеджер протоколов и обработчиков 54
- Менеджер расширений IE 51
- Менеджер расширений проводника 51
- Менеджер расширений системы печати 52
- Менеджер файла Hosts 55
- Модули пространства ядра 48
- монитор 101
- на диске 40
- Назначение 18
- наличие 139
- наличие файла 147
- настройки браузера 100
- невозможен запуск AVZ 103
- нешних скриптов управления 105
- нормализация 145
- нормализованное имя файла 145
- Область поиска 23
- Обновление баз 29
- обновление базы 179
- обновления баз 110
- Обновляет список процессов 157
- Общие ресурсы 55
- Общие ресурсы и сетевые сеансы 55
- Описание компьютера 108
- освобождение памяти 159
- Основные параметры 93
- Открытые порты TCP 50
- Открытые порты UDP 50
- Отложенное удаление 31
- отправка письма под управлением скрипта 173
- отправка результатов по почте 183
- очистка карантина 115
- Ошибка загрузки драйвера 101
- папка 145
- папка AVZ 115
- папок 116
- параметр 139, 168
- параметр с указанным номером 168
- параметры 28, 45, 156
- Параметры запуска 174
- Параметры лечения 28
- Параметры поиска 26
- переименование файла 149
- по условию 176
- поддержка 17
- подозрение на keylogger 99
- подозрение на RootKit 99
- подозрение на вирус 99

- Поиск 39, 40, 181
Поиск Cookie 42
Поиск подозрительных объектов по именам 181
поиск сигнатуры 160
поиск файлов и папок 154
порт 514 UDP 173
Порт TCP 5000 100
порты TCP 50
правила сигнатурного поиска 160
Пример 180
Пример конфигурации 97
Примеры поиска 155
проверка 147
Проверять запущенные процессы 23
прокси 111
пространство ядра 48
Протокол 29
протокол в формате CSV 112
процесс 157
процессов 45
Процессы и библиотеки 45
Пуск 22
путь 115, 145
Разблокировать 117
раздел 142
размер буфера 159
Ревизор 78, 79, 80
реестр 39, 138, 139
режим защиты от сбоев 107
режим защиты от сбоев с поддержкой сети 107
руткит 126
сбор подозрительных файлов 176
сетевое имя компьютера 108
сетевые сеансы 55
Сигнатура 158, 159, 160
система 144
Сканирование 175
Сканирование компьютера 175
Сканирование папки 177
сканирования множества ПК 178
сканируемых 116
Скрипт 105
скрипт управления 105
скрипты 63
служба 46
Создает каталог 147
создание 139
Создание базы 79
сообщение 172
Сохранение протокола 112
Специализированные ключи 96
список 116, 157
список автозапускаемых программ 50
список модулей 54
список модулей расширений системы печати 52
список модулей расширения IE 51
список подключенных к Explorer модулей 51
список сканируемых папок 116
Сравнение диска с базой 80
Стандартные скрипты 63
Стоп 22
сторожевой таймер 106
существует 138, 139
существует указанный каталог 146
текстовое сообщение 109
текущий 144
текущий каталог 144
Техническая поддержка 17
Типы файлов 24
троянской программы 180
удаление 31, 138, 139, 180
Удаление каталога 147
удаление файла 148
Удаляет SPI/LSP провайдер 136
условия поиска 40
Файл 40
Часто задаваемые вопросы 99
чтение 139
чтение параметра 143
Эвристическая проверка системы 23
эвристический анализатор 87

Endnotes 2... (after index)

Back Cover