Competitive Uninstaller

This document includes the following topics:

- About the Competitive Uninstaller
- System requirements
- Using the Competitive Uninstaller
- Working with scripts
- Script command reference

About the Competitive Uninstaller

The Competitive Uninstaller (AV32.exe) is a command-line utility that uses script files to completely remove antivirus and firewall programs and their components that are not related to Symantec antivirus protection. Since legacy products can cause conflicts or false positives during virus scans, you should completely remove these unnecessary programs and their components. The Competitive Uninstaller is provided with this release as an unsupported utility.

The Competitive Uninstaller includes a set of predefined scripts that are configured to remove specific antivirus and firewall programs and their components and update the appropriate registry settings. These scripts are stored in the same directory as the utility. In addition, you can create your own scripts for use by the Competitive Uninstaller to remove unwanted components.

System requirements

The following operating systems are supported:

- Windows 95/98/ME
- Windows NT/2000/XP

Using the Competitive Uninstaller

When the Competitive Uninstaller is invoked, the directory in which the AV32.exe program resides is automatically checked for scripts with the .aut extension. These files typically contain instructions for invoking the uninstallation program that is associated with the software being removed. Various antivirus or firewall components are removed, and registry entries are either reset or removed. Since each script is platform and software specific, each script is compared with the registry to determine whether it matches the platform and any existing antivirus or firewall programs that are installed. When a match is found, the removal instructions in the script are executed.

The Competitive Uninstaller also includes the option to specify the execution of a single script, which eliminates the scanning of all .aut files in the Competitive Uninstaller directory.

Removing multiple antivirus or firewall programs

If more than one legacy antivirus or firewall program is found, the Competitive Uninstaller continues to execute each applicable uninstallation script. However, on Windows NT/2000/XP platforms, if the computer restarts during the

uninstallation process (for example, during MSI program removal), the Competitive Uninstaller must be restarted manually.

Note: The Competitive Uninstaller supports the use of a single file name parameter. If you specify more than one script to be executed, only the first script that is entered as a parameter will be invoked.

Running the Competitive Uninstaller

By default, the Competitive Uninstaller scans the directory in which it resides for scripts with the .aut extension and executes all applicable scripts sequentially.

To run the Competitive Uninstaller

1. Open a command line window.
2. Navigate to the directory in which the Competitive Uninstaller (AV32.exe) resides.
3. At the command line prompt, type the following:
   AV32.exe

Specifying a single script to be executed

You can direct the Competitive Uninstaller to use a single script by specifying the file name as a command line parameter. Specifying a file to be executed eliminates the scan of the directory in which the Competitive Uninstaller resides for additional scripts.

At the command line, you can specify the location of the script to be executed, so it is not necessary to place the file in the same directory as the Competitive Uninstaller.

To specify a single script file to be executed

1. Open a command line window.
2. Navigate to the directory in which the Competitive Uninstaller (AV32.exe) resides.
3. At the command line prompt, type the following, where “filename” includes the path (if the file is not in the same directory as the utility) and the file name of the script to be executed:
   AV32.exe *filename*.aut

Performing silent uninstallation

The Competitive Uninstaller typically runs in silent mode which requires no interaction with the user. Silent mode settings are determined by the specifications of each uninstallation program that is invoked by the Competitive Uninstaller. If the uninstallation program requires parameters, use the AddProgramParameters command to include the values. For more information on the AddProgramParameters command, see the Script command reference.

When the Competitive Uninstaller removes legacy antivirus or firewall components from Windows 2000/XP computers, there may be additional confirmation windows that require the user to decide whether or not to remove components such as .dlls or drivers.

For Windows NT/2000/XP platforms, if the computer restarts during the uninstallation process (for example, during MSI program removal), the Competitive Uninstaller must be restarted manually.

Creating a log file

The Competitive Uninstaller creates a log file (Status.ini) that describes whether the removal of legacy antivirus or firewall components succeeded or failed. The log file is stored in the directory in which the Competitive Uninstaller resides.

Each time that a script is executed, the log file is overwritten. If you are removing multiple legacy programs, only the most recent removal attempt is recorded in the log file. In addition, only one error is reported each time a script is executed.

The log file format is as follows:

```
[RESULT]
RETURN=1 (success) or 0 (failure)
STATUS= (message)
```

The status message provides additional details primarily if the removal has failed.

Working with scripts

The Competitive Uninstaller uses a set of scripts to remove unnecessary antivirus and firewall programs that may conflict with Symantec antivirus software. The Competitive Uninstaller includes a set of predefined scripts as well as support for creating your own customized scripts.

Creating a new script file

Scripts are text files that are saved with the .aut extension. Using a text editor such as Notepad, you can create a custom script to invoke an antivirus or firewall uninstallation program, remove or edit registry entries, and remove legacy components.

Scripts have the following guidelines:

- Platform designation: Each script must include the Platform command as the first command in the file, which specifies the platform on which the script is to be run. For more information on valid parameter values, see the Script command reference.
- File extension: Scripts must be saved using the .aut extension.
- Code comments: Use a semicolon (;). The comment character can be used only at the beginning of a line.
- Extra spaces or empty lines: Do not include.

The standard set of scripts are included with the Competitive Uninstaller in the directory in which the program resides. For new script files, you can either save them in the same directory or specify the path when you use the Competitive Uninstaller to invoke your script individually.

Required restarts during uninstallation

Some uninstallation programs that are invoked by the Competitive Uninstaller require the computer to be restarted and the script to be restarted in order to complete the uninstallation process. If that is the case with the program that you are uninstalling, your script does not need to include the GetProgramName or ClearRegistry commands to be run after the restart. The values that were gathered during the first pass are automatically stored by the Competitive Uninstaller and retrieved during the second pass. To complete the second run, the only requirement is to add the RunRobot command after the RestartComputer or PauseForNotificationOfCompletion commands.

Script command reference

This section contains reference information for each script option that is recognized by the Competitive Uninstaller. Script entries are arranged in alphabetical order.

AddProgramParameters

The AddProgramParameters command sets a member variable and is used to pass parameters to the antivirus or firewall uninstallation program. Only one parameter is supported. Therefore, all parameters should be passed with only one call of this command.

RunRobot uses the result of this command.

Syntax

```
addprogramparameters=parameter
```

Example

```
addprogramparameters=/s /i
```

Parameters

parameter

parameters used by the uninstallation program that is being invoked.

Remarks

Do not separate multiple parameters with delimiters. All parameters should be entered as they are to be used in the command line.

ClearRegistry

The ClearRegistry command removes a registry key or key value.

Warning: This feature automatically removes registry entries without prompting you for confirmation of these changes. There is no validation of the hKey, SubKey or Key parameters. All of the keys in the tree that is below the specified key are also removed.

Syntax

```
clearregistry=hKey, SubKey, Key
```

Example

```
clearregistry=HKEY_CURRENT_USER, SOFTWARE\Microsoft\
Windows NT\CurrentVersion\Windows, run
```

Parameters

hKey

The hKey that is passed into the RegOpenKey.

SubKey

The SubKey that is passed into the RegOpenKey.

Key

The Key that is passed into RegDeleteValue/RegDeleteKey.

GetUninstallProgramName

The GetUninstallProgramName command reads the registry and sets a member variable to the specified key value. This command reads the registry key that contains the full path and file name for the antivirus or firewall uninstallation program that is being invoked.

Syntax

```
getuninstallprogramname=hKey, SubKey, Key
```

Example

```
getuninstallprogramname=HKEY_LOCAL_MACHINE,
SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\IBM AntiVirus,
UninstallString
```

Parameters

hKey

The hKey that is passed into the RegOpenKey.

SubKey

The SubKey that is passed into the RegOpenKey.

Key

The Key that is passed into RegQueryValueEx.

GetUninstallProgramPath

The GetUninstallProgramPath command reads the registry and sets a member variable to the specified key value. This command reads the registry key that contains the path for the antivirus or firewall uninstallation program that is being invoked. This command is called prior to calling SetUninstallProgramName and RunRobot.

Syntax

```
getuninstallprogrampath=hKey, SubKey, Key
```

Example

```
getuninstallprogrampath=HKEY_LOCAL_MACHINE,
SOFTWARE\McAfee\Scan95,Dat
```

Parameters

hKey

The hKey that is passed into the RegOpenKey.

SubKey

The SubKey that is passed into the RegOpenKey.

Key

The Key that is passed into RegQueryValueEx.

GetUninstallScriptPath

The GetUninstallScriptPath command reads the registry and sets a member variable to the specified key value. This command reads a registry key that contains the path for the antivirus or firewall uninstallation script file. This command is called prior to calling SetUninstallScriptName or RunRobot.

Syntax

```
getuninstallscriptpath=hKey, SubKey, Key
```

Example

```
getuninstallscriptpath=HKEY_LOCAL_MACHINE,
SOFTWARE\McAfee\Scan95,Dat
```

Parameters

hKey

The hKey that is passed into the RegOpenKey.

SubKey

The SubKey that is passed into the RegOpenKey.

Key

The Key that is passed into RegQueryValueEx.

GetUninstallShieldProgramName

The GetUninstallShieldProgramName command reads the registry and sets a member variable to the specified key value. This command reads the registry key that contains the full path and file name for the antivirus or firewall uninstallation program. This command is to be used only if the antivirus or firewall uninstallation program internally calls the InstallShield uninstallation program (Uinst.exe). This command opens the antivirus or firewall uninstallation log script and places the -y -a run silent parameters in the command line for the InstallShield uninstallation program. This is currently used for McAfee.

Syntax

```
getuninstallshieldprogramname=hKey, SubKey, Key
```

Example

```
getuninstallshieldprogramname=HKEY_LOCAL_MACHINE,
SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\IBM AntiVirus,
UninstallString
```

Parameters

hKey

The hKey that is passed into the RegOpenKey.

SubKey

The SubKey that is passed into the RegOpenKey.

Key

The Key that is passed into RegQueryValueEx.

PauseForNotificationOfCompletion

The PauseForNotificationOfCompletion command is used to prepare for a restart and second pass on computers that cannot be automatically restarted (Windows NT servers, for example). Information that is needed on the second pass and the Run registry settings are written to a separate registry location, then a message box lets the user know that the computer needs to be manually restarted.

Syntax

```
pausefornotificationofcompletion
```

Example

```
pausefornotificationofcompletion
```

Parameters

N/A

Remarks

This command is supported for Windows NT only.

Platform

The Platform command is used to signify the script file platform. This should be the first command in the script file. This command is required for all script files.

Syntax

```
platform=platformname
```

Example

```
platform=Win95
```

Parameters

platformname

Valid values are : Win95, Win98, and WinNT.

Remarks

For Windows 2000/Me/XP, use the WinNT value.

RemoveFromAutoexec

The RemoveFromAutoexec command searches Autoexec.bat for the search string and deletes any lines that contain that string.

Syntax

```
removefromautoexec=searchstring
```

Example

```
removefromautoexec=scan.exe
```

Parameters

searchstring

Text to search for in the Autoexec.bat file.

RestartComputer

The RestartComputer command writes all necessary information to the registry for the second run, if needed, and then restarts the computer. This command is supported under Windows NT only.

Syntax

```
restartcomputer
```

Example

```
restartcomputer
```

Parameters

N/A

Remarks

This command is supported for Windows NT only.

RunProgram

The RunProgram command executes the program that is read from the provided Registry keys. If there are any parameters in the registry, they are used as well. This command uses the WinExec API call and does not wait for the program to complete. This command can be used when you need to run a program that RunRobot is unable to shut down. This is currently used for Fprot 95.

Syntax

```
runprogram=hKey, SubKey, Key
```

Example

```
runprogram=HKEY_LOCAL_MACHINE,
SOFTWARE\Windows\CurrentVersion\Uninstall\F-Prot 95,UninstallString
```

Parameters

hKey

The hKey that is passed into the RegOpenKey.

SubKey

The SubKey that is passed into the RegOpenKey.

Key

The Key that is passed into RegQueryValueEx.

RunRobot

The RunRobot command automatically runs and performs all IDOK and IDYES key presses for the antivirus or firewall uninstallation program dialog boxes and then writes to the registry for the Run key. This command is used to restart the Competitive Uninstaller after the computer has been restarted by the invoked antivirus or firewall uninstallation program.

Syntax

```
runrobot=IdleTimeOut
```

Example

```
runrobot=300
```

Parameters

IdleTimeOut [optional]

Maximum time in seconds for the Competitive Uninstaller to wait for the uninstallation program to return an idle message. If no value is supplied, the default is five minutes.

Remarks

This command requires that a program path and name be previously set using one of the following sets of commands:

```
GetUninstallProgramName
```

```
GetUninstallShieldProgramName
```

or:

```
GetUninstallProgramPath
```

```
SetUninstallProgramName
```

To add parameters to the uninstallation program set above, add the following command and specify the appropriate parameters:

```
AddProgramParameters=[parameters]
```

To add a script for the uninstallation program set above, add the following commands:

```
GetUninstallScriptPath
SetUninstallScriptName
```

RunRobotSendCancel

The RunRobotSendCancel command automatically runs and performs all IDOK, IDYES and IDCANCEL key presses for the antivirus or firewall uninstallation program dialog boxes, and then writes to the registry for the Run key. This command is used to restart the Competitive Uninstaller after the computer has been restarted by the invoked antivirus or firewall uninstallation program.

Syntax

```
runrobotsendcancel=IdleTimeOut
```

Example

```
runrobotsendcancel=300
```

Parameters

IdleTimeOut [optional]

Maximum time in seconds for the Competitive Uninstaller to wait for the uninstallation program to return an idle message. If no value is supplied, the default is five minutes.

Remarks

This command requires that a program path and name be previously set using one of the following sets of commands:

```
GetUninstallProgramName
GetUninstallShieldProgramName
```

or:

```
GetUninstallProgramPath
SetUninstallProgramName
```

To add parameters to the uninstallation program set above, add the following command and specify the appropriate parameters:

```
AddProgramParameters=[parameters]
```

To add a script for the uninstallation program set above, add the following commands:

```
GetUninstallScriptPath
SetUninstallScriptName
```

RunRobotSetRunRegistry

The RunRobotSetRunRegistry command automatically runs and performs all IDOK and IDYES key presses for the antivirus or firewall uninstallation program dialog boxes, and then writes to the registry for the Run key so that the Competitive Uninstaller is restarted once the computer has been restarted by the uninstallation program that is being invoked. This command should be used when the uninstallation program automatically restarts the computer and then creates a a dialog box that requires a keystroke after the restart.

Syntax

```
runrobotsetrunregistry=IdleTimeOut
```

Example

```
runrobotsetrunregistry=300
```

Parameters

IdleTimeOut [optional]

Maximum time in seconds for the Competitive Uninstaller to wait for the uninstallation program to return an idle message. If no value is supplied, the default is five minutes.

Remarks

This command requires that a program path and name be previously set using one of the following sets of commands:

```
GetUninstallProgramName
```

or:

```
GetUninstallProgramPath
SetUninstallProgramName
```

To add parameters to the uninstallation program set above, add the following command, and specify the appropriate parameters:

```
AddProgramParameters=[parameters]
```

To add a script for the uninstallation program set above, add the following commands:

```
GetUninstallScriptPath
```

```
SetUninstallScriptName
```

This command combines the features of RunRobot and SetRegistryForUninstallSecondRun.

SendAffirmativeToWindow

The SendAffirmativeToWindow command sends the IDOK and IDYES messages to the window whose window name has been passed in. This shuts down dialog boxes that are not controlled by RunRobot. This is used currently for Cheyenne.

Syntax

```
sendaffirmativetowindow=windowname, waittime
```

Example

```
sendwindowmessage=VshieldMonitor, 60
```

Parameters

windowname

The window name is the text in the header of the window. This parameter is used by FindWindowEx to locate the antivirus or firewall dialog box to which to send the message. This parameter is case sensitive.

waittime

Number of seconds to search for the window name.

SendAffirmativeToWindowAfterEnable

The SendAffirmativeToWindowAfterEnable command sends the IDOK and IDYES messages to the specified window. This command will repeat until the window name is found. When the window name is found, it will search for the specified button, and then enable the button. This can be used to shut down the Uninstall Shield dialog. This will shut down dialog boxes that are not controlled by RunRobot. This is currently used for FProt 95.

Syntax

```
sendaffirmativetowindow=windowname, buttonname, waittime
```

Example

```
sendwindowmessage=Remove Programs From Your Computer, OK, 60
```

Parameters

windowname

The window name is the text in the header of the window. This parameter is used by FindWindowEx to locate the antivirus or firewall dialog to which to send the message. This parameter is case sensitive.

buttonname

The text on the button to search for. This parameter is case sensitive.

waittime

The maximum number of seconds to search for the window name, button, and button enable status.

SendWindowsMessage

The SendWindowsMessage command sends a message to the window whose class has been passed in. This is designed to shut down antivirus or firewall programs that might be running so that they can be properly uninstalled.

Syntax

```
sendwindowmessage=windowname, ASCIIMessage
```

Example

```
sendwindowmessage=VshieldMonitor, 16
```

Parameters

windowname

The window name is the text in the header of the window. This parameter is used by FindWindowEx to locate the antivirus or firewall dialog box to which to send the message. This parameter is case sensitive.

ASCIIMessage

ASCII value for the windows message to be sent.

Remarks

Message 16 is the WM_CLOSE. This command cannot pass the wparam or the lparam.

SendWindowsMessageAndWait

The SendWindowsMessageAndWait command sends a message to the specified window. If the window is not found, it retries until the window is found or the wait time has expired. This shuts down windows that are not controlled by RunRobot.

Syntax

```
sendwindowmessage=windowname, ASCII message, waittime
```

Example

```
sendwindowmessage=VshieldMonitor, 16, 60
```

Parameters

windowname

The window name is the text in the header of the window. This parameter is used by FindWindowEx to locate the antivirus or firewall dialog box to which to send the message. This parameter is case sensitive.

message

ASCII value for the windows message to be sent.

waittime

Number of seconds to attempt to find the window.

Remarks

Message 16 is the WM_CLOSE. This cannot pass the wparam or the lparam.

SetRegistry

The SetRegistry command sets the specified registry key value.

Syntax

```
setregistry=hKey, SubKey, Key, Value
```

Example

```
setregistry=HKEY_CURRENT_USER, SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Windows, run, E:\S32UI\debug\s32ui.exe
```

Parameters

hKey

The HKey that is passed into RegCreateKey/RegOpenKey.

SubKey

The SubKey that is passed into RegCreateKey/RegOpenKey.

Key

The Key that is passed into RegSetValueEx.

Value

the Key value that is passed into RegSetValueEx.

SetRegistryForUninstallSecondRun

The SetRegistryForUninstallSecondRun command sets the Run registry setting so that after the RebootSystem command is performed, the Competitive Uninstaller automatically restarts.

Syntax

```
setregistryforuninstallsecondrun
```

Example

```
setregistryforuninstallsecondrun
```

Parameters

N/A

SetUninstallProgramName

The SetUninstallProgramName command combines the member variable set by GetUninstallProgramPath and the file name that is passed in to this command. RunRobot uses the result of this command.

Syntax

```
setuninstallprogramname=filename
```

Example

```
setuninstallprogramname=uninstall.exe
```

Parameters

filename

The file that performs the uninstallation.

SetUninstallScriptName

The SetUninstallScriptName command combines the member variable set by GetUninstallScriptPath and the file name that is passed in to this command. RunRobot uses the result of this command.

Syntax

```
setuninstallscriptname=filename
```

Example

```
setuninstallscriptname=install.log
```

Parameters

filename

The script that is used by the uninstallation program.

SetupForSecondRun

The SetupForSecondRun command reads the registry and prepares for a restart when a second pass is required. Information that is needed on the second pass and the Run registry settings are written.

Syntax

```
setupforsecondrun=runkeyname
```

Example

```
setupforsecondrun=run
```

Parameters

runkeyname

For Windows NT, the runkeyname is run. For Windows 95, this parameter is the name that is in the RunOnce key of the HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion registry key.

Remarks

- Windows NT: This command only needs to be used if the antivirus or firewall uninstallation program is different than the one that was run on the first pass.
- Windows 95: For uninstallation programs that require a restart, this command should be placed just before the RebootComputer or PauseForNotificationOfCompletion commands.

StopProgramForReboot

The StopProgramForReboot command is used to prepare for a restart by the uninstallation program when a second pass is needed. Information that is needed for the second pass and the Run registry settings are written.

Syntax

```
stopprogramforreboot
```

Example

```
stopprogramforreboot
```

Parameters

N/A