



USB Infecting Malwares

A detailed study of USB Removable drive infections

12th December 2008

*Rajdeep Chakraborty - aka ~MaliciousBrains~ (founder of www.malwareinfo.org & www.malwareanalysis.org)
rajdeep@malwareinfo.org | rajdeep.chakraborty@gmail.com*

The purpose of this article is to show how USB infections propagate from one system to another and what we can do to stop the propagation of these USB Malwares.

Introduction

With the improvements in technology, there are times when we are faced with questions that don't have, at times, a straight forward answer. Instead, in a continuous manner, we all get involved in debating the pros and cons of a new technology. Unfortunately, it is true that whatever we have invented till date has its chances of use & misuse. This holds true for everything, starting from a needle to nuclear energy, from internet to a small device like USB Removable drives. Most of the times, these are invented with an intension to help us, however, misuse or overuse bring us face to face with their negative effects.

In this digitally connected world the emergence of the portable USB Removable drives has come as a definite blessing. Gone are the days when Floppy Diskettes or Re-Writable CD or DVD ROMs were the medium of carrying data in your pocket or carry bags. Although these were portable enough but they couldn't provide the usage freedom or ease that we get now days by using a USB Removable drives. USB Removable drives are not only portable but also provide an array of usage benefits. As technology is developing, USB Removable drives are getting more and more capable to hold huge amount of data, higher data transmission rates, endless opportunities for biometric usage, act as a portable and extended part of your HDD that you can carry around, boot system or run applications without HDD interaction etc. Possibilities are endless and every second day we see these devices coming up with newer and more technologically advanced features and modifications. Added to these, their easy availability and low costs are making them hugely popular.

Windows Autorun Feature –

The Autorun Feature is a functionality that has been provided in the Windows OS which would react in a predefined way when certain devices like CD/DVD ROMs are inserted in the specified drive. In Windows OS, Autorun is handled by the Explorer.exe process. It was introduced in the Windows OS keeping in mind the increase in the number of Multi Media Presentations that is used by Corporate Users/Students/Media Industry people etc. Business Cards, Product Presentations, Business and Project Presentations, Presentations for Seminars, Media and Graphics demos for designers etc represent just a fraction of the areas where CDs/DVDs were extensively used. Identifying these requirements Microsoft had made the Windows OS smart enough to recognize and auto executes the content of these CDs/DVDs whenever a CD/DVD was inserted. The whole point of providing this feature was to make it as easy as possible for the users to enjoy the content of the CD/DVD even if he/she is oblivious of the actual file or program that is launching the content. This Autorun feature is enabled by default, but we also have the option to manually enable/disable it as well.

Autorun.inf -

An Autorun CD/DVD has a file in its root called "*Autorun.inf*". We will look into the contents of this file a little later, but as of now, it is important to know that when the Autorun Feature of Windows is enabled (by default/manually), the Windows Shell will parse through the content of the Autorun.inf file and execute the instructions provided in it. This means, the moment an Autorun CD/DVD is inserted, the contents of the Autorun.inf will be executed and the referred launch program of that CD/DVD will load automatically.

How does Autorun work -

A **WM_DEVICECHANGE** message is sent to all applications whenever some hardware change occurs. It even includes when a flash drive (or other removable device) is inserted or removed. This message is sent to all the top-level windows according to their z-order. This means that the window at the top most position receives the message first, then only the window below it and so on. For reference, we can also trap this message in our own application by using the below API:

```
Public Declare Function CallWindowProc Lib "user32" Alias "CallWindowProcA" (ByVal lpPrevWndFunc As Long, _  
    ByVal hWnd As Long, _  
    ByVal Msg As Long, _  
    ByVal wParam As Long, _  
    ByVal lParam As Long) As Long
```

The **wParam** parameter of this message contains code which specifies exactly what event occurred. One of the events useful to us is **DBT_DEVICEARRIVAL**. This message is sent after a device or a media (CD\DVD) has been inserted. A program will receive this message when the device is ready for use, at the same time when Explorer displays the AutoPlay Window which lets us choose what to do with the inserted media.

The Windows Shell processes the **WM_DEVICECHANGE** messages and sends an interrupt request. The first thing that happens here is, the Windows Shell will check the registry entries to determine if Autorun Function is enabled for the

associated drive. If the Autorun Function is enabled, the Windows Operating System tries to locate the "Autorun.inf" file in the root directory of the newly entered device. Once this file is located, the instructions mentioned in the file will be executed.

What is inside Autorun.inf –

Autorun.inf is the primary configuration file that contains instructions for the Autorun Feature. This file directs the Windows Shell on how to and what to load and run from the device or drive with which it is associated. The instruction in an Autorun.inf file will define the below mentioned things:

- Application that will automatically run when the associated drive is activated or plugged in
- Icon that will represent the drive when it is viewed with My Computer or Windows Explorer
- Menu options to be displayed when the associated drive letter is right clicked from My Computer or Windows Explorer

Autorun.inf files have one or more sections where each of the section has a name that is enclosed in square brackets. Each section on the other hand will have a series of instruction that the Windows Shell executes when a device or a media (CD/DVD) has been inserted. So in a sense Autorun.inf files are very much like the .ini files in structure. Each instruction under the sections determines how the Autorun operation occurs. The following five commands are available:

- **Defaulticon** - Specifies the default icon for the application.
- **Icon** - Specifies the path and the file name of an application-specific icon for the drive.
- **Open** - Specifies the path and the file name of the startup application.
- **Shell** - Defines the default command in the shortcut menu of the drive.
- **Shell\verb** - Adds options to the right-click shortcut menu of the drive.

Example Autorun.inf File:

```
[Autorun]
Open=sys.exe /argument1
Icon=\foldername\filename.dll,5
```

Description of sections & instructions:

- **[autorun]** - This is the primary, required section name
- **open=sys.exe /argument1** - *Open* is the keyword to determine which application to load/execute
- **sys.exe** - This is the value referring to the application that will be automatically started. The system looks for the file in the root directory of the inserted disk. If we want to access a file located in a specific folder or subdirectory, we can specify a path relative to the root. E.g. **open=foldername\sys.exe**
- **/argument1** - This is the switch that is passed to the application as a command line. This can be any command line parameter that can be used. However, this command line parameter must be supported by the application
- **icon=\foldername\filename.dll,5** - *Icon* is the keyword to determine the icon associated for the drive
- **filename.dll** – This is the value referring to the file containing the icon
- **,5** – This is the argument to the icon resource specifying which icon to display.

How can USB Removable drives become a "Threat"?

Unfortunately, it is because of this fact that these USB Removable drives are so popular and are so commonly used to transfer/share data between systems that are not connected to each other, have limited connectivity or are physically located at different places, they are becoming a prime target for attackers or Malware authors who use them as a medium for spreading infections from one system to another in a very successful way. Off late there has been a sharp rise in the number of Malwares that are spreading through these USB Mass Storage devices. The moment you plug in the USB Removable drive and try to access it you might probably get infected. The reason for writing this article is to show how these infections traverse from one system to another and what we can do to stop the propagation of these USB Malwares.

Autorun.inf for USB –

As pointed out above, Autorun Feature is primarily intended for public distribution of applications on CD/DVD ROMs. However, it is often useful to enable Autorun Feature on other types of removable storage media as well. As per the MSDN website, it's mentioned under the topic "*Autorun for Other Types of Storage Media*", Autorun only works on removable storage devices when the following criteria are met:

- The device driver must notified that a disk has been inserted by sending a **WM_DEVICECHANGE** message
- The root directory of the inserted media must contain an *Autorun.inf* file
- The device must not have Autorun Feature disabled through the Windows Registry
- The foreground application has not suppressed Autorun Feature

In a normal scenario, whenever a USB Removable device is plugged in, the driver will send a **WM_DEVICECHANGE** message to the Windows Shell. This is where the first criteria is met. If the USB Removable device has an *Autorun.inf* file in its root, then the second criteria is also met. By default, the Autorun Feature is not disabled through the Windows Registry and it meets the third criteria as well. The fourth and the last criteria is not coming into play since we are not using any third party application to suppress the Autorun Feature.

Keeping the above scenario in mind, now let's say a user wants to open the USB Removable drive. The most obvious way he/she would go ahead is either by double clicking the newly installed Removable drive from *My Computer/Windows Explorer* or by selecting the "Open folder to view files" option in the AutoPlay window. Once either of these options is used to open the drive, as per the nature of the Autorun Feature, the application that is being referred to from the Autorun.inf will get executed.

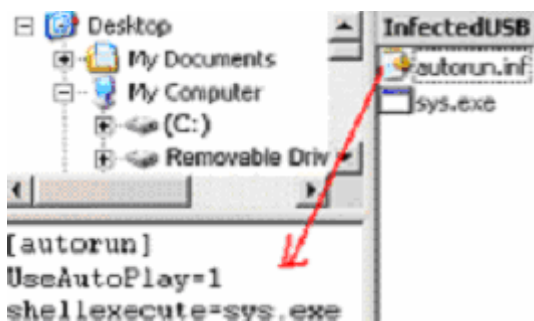
Note: AutoPlay is a feature in Windows where in the Windows OS will detect **DBT_DEVICEARRIVAL** event and will detect the contents of the device. Once it has detected the contents, depending on the content type, it will pop up a window with the most relevant options viz. *Open program, Copy pictures..., View slideshow..., Print pictures, Open folder to view files* etc.

Difference in USB Autorun.inf -

For USB Removable drives, the *autorun.inf* file needs to be a little different in order to be launched automatically.

Example of USB Autorun.inf File:

```
[Autorun]
UseAutoPlay=1
ShellExecute=sys.exe
Shell\open\command=sys.exe
Icon= %SystemRoot%\system32\SHELL32.dll,2
Label=Label of flash drive
Action=Run Program
```



Description of sections & instructions:

- **[autorun]** - This is the primary, required section name
- **UseAutoPlay=1** - The entry that makes the USB actually autorun
- **ShellExecute=sys.exe** - This is the keyword to determine which application to load/execute
- **sys.exe** - This is the value referring to the application that will be automatically started. The system looks for the file in the root directory of the inserted disk. If we want to access a file located in a specific folder or subdirectory, we can specify a path relative to the root. E.g. **ShellExecute=foldername\sys.exe**
- **Shell\open\command=sys.exe** - This specifies the program that should autorun. Similar to ShellExecute
- **Icon=%SystemRoot%\system32\SHELL32.dll,2** - The keyword to determine the icon associated for the drive
- **%SystemRoot%\system32\SHELL32.dll** - This is the value referring to the file containing the icon
- **,2** - This is the argument to the icon resource specifying which icon to display.
- **Label=Label of flash drive** - The label parameter is used to specify the name of the drive
- **Action=Run Program** - This describes the action that will be performed. This parameter is used by Windows Explorer in the AutoPlay Window

Example of a worm that spreads through USB Removable drive –

Worm:Win32/SillyFDC –

This is a generic detection for a family of worms that spreads by copying itself into the Removable Drives. This worm may download other Malwares into the system, however, for the sake of the article; we are focusing only on the propagation methodology of this worm.

Method of propagation - These worms spread by copying itself into the root of the system drive and also into the root of the Removable drive. It also creates an autorun.inf file with instructions that will invoke the Malware when the said drive is accessed from My Computer/Windows Explorer.

Content of W32/SillyFDC Autorun.inf - Shown below is the content of the autorun.inf file that is created by W32/SillyFDC.

```
[Autorun]
open=file.exe
shell\Open\Command=file.exe
```

```
shell\open\Default=1
shell\Explore\Command=file.exe
shell\Autoplay\command=file.exe
```

As we have seen before, this file will be responsible in infecting a virgin system when an infected USB Removable drive is plugged in it and accessed. Opening an infected USB Removable drive will automatically execute the 'file.exe'. Once this infected file is executed it will copy itself to the below mentioned Shell Folders:

- %System%
- %Windir%
- %Temp%
- %UserProfile%
- %ProgramFiles%
- %SystemDrive%
- %CommonProgramFiles%
- %CurrentFolder%

It may copy itself to the above Shell Folders with names like *password_viewer.exe*, *CALC*, *calc*, *mscalx.exe*, *startupfolder*, *config_startupfolder.com*, *config_.com* etc.

It will create entries in the Registry locations mentioned below so that the infected process can run every time automatically when the system restarts.

- HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
- HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
- HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows\load"

Other activities that these USB Infectors may carry out in general are like:

- Bypasses Windows Firewall
- Downloads & Drops Additional Malwares
- Disables Task Manager/Regedit/CMD
- Disabled Folder Options
- Spread via Unprotected Network Shares

Some other USB Infecting Malwares are:

- Worm: VBS/SillyFDC.F
- Worm: Win32/SillyShareCopy.AC
- Worm: Win32/Autorun.A
- PWS: Win32/Wowsteal.ZE!inf
- Worm: Win32/Nuj.A
- Worm: Win32/Autorun.PH
- Worm: Win32/Nhatq
- Worm: Win32/Autorun.BO
- Worm: Win32/Autorun.RA
- Worm: Autolt/Renocide.gen!A
- Worm: Win32/SillyShareCopy.E
- Worm: Win32/VB.CD
- Worm: Win32/Emold.B
- Worm: Win32/Slenfbot.ACP
- Worm: Win32/Slenfbot.ACU

You can refer to these and many more USB Infecting Malwares from the *Microsoft Malware Protection Center* website. Please refer to the below URL to view complete descriptions about these USB Infectors:

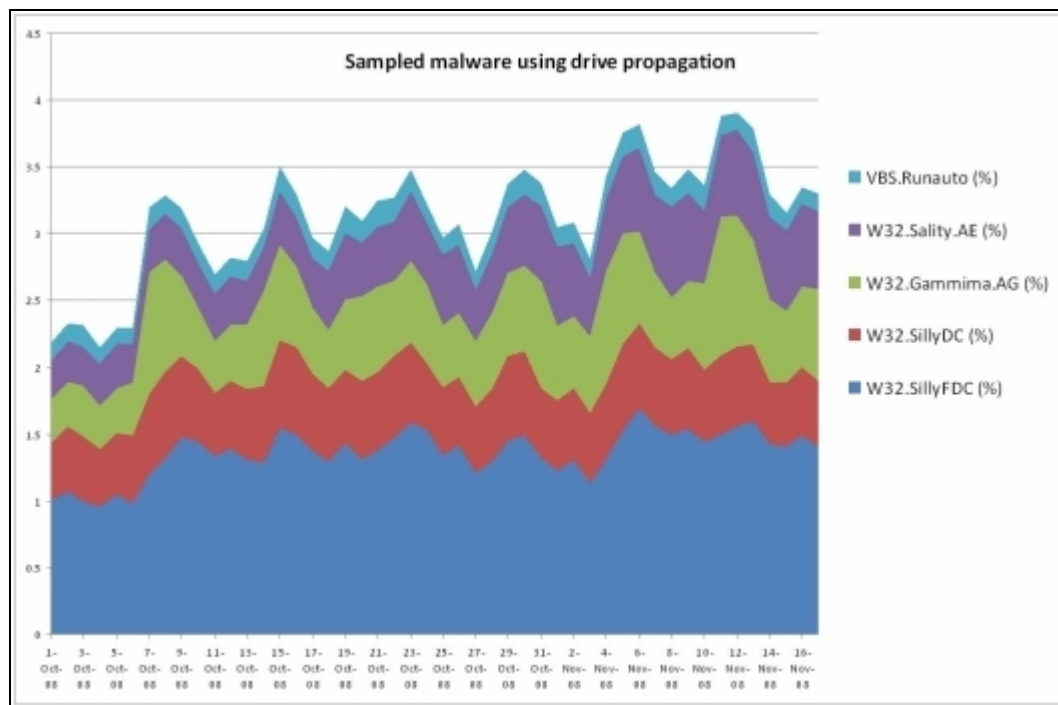
<http://www.microsoft.com/security/portal>

Increasing trend of USB Infectors –

It has been observed by Symantec and many other Antivirus vendors that there has been a rapid growth of these USB infecting Malwares. Almost everyday newer variants of these Malwares are detected in the wild and keeping in mind the frequency in which they are coming, it is becoming more and more troublesome for the Antivirus companies to keep their signatures updated. It has been published in the volume XIII of Symantec's Internet Security Threat Report (Page 56, Malicious Code Trends), that describes the propagation mechanisms of these Malwares. To quote from the report published:

"In the second half of 2007, 40 percent of malicious code that propagated did so as shared executable files (table 9), a significant increase from 14 percent in the first half of 2007. Shared executable files are the propagation mechanism employed by viruses and some worms that copy themselves to removable media. As stated in the "Malicious code types" section above, the increasing use of USB drives and media players has resulted in a resurgence of malicious code that propagates through this vector.

This vector lost popularity among malicious code authors when the use of floppy disks declined and attackers instead concentrated on other more widely used file transfer mechanisms such as email and shared network drives. However, as use of removable drives has become more widespread, attackers have again begun to employ this propagation technique. Although current removable drives differ from floppy disks, the principle remains the same, enabling attackers to make simple modifications to old propagation techniques"



The above graph shows the increase in propagation of Malwares that spread by infecting drives. From this graph also, its quite evident that the Malware Authors are using the Removable Drives, in a very successful way, as a their primary infection vector. However, certain configurations, settings and proper user awareness can definitely help in mitigating this threat vector to a great extent.

USB Removable devices - A blessing or a curse?

I have already pointed out that, at times we all get involved in debating the pros and cons of a new technology, so a pretty obvious question that may arise is, whether USB Removable devices are a blessing or they are more of a necessary evil? Keeping in mind the popularity and usage benefits of USB Removable drives, it is pretty much understood that they serve their purpose with utmost efficiency and has helped us in many ways. Although USB Removable drives are portable enough and may hold huge amounts of data or run applications without HDD interaction, the list of possible misuses & the associated risks are endless as well. It could be the prime cause for propagating infection from one computer to another and in the worst case in the entire network as well, sensitive or classified data may be stolen etc. Unfortunately, the "Threat" that these USB Removable drives pose in an Organizations perspective are endless as well.

In my opinion, although it may vary from person to person, it represents both side of the same coin. We simply can't draw a straightforward conclusion to this question until and unless we identify, on what aspect we are considering its usage? The question that we are often faced with is, to what extent or under what circumstances the usage of these devices can be allowed in an Organization. Depending on the "Risk Acceptance Level" of an Organization and considering the numerous "Risks" or "Threats" associated, the usage of these USB Removable Devices has been restricted in many Organizations. In the next section we will try to understand about how to mitigate the risk of these USB Malwares.

Protection from USB Malwares –

When it comes to protection from these USB borne Malwares, we have to identify what is the level of protection that we need. In other terms, as mentioned above, depending on the "Risk Acceptance Level" of an Organization, we can decide whether to permanently block the usage of USB Removable devices by disabling it from the System BIOS or we want to allow restricted usage by modifying certain parameters in the Windows Registry or by implementing certain restrictions through the Group Policy. In the below sections, I will speak about certain methodologies that can be implemented to mitigate the risk of USB Malwares.

Disable Usage of Autorun.inf Completely -

A quick way to deal with the Autorun.inf file is disable the usage of Autorun.inf files completely from the system. It can be done by adding a key called Autorun.inf in the following Registry path:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\IniFileMapping

Create an entry under the newly created Autorun.inf key called @ and setting the value of the @ entry to "@SYS:DoesNotExist". Or simply copy the below mentioned text to a text file and save it with a .reg extension:

```
----- Begin Copy -----  
Windows Registry Editor Version 5.00  
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\IniFileMapping\Autorun.inf]  
@="@SYS:DoesNotExist"  
----- End Copy -----
```

Once this is done, double click this .reg file to incorporate the setting.

What this setting does is, it tells Windows to treat Autorun.inf as if it were a configuration file from a pre-Windows 95 application. "*IniFileMapping*" is a key which tells Windows how to handle the .inf files. In this case it means "whenever you the OS has to handle a file called Autorun.inf, don't use the values from the file. Rather, there is an alternative value at the below Registry Path:

HKEY_LOCAL_MACHINE\SOFTWARE\DoesNotExist

However, since this "*DoesNotExist*" key, in reality doesn't exist, the OS assumes as if the Autorun.inf itself is completely empty, so the instructions mentioned in the Autorun.inf are not executed.

The only draw back of this method is, it will make the system inert to all the Autorun.inf files. Even if an Autorun.inf file is in the CD/DVD drive's root folder, it will not Autorun the CD/DVD automatically.

Please refer to the below URL for a complete understanding of how the *IniFileMapping* and *Registry* works:

[http://technet.microsoft.com/hi-in/library/cc722567\(en-us\).aspx](http://technet.microsoft.com/hi-in/library/cc722567(en-us).aspx)

Disable Autorun Feature for Optical Media –

Traditionally, Autorun Feature can be bypassed by keeping the "Shift" key pressed as we enter the Optical media (CD/DVD). This will bypass the Autorun Feature for the Optical media even if there is an autorun.inf file in the root of the media. However, from Windows Vista onwards, this method of bypassing the Autorun Feature for Optical media doesn't work.

There is another way of disabling the CD/DVD Autorun Feature by making certain changes in the below mentioned Registry Key:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Cdrom

If we change the value of the "*AutoRun*" DWORD entry to "0" then this will explicitly disable the Autorun Feature of the Optical media. Apart from this DWORD entry under this key, you may find a Multi-String entry called "*AutoRunAlwaysDisable*". Do not change the value of this entry because your CD-ROM drives might not operate properly if these values are changed.

Disable Autorun Feature (Customized) –

Autorun Feature can be disabled from any drives by changing the below Registry Key:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\Explorer
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer

Autorun is also disabled on any drive if it is disabled by the DWORD value of "*NoDriveTypeAutoRun*" (in HKLM or HKCU). To disable Autorun Feature for any drive, the value should be set to "0xFF"

Note: But if the entry ("*NoDriveTypeAutoRun*") appears in HKEY_LOCAL_MACHINE, the corresponding entries in HKEY_CURRENT_USER are ignored.

This entry is a bitmapped value. To disable Autorun on a particular type of drive, set the bit representing that drive type to 1. To disable more than one type of drive, set the bits representing each type to 1, or sum the hexadecimal values of the representative bits.

Hex	Description
0x1	Disables Autorun on drives of unknown type
0x4	Disables Autorun on removable drives
0x8	Disables Autorun on fixed drives
0x10	Disables Autorun on network drives
0x20	Disables Autorun on CD/DVD drives
0x40	Disables Autorun on RAM disks
0x80	Disables Autorun on drives of unknown type

0xFF	Disables Autorun on all types of drives
------	---

By default, Autorun is disabled on removable drives, such as the floppy disk drive (but not the CD-ROM drive), and on network drives. The default value 0x95 (149) is the sum of 0x1, 0x81 (unknown types), 0x4 (floppy drives), and 0x10 (network drives).

Disable On-Click Autorun Feature –

Even when Autorun is disabled, opening a drive, by double-clicking or pressing Enter, containing *Autorun.inf* in its root directory will still activate Autorun Feature. This happens because Explorer determines if the drive type is capable of Autorun. Usually this is how Malwares spread via USB Removable drives (the Malware binary is executed not when a USB Removable drive is inserted, instead, it is executed when a user opens the drive by double clicking the drive icon from *My computer/Windows Explorer*).

There is a way to disable this *On-Click Autorun Feature*. Manually editing the "*NoDriveTypeAutoRun*" DWORD entry to the required Hex value (refer to the table below) will disable the Autorun Feature for the corresponding drive. This method will help us to explicitly disable Autorun Feature for the chosen drive and instead of disabling it for both CD/DVD and USB Removable drives; we can choose to disable it for only the USB Removable drive. At times the Autorun Feature for CD/DVD drives is useful but for the USB Removable drives it is nothing more than a "Security Threat". This will at least safe guard the systems from getting infected with the ever increasing number of USB Malwares.

Autorun Feature can be explicitly disabled by changing the below Registry Key:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\Explorer

Set the value of the "*NoDriveTypeAutoRun*" DWORD entry to the required Hex figure from the below table:

Description	Hex
Disables Autorun on all types of drives	0xB5
Disable Autorun only for CD/DVD drives	0x95
Disable Autorun only for USB Removable drives	0xB1
Disable Autorun both for CD/DVD and USB Removable drives (default)	0x91

Disable Autorun Feature with gpedit.msc –

We can also disable the Autorun Feature by opening the gpedit.msc console. This gpedit.msc console can be used to turn off Autorun / Autoplay functionality. To use the gpedit.msc method follow the below steps:

1. Go to **Start -> Run -> gpedit.msc**
2. **Computer Configuration -> Administrative Template -> System**
3. Double click the sub key "*Turn off Autoplay*" and enable it

To quote Microsoft's gpedit.msc dialog:

Turns off the Autoplay feature.

Autoplay begins reading from a drive as soon as you insert media in the drive. As a result, the setup file of programs and the music on audio media start immediately.

By default, Autoplay is disabled on removable drives, such as the floppy disk drive (but not the CD-ROM drive), and on network drives.

If you enable this setting, you can also disable Autoplay on CD-ROM drives or disable Autoplay on all drives. This setting disables Autoplay on additional types of drives. You cannot use this setting to enable Autoplay on drives on which it is disabled by default.

Note: This setting appears in both the Computer Configuration and User Configuration folders. If the settings conflict, the setting in Computer Configuration takes precedence over the setting in User Configuration.

Note: This setting does not prevent Autoplay for music CDs.

Note: However, it should be kept in mind that this setting does not prevent Autoplay for any removable drive that is formatted with CDFS.

For any miss configuration of the Autoplay settings, Microsoft has provided a downloadable utility called the "*Microsoft AutoPlay Repair Wizard*". The Microsoft AutoPlay Repair Wizard will scan the computer devices to find defective AutoPlay settings, and attempts to fix those it finds. This utility can be downloaded from the below URL:

<http://www.microsoft.com/downloads/details.aspx?familyid=C680A7B6-E8FA-45C4-A171-1B389CFACDAD&displaylang=en#Overview>

Conclusion –

From the Organizational point of view USB Removable drives have become a necessary evil. The reasons for it being considered a curse has already been pointed out and also it can continue to a full fledged debate. I will conclude this article with a note that most of the Security threats in today's IT infrastructure are more because of personal lapses. However, if used in a restricted way then the nuisance that might arise from these devices can be kept under control and also the "Threats" that this small useful device can pose to the Organizations integrity, credibility and its infrastructure can also be, to a considerable extent, brought down.

However, as easier way is to use small third party utilities. I have myself created a small application that protects the system from USB borne Malwares. Its called USB Protect and can be downloaded from the Utilities section of MalwareInfo.Org. Below mentioned are the details about this application:

USB Protect 1.0.0 (1.23 MB)

Published: January 23, 2009

Aka: [USB Protect](#)
File size: 1.23 MB

Installation: [Simply extract and run USB Protect.exe](#)

Description:

USB Protect runs in the background and monitors the DBT_DEVICEARRIVAL events. Once it detects a DBT_DEVICEARRIVAL event, it identifies if its a REMOVABLE media like USB. If it detects a USB DBT_DEVICEARRIVAL, it detects the drive entry and checks for the existence of Autorun.inf and the malware binary that is being called through it. On a positive detection, it deactivates both the Malware binary and the Autorun.inf file. USB Protect also gives a voice confirmation when an Autorun.inf file is detected in the USB drive. On a positive detection, USB Protect changes the Malware binary to *.blocked* and Autorun.inf to *.usb* extensions, so nothing is deleted or lost. It creates a blank harmless autorun file so that Open With window doesn't appear when the USB Drive is clicked.



USB Protect will save log files in *C:\USBProtectLog* with names like *USBProtectLog_23012009_163525.log*

Shortcut Keys:

CTRL + NUM 1 --> Shows application window
CTRL + NUM 2 --> Shows About Me window

Sample log file entries will look like:

```
1/23/2009 4:46:15 PM: WM_DEVICECHANGE 537
1/23/2009 4:46:15 PM: wParam = DBT_DEVICEARRIVAL <---- detects device insertions
1/23/2009 4:46:15 PM: Device Type: DBT_DEVTYP_DEVICEINTERFACE
1/23/2009 4:46:15 PM: Device Name: STORAGE <---- detects if storage is USB Removable drive
1/23/2009 4:46:15 PM: Vendor\Product ID: REMOVABLEMEDIA
1/23/2009 4:46:15 PM: Device Unique ID: 7&14A32FOA&0&RM
1/23/2009 4:46:15 PM: Device CLSID: {53F5630D-B6BF-11D0-94F2-00A0C91EFB8B}
1/23/2009 4:46:15 PM: Drive Letter: I <---- detects drive letter
1/23/2009 4:46:15 PM: Renamed To: I:\USBProtect_23012009_164615.usb
1/23/2009 4:46:15 PM: Autorun Detected In Drive I
1/23/2009 4:46:15 PM: Autorun Path: I:\USBProtect_23012009_164615.usb <---- renames autorun.inf
1/23/2009 4:46:15 PM: Autorun Content: <---- displays the autorun.inf text
[AutoRun]
> open=Malware.exe <---- shows the malware binary
```

```
shell\open=Malware
> shell\open\Command=Malware.exe
shell\explore=Malware
> shell\explore\Command="Malware.exe"
1/23/2009 4:46:16 PM: Binary Renamed To: I:\Malware.blocked <---- deactivates malware binary
```

Reference –

<http://msdn.microsoft.com>
<http://www.microsoft.com/technet>
<http://en.wikipedia.org>
<http://www.codeproject.com>
<http://www.experts-exchange.com>