

Re: Controlling specific USB devices on Windows XP

Source: <http://www.derkeiler.com/Mailing-Lists/securityfocus/focus-ms/2006-06/msg00064.html>

- *From:* Chris Poldervaart <chris.listserv@xxxxxxxxxxx>
 - *Date:* Mon, 19 Jun 2006 08:02:19 -0600
-

I did some testing on the USB AutoRun/AutoPlay as a source for running malicious code theory, and this is what I came up with:

The first step is to ensure whether it is possible for this to occur. The answer is without a doubt yes. I saw it first hand with a USB device bought from Best Buy that had a hard coded partition which mimicked a CD-ROM. When inserted, that partition would be recognized as a CD-ROM device, and would autorun the content. This device was special, however, as this partition was not accessible by the user. It seemed to have been hard coded with whatever device setting would trick MS Windows into thinking it was a CD-Rom rather than a USB device. This fact, and research on the Net leads us to believe that a special device is required to execute an application automatically from a USB device. If this were the case, then the devices used to propagate our malware would have to be "special" and thus creates new questions about access to these types of devices, and lots of other "how" type questions.

Microsoft provides some insight to this on their MSDN site:

"AutoRun for Other Types of Storage Media

AutoRun is primarily intended for public distribution of applications on CD-ROM and DVD-ROM. However, it is often useful to enable AutoRun on other types of removable storage media. This feature is typically used simplify the debugging of AutoRun.inf files. AutoRun only works on removable storage devices when the following criteria are met:

- * The device must have AutoRun-compatible drivers. To be AutoRun-compatible, a driver must notify the system 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 disabled through the registry.
- * The foreground application has not suppressed AutoRun."

All but the first bullet seem to be a given. What makes a device special is the "AutoRun-compatible drivers". My guess is that most of this has to do with the hardware encoding that establishes which drivers are to be loaded, or providing a special set of drivers for a specific device. Both of these options seem to rule out USB autorun as an initial infection vector in general (assuming we are not dealing with any special hardware).

So I set out to determine how a user can become automatically infected from a typical USB device containing malware using autorun. Here is what I found.

A machine taking advantage of AutoRun had the following registry keys in a user's NTUSER.DAT file:

Re: Controlling specific USB devices on Windows XP

```
\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\ {CLSID}\Shell\Autoplay
\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\ {CLSID}\Shell\Autoplay\DropTarget
\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\ {CLSID}\Shell\AutoRun
\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\ {CLSID}\Shell\AutoRun\command
\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\ {CLSID}\Shell\Open
\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\ {CLSID}\Shell\Open\command
```

Now, testing has shown that even a USB device with no autorun.inf will create the Autoplay and DropTarget keys. It is the AutoRun and Open keys that are created on the device when an application is executed from the autorun.inf on the device. They contain the following data —provided from the test scenario below:

```
\Shell\AutoRun\command\F:\.\cmd.exe
\Shell\Open\command\F:\.\cmd.exe
```

So, on a typical device, can malware be executed automatically? The answer is yes, and no. It seems that there are some devices that are hard-coded in such a way that the device does allow for the WM_DEVICECHANGE message call.

These seem to be "normal" consumer USB flash drives. It is kind of iffy on whether a device will meet that criteria or not, but my research is revealing that most fall into the "or not" category. So in probably 90% of the cases (just pulled that from fake-statistic-land) a USB device will not autorun anything upon insertion into the system.

So assuming that our infected user has a typical USB device, how does malware get from it to the system without the user executing it manually?

Autorun.inf still applies. While the device will not execute autorun.inf upon insertion, there is another means by which autorun can be used to accomplish this task fairly simply. This is best explained by re-creating the process by which I discovered this.

First, the device needs to have autorun.inf. I made one that was identical to the infected USB device (with obvious exception of the call to malicious code). Cmd.exe was the application of choice, so the autorun.inf looks like this:

```
[AutoRun]
Open= .\cmd.exe
shell\Open\command= .\cmd.exe
```

I tried this both with the code (cmd.exe) on the USB device itself, and also on the system using the complete path. Both work fine. In our example, as with the real situation, the malware exists on the device. You will also notice a direct correlation between these "open" statements and the registry keys listed above.

I then copied cmd.exe to the root of the USB device since that is where I am telling autorun.inf to execute it from.

Then I plug the device into the system...and what happens? I get the Autoplay window that asks me what I want to do: Copy pictures, View a slideshow, Open a folder, or take no action. This is the autoplay function that happens whenever any removable device is plugged into the system. This is also why all removable device registry entries will have "Autoplay" keys.

Windows is trying to be helpful by automatically searching the device for media files, documents, etc and

Re: Controlling specific USB devices on Windows XP

offering you a convenient way to get to them. I can either chose to open the folder, select "Do Nothing", or even hit cancel. No cmd.exe (and no malware).

This is where things start to happen. Lets say a user wants to access that USB drive (if he/she didn't immediately open it via autoplay --or if autoplay is disabled). How does one go about that? My guess is that the typical user will go to "My Computer" and click the drive letter that they want to open (from the start menu) or double-click the icon if within the "My Computer" browser window. This is where autorun.inf comes into play.

Doing either of these actions will execute autorun.inf, and in turn, cmd.exe. Bam!

Just trying to open the browser window to the USB device will execute cmd.exe. This happens because the default "Open" command is replaced by a default "Autorun" command. This is easily seen by right-clicking on the drive letter and viewing the context menu. The system recognizes the autorun.inf, and assumes that if you did anything with that volume, you would prefer to execute whatever autorun.inf points to. This makes malware coders happy.

While this seems kind of clunky trying to explain it in text...test it yourself. It is quite easy...and fun! Who knows, you may have the magic USB device that actually MAY autorun upon insertion. A simple way to test is to create an autorun.inf file on your USB device that has this content:

```
[autorun]
shellexecute=c:\windows\system32\cmd.exe
```

That's it. Just unplug your device, re-insert, and try and access it from "My Computer" and you will get a command prompt instead. Now pretend that that was malicious code, and you will see how divine this method really is.

Of course, this presents another problem. When the user tries to access the USB device, and malware executes, nothing will happen as far as the user is able to witness (they aren't nice enough to pop open a terminal for you).

That might be suspicious, right? So how do we get around that while still executing our malware? Easy. Why not include a call to the Explorer browser from within the malicious code?

The user clicks on the USB drive icon, which looks to autorun.inf, which executes malware.exe, which executes ". ". To the user, the result is expected: Click on icon results in window opening to browse folder. You can try that one too...just go to Start --> Run and type a single period (". ") and hit return. Whalah! A browser window. It wouldn't be that hard for a piece of malware to use the same function to pop open a window in the root of whatever device it is executed from.

I know that there has been some discussion of disabling autoplay as a means of preventing this method of infection. While it certainly cannot hurt to disable...tactically this will do nothing to prevent the method of infection I describe in this message. Disabling of autoplay does prevent the system from looking into the device for media files and such, and will prevent the user being prompted for an action upon insertion. However, the attempted access of the device by clicking in "My Computer" still executes the autorun.inf.

There may me a more in-depth protection setting that I have not explored. I disabled autoplay using MS PowerToys TweakUI. If someone would like to test this with other known autoplay-disabling settings, please let do and let me know the results.

Distinction here that is easily confused is autoplay vs autorun. Most references I found on the Internet about

