

Re: IE handling of exe files

Source: <http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.general/2004-02/0684.html>

From: David Sworder (*GilGrissom_at_CSILasVegas.com*)

Date: 02/07/04

Date: Sat, 7 Feb 2004 13:46:15 -0800

> *Thanks for your reply. How feasible would it be to set it up to actually
> run over the net with zero permissions but in a sort of demo mode
(assuming
> that would make sense for this application and in this case it does).*

I've been where you are and have posted the very same questions that you're posting now and can tell you that the path you're on leads to a dead end. Here's the basic scoop on this:

Around the time that .NET was released, Microsoft started hyping something called zero-touch deployment. The idea is that you place a .NET executable on a web server (be it IIS, Apache, or whatever). Internet Explorer would then download the EXE. At some point IE would look at the executable and determine if it were a .NET exe. If so, it would try to run it, automatically downloading all of your custom assemblies on which your EXE depends. The idea was cool, but it had problems:

First of all, it was slow. Not only would your app hang while the .NET runtime tried to download all of your other assemblies, but as posted in the last message, .NET likes to "probe" for culture specific assemblies. Aside from the slowness, with .NET 1.0, the downloaded assembly didn't have access to do much. Pretty much anything your app tried to do, even doing seemingly innocuous things like opening a new form, would lead to some kind of warning that the user would have to read. Other operations would just throw exceptions. If that weren't enough, your app was loaded into the IEEExec process which means that if your application made use of certain legacy COM components (i.e. the web browser control), you'd end up with all kinds of weird threading exceptions if you didn't configure everything just right.

Around the time .NET 1.1 came out, Microsoft was getting some bad press for virus vulnerabilities. So, to score some press points, they announced that they were making .NET 1.1 "more secure"... part of this initiative included tightening the security in such a way that .NET exes located on a web server wouldn't even *execute.* So much for zero-touch deployment over the Internet.

With the next release of .NET, Microsoft will reverse course again,

allowing these remote EXEs to execute and with more permissions than 1.0. Exactly what those permissions are I'm not sure. I don't think it's been announced. But as you can see, the damage is already done. You as the developer have no idea whether your users will be using 1.0, 1.1, or this new Whidbey version of .NET so this idea of putting an EXE on a web server and have users run it just a no good. This idea is only practical in certain types of Intranet scenarios... and from what I can tell, there's no way to force the IE user to get a the "Save As.." dialog without forcing him through the context menu.

If you absolutely refuse to use an MSI file for deployment, you're going to have to find a clever way to get the user to run your executable with FullTrust. One thing I've done in the past is put some instructions on the site telling users to right-click the exe link, save it to their harddisk, and run it. That way the EXE runs with full permissions. As part of the startup code, you can modify the security settings on the user's computer so that it grants FullTrust to all other EXEs originating from your domain. I don't really recommend this though. Then you're assuming that the user has administrative access to his own machine, etc. There also might be a backdoor through ActiveX [browser-based unmanaged code] that would allow you to modify the user's security settings, but I haven't tried this.

I think for now you're stuck using an MSI file. :(

--
Sincerely,
David Sworder
<http://www.CodeFanatic.com>