

## dotNet Protector 5® tutorials

### Building a Visual Studio setup project with protected assemblies

#### 1. Exe Projects with embedded runtime

If dotNet Protector's runtime is embedded in the exe, VS 2005 can't successfully build a setup project with it (it complains about the missing embedded.netmodule).  
Ignore the missing PvLogiciels.dotNetProtector.Runtime dependency warning (this assembly is also embedded in your exe).

##### **Solution :**

Build the vdproj with a dummy netmodule and then remove this netmodule from the built msi.

##### **Content of EmbeddedSetupPatch.zip**

embedded.netmodule : the dummy netmodule used by PV Logiciels to compile the dotNet Protector® embedded runtime.  
MsiRemoveEmbeddedModule.exe : a command-line utility to remove the dummy netmodule from the generated msi.

##### **Executables built by dotNet Protector when using the 'runtime embedding feature'**

Starting with the v2 of .NET, assemblies can run in 64 bit mode on 64 bit platforms.

If your assembly is compiled as 'x86 cpu', it will run in a x86(emulated on 64 bits windows) box on each platform. In this case, dotNet Protector will build a single x86 protected exe.

If your assembly is compiled as 'any cpu', the default for pure-IL projects (C#, VB.NET.), it will be converted to a 64 bits executable at run time on 64 bits windows. In this case, dotNet Protector will build 3 executables, one for each platform : x86, ADM64 and Itanium.

If you don't have the 64 bits platforms available to test the 64 bits exe, forget them and distribute only the x86 one. If you can test the 64 bits, you can build up to 3 setup projects, one for x86, one for AMD64 and one for Itanium. ( change the deployment project's 'TargetPlatform' property accordingly)

##### **Building the vdproj project**

This step is the same for each platform.

To successfully build the deployment project, you need to provide a compatible embedded.netmodule. Copy the file from EmbeddedSetupPatch into the same location as your exe embedding dotNet Protector's runtime.

Build your project. In the following, we suppose the vdproj builds *mysetup.msi*.

**WARNING!** If you run your exe with an embedded.netmodule on disk, the framework will load this one instead of the embedded one, and your program will fail. This dummy module should be absent in the distribution folder. The second step removes embedded.netmodule from the distribution folder by patching the msi database.

Run the command

MsiRemoveEmbeddedModule mysetup.msi

After this step, mysetup.msi won't install embedded.netmodule

##### **Automatic build of the vdproj project**

You can automate the build and patch process using a batch file

Suppose

the solution path is 'C:\My Projects\mysetup.sln'

the dotNet Protector output path is 'C:\My Projects\MyExe\Protected'

EmbeddedSetupPatch has been uncompressed in C:\ EmbeddedSetupPatch

## X86 Only

Say the project name is mysetup, its output directory is 'C:\My Projects\mysetup\Release'

The batch file could be

```
@ECHO OFF
COPY /Y "C:\dotNetProtector\EmbeddedSetupPatch\embedded.netmodule" "C:\My Projects\MyExe\Protected"
SETLOCAL
CALL "%VS80COMNTOOLS%\VSARS32"
"%VSINSTALLDIR%\Common7\IDE\DEVENV" "C:\My Projects\mysetup.sln" /build "Release" /project mysetup
"C:\EmbeddedSetupPatch\MsiRemoveEmbeddedModule" "C:\My Projects\mysetup\Release\mysetup.msi"
ENDLOCAL
DEL C:\My Projects\MyExe\Protected\embedded.netmodule
```

## Multiple Platform

Say the dotNet Protector output path are 'C:\My Projects\MyExe\x86', 'C:\My Projects\MyExe\AMD64' and 'C:\My Projects\MyExe\Itanium' respectively

the x86 project name is mysetupx86, its output directory is 'C:\My Projects\mysetupx86\Release',  
the AMD64 project name is mysetupx64, its output directory is 'C:\My Projects\mysetupx64\Release', and  
the Itanium project name is mysetupia64, its output directory is 'C:\My Projects\mysetupia64\Release'  
The batch file could be

```
@ECHO OFF
COPY /Y "C:\dotNetProtector\EmbeddedSetupPatch\embedded.netmodule" "C:\My Projects\MyExe\x86"
COPY /Y "C:\dotNetProtector\EmbeddedSetupPatch\embedded.netmodule" "C:\My Projects\MyExe\AMD64"
COPY /Y "C:\dotNetProtector\EmbeddedSetupPatch\embedded.netmodule" "C:\My Projects\MyExe\Itanium"
SETLOCAL
CALL "%VS80COMNTOOLS%\VSARS32"
"%VSINSTALLDIR%\Common7\IDE\DEVENV" "C:\My Projects\mysetup.sln" /build "Release" /project mysetupx86
"C:\EmbeddedSetupPatch\MsiRemoveEmbeddedModule" "C:\My Projects\mysetupx86\Release\mysetupx86.msi"
"%VSINSTALLDIR%\Common7\IDE\DEVENV" "C:\My Projects\mysetup.sln" /build "Release" /project mysetupx64
"C:\EmbeddedSetupPatch\MsiRemoveEmbeddedModule" "C:\My Projects\mysetupx64\Release\mysetupx64.msi"
"%VSINSTALLDIR%\Common7\IDE\DEVENV" "C:\My Projects\mysetup.sln" /build "Release" /project mysetupia64
"C:\EmbeddedSetupPatch\MsiRemoveEmbeddedModule" "C:\My Projects\mysetupia64\Release\mysetupia64.msi"
ENDLOCAL
DEL C:\My Projects\MyExe\x86\embedded.netmodule
DEL C:\My Projects\MyExe\AMD64\embedded.netmodule
DEL C:\My Projects\MyExe\Itanium\embedded.netmodule
```

## 2. Projects not embedding dotNet Protector's runtime

In projects not embedding the runtime, the vproj should build correctly, but in these projects, dotNet Protector's runtime is needed to run.

dotNet Protector's runtime is composed of 5 files :

- . PvLogiciels.dotNetProtector.Runtime.dll : this is the common runtime, needed in all cases
- . PvLogiciels.dotNetProtector.RuntimeV1.dll : this is the v1.1 framework compatible runtime. You don't need it if your assembly is a v2.0 or later assembly
- . PvLogiciels.dotNetProtector.RuntimeX86.dll : this is the v2.0 x86 runtime . You need it if your assembly is loaded by the v2.0(or later) .NET Framework, except if your assembly only runs on a 64 bit platform.
- . PvLogiciels.dotNetProtector.RuntimeAMD64.dll : this is the v2.0 ADM64 runtime. You need it if your assembly is loaded by the v2.0(or later) .NET Framework on windows x64, except if your assembly only runs on x86 (x86 instead of AnyCPU)
- . PvLogiciels.dotNetProtector.RuntimeItanium.dll : this is the v2.0 Itanium runtime. You need it if your assembly is loaded by the v2.0(or later) .NET Framework on windows IA64, except if your assembly only runs on x86 (x86 instead of AnyCPU)

If your assembly is a v2+ assembly and not built to run only on a 32 box (x86), you NEED 3 setup projects (for each platform)

Adding the runtime to your project. If the common runtime is present on your assembly's directory, Visual Studio will add PvLogiciels.dotNetProtector.Runtime.dll by itself. In the other case, add it manually in your application folder.

Depending on the version/platform of your setup project, you'll have to add additional files:

V1.1 assembly/x86 project : add RuntimeV1 and RuntimeX86

V1.1 assembly/x64 project : add RuntimeV1, RuntimeX86 and RuntimeAMD64

V1.1 assembly/IA64 project : add RuntimeV1, RuntimeX86 and RuntimeItanium

V2+assembly/x86 project : add RuntimeX86

V2+assembly/x64 project : add RuntimeX86 and RuntimeAMD64

V2+assembly/IA64 project : add RuntimeX86 and RuntimeItanium