Code injection in ASP.NET is not a well known issue. It is because in ASP.NET, code files are not inserted one into another dynamically (like in PHP). Programmers can only register controls in the web.config file or on a page. But dynamic code injection in ASP.NET is still possible. The aim is to insert C# (or VB.NET, etc.) code that is executed directly.

The attacker can achieve this in the following situations:

- When you use the **ProcessStartInfo** class in your code and execute commands which are put together from external sources.
- When your virtual path provider is able to read files from different servers, and parameters are taken from an external source.
- When you load a control dynamically and the source of the control is loaded from an external source.

The attacker can also manage to insert a file with code into your application directory.

What can command injection attack do

Simply anything that can be achieved programmatically.

Finding command injection vulnerabilities in Kentico

There is no direct procedure to find code injection, but here are some tips for discovering possibly vulnerable places in Kentico:

- Search for **ProcessStartInfo** in source code and check its input parameters.
- Analyze the virtual path provider module and search for any possibility of getting a file which is not a regular Kentico virtual file.
- Try to edit a transformation without administrator privileges.
- Search for usages of the **LoadControl()** method and check the input of the method.

Avoiding command injection

You will probably never have to deal with this issue because code injection only poses a threat in the special cases described at the beginning of this chapter. Nevertheless, the general recommendations are:

- Never load controls dynamically when their path is taken from an external source.
- Do not ever use ProcessStartInfo and other classes which execute commands or run .NET code.
- If you want to customize the virtual path provider or transformation management, be very careful.