La libreria HtmlSanitizer permette di depurare una stringa in input da costrutti sintattici alla base di un attacco XSS. Di seguito una funzione che utilizza tale libreria:

```
public static string SanitizeHtml(string html, params string[] blackList)
{
   var sanitizer = new HtmlSanitizer();
   if (blackList != null && blackList.Length > 0)
   {
     sanitizer.BlackList.Clear();
     foreach (string item in blackList)
      sanitizer.BlackList.Add(item);
   }
   return sanitizer.Sanitize(html);
}
```

Qui viene mostrato del codice C# vulnerabile alla XSS reflected:

```
string nome = Request.QueryString["nome"];
Response.Write("Ciao " + nome); // non sicuro
```

Di seguito lo stesso codice, messo in sicurezza:

```
string nome = Request.QueryString["nome"];
nome = System.Web.Security.AntiXss.AntiXssEncoder.HtmlEncode(nome, true);
Response.Write("Ciao " + nome);
```

Per ulteriori informazioni si veda: http://cwe.mitre.org/data/definitions/79.html,
CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting').

7.6.2 Code Injection

Come riconoscerla

L'applicazione esegue del codice ricevuto attraverso l'input che non è stato sufficientemente verificato. Un utente in grado di inserire codice arbitrario può prendere il controllo dell'applicazione e del server, se non sono state adottate tecniche di difesa in profondità.

Esempio:

Il codice seguente mostra come del codice C# può essere passibile di code injection:

```
String codiceUtente = request.Form["Codice"];

CSharpCodeProvider compiler = new CSharpCodeProvider();
CompilerParameters parametri = new CompilerParameters();
parametri.GenerateInMemory = true;
parametri.GenerateExecutable = true;

try
{
  CompilerResults risultati = compiler.CompileAssemblyFromSource(parametri, codiceUtente);

  Assembly compilato = risultati.CompiledAssembly;
  exitCode = (int)compilato.EntryPoint.Invoke(null, new object[0]);
  […]
```

Come difendersi.

- È vietata qualsiasi esecuzione dinamica di codice ricevuto da canali non attendibili. Se è prorpio necessario compilare ed eseguire dinamicamente del codice dinamico, occorre allore predisporre una sandbox isolata, ad esempio AppDomain di .NET o un thread isolato.
- Devono essere effettuati tutti i controlli possibili per validare il codice in ingresso.
- Configurare l'applicazione da eseguire utilizzando un account utente limitato che non disponga di privilegi non necessari.
- Se è possibile optare per isolare tutta l'esecuzione dinamica utilizzando un account utente separato e dedicato che abbia privilegi solo per le operazioni e i file specifici utilizzati dal codice da eseguire, in base al principio denominato "Principle of Least Privilege". Il principio stabilisce che agli utenti