```
        }
    }

    public static byte[] ComputeHMACSHA1(byte[] toBeHashed, byte[] key)
    {
        using (var hmac = new HMACSHA1(key))
        {
            return hmac.ComputeHash(toBeHashed);
        }
    }

    public static byte[] ComputeHMACSHA512(byte[] toBeHashed, byte[] key)
    {
        using (var hmac = new HMACSHA512(key))
        {
            return hmac.ComputeHash(toBeHashed);
        }
    }

    public static byte[] ComputeHMACMD5(byte[] toBeHashed, byte[] key)
    {
        using (var hmac = new HMACMD5(key))
        {
            return hmac.ComputeHash(toBeHashed);
        }
    }
}
```

Code Listing 7

The hash class contains a method to generate the key by using the **RNGCryptoServiceProvider** that we have previously discussed. Then, there is the **ComputeHmacSHA256** method that takes a byte array of the code that you want to hash and a byte array for the encryption key.

To use the code, you do the following.

```
class Program
{
    static void Main(string[] args)
    {
        var originalMessage = "Original Message to hash";
        var originalMessage2 = "This is another message to hash";

        Console.WriteLine("HMAC Demonstration in .NET");
        Console.WriteLine("--------------------------");
        Console.WriteLine();

        var key = HMAC.GenerateKey();

        var hmacMd5Message =
HMAC.ComputeHMACMD5(Encoding.UTF8.GetBytes(originalMessage), key);
```