IOActive Security Advisory

Title	Buffer Overflow in MonoBigInteger Montgomery Reduction Method
Severity	High
Date Discovered	07.25.2007
Date Reported	08.24.2007
Date Disclosed	09.20.2007
Authors	Jason Larsen, Walter Pearce

Affected Products

Mono Framework, all versions prior to 1.2.6

Synopsis

IOActive has discovered an exploitable buffer overflow vulnerability in the Montgomery reduction method within the Mono Frameworks BigInteger Class (Mono.Math.BigInteger).

Description

When the unsafe Mono.Math.BigInteger.Montgomery.Reduce method (Mono\mcs\class\Mono.Security\Mono.Math\BigInteger.cs) is supplied two BigInteger objects of varying length, a buffer copy results in a class buffer overflow, allowing for arbitrary overwriting of object structures and pointers, and leading to code execution. Additionally, this method is utilized by the RSA Cryptography methods implemented within the Mono Framework, thus allowing for possible remote code execution under the correct conditions.

Technical Details

The buffer overflow takes place during the copy and reduction loop performed within the Reduce method of the Montgomery sealed class. The method itself takes three arguments—two BigInteger instances and a prime number—in order to perform the reduction.

```
public static unsafe BigInteger Reduce (BigInteger n, BigInteger
m, uint mPrime)
```

Two pointers are created, one for each BigInteger data segment, that contain the byte array representation of the large number. Finally, the actual reduction is performed, which consists of a pointer walk and copy between the two buffers without proper length limitations. The vulnerability exists when the array in *m* is greater than the array in *n*.

```
for (; j < m.length; j++) {
c += (ulong)u_i * (ulong)*(mP++) + *(aSP++);
*(aDP++) = (uint)c;
c >>= 32;
```

In the scenario where m is larger than n, aSP (the pointer to n) writes out of the array's bounds, overwriting memory within the block allocated for the method. In this manner, arbitrary code execution can be obtained by overwriting referenced object structures and pointers that are later de-referenced and operated upon.

Remediation

Mono 1.2.6 and later have been patched for this issue. Upgrade to the latest bug-fix version of the Mono Framework.