



**CYBSEC S.A.**

[www.cybsec.com](http://www.cybsec.com)

**Advisory Name:** Microsoft Windows DHCP-Client Remote Buffer Overflow

**Vulnerability Class:** Stack Buffer Overflow

**Release Date:** 08/29/2006

**Affected Platforms:**

- Microsoft Windows 2000 (≤ SP4)
- Microsoft Windows XP (≤ SP2)
- Microsoft Windows 2003 (≤ SP1)

**Local / Remote:** Remote

**Severity:** High

**Author:** Mariano Nuñez Di Croce

**Vendor Status:**

- Confirmed. Update released.

**Reference to Vulnerability Disclosure Policy:**

[http://www.cybsec.com/vulnerability\\_policy.pdf](http://www.cybsec.com/vulnerability_policy.pdf)

**Vulnerability Description:**

A remote buffer overflow vulnerability has been identified in Microsoft Windows DHCP-Client Service.

**Technical Details:**

The vulnerability specifically exists in the way the *dhcpcsvc.dll* system library processes DHCP messages.

To extend the limit of 255 bytes defined by the length octet of the DHCP Option Field, Microsoft uses the private Option Code *0xFA (250)*, allowing the use of larger option values.

Therefore, a client trying to obtain its IP information may receive a DHCPACK packet with the following structure defined, after the MAGIC\_COOKIE:

```
2F FF [255 bytes] FA 09 [8 bytes] 00
```

In the packet previously described, a *NetBIOS over TCP/IP Scope* option is being used, consisting of 264 bytes (including the NULL), effectively extending the limit of 255 bytes per Option.

The use of this Private Option Code is not restricted to be used only once after a known Option Code, so the following structure can also be used:

```
2F FF [255 bytes] FA FF [255 bytes] FA FF [255 bytes] ... FA FF [254 bytes] 00
```

Option Code 0x2F (*NetBIOS over TCP/IP Scope*) was used for our internal proof-of-concept but other Option Codes can also be used.

Once the client receives this DHCPACK message, an option-reassembling process is undertaken so that intermediate "0xFA"s and *Length octets* are removed. In this state the requesting client has an *N* bytes ( $N > 255$ ) valid DHCP option value, which will now be saved in the stack, after *trying* to perform a Unicode translation process (Windows 2000):

In function *DhcpRegSaveOptionAtLocation* 2516 bytes of stack space is reserved:

```
55          PUSH EBP
8BEC       MOV EBP,ESP
81EC D4090000 SUB ESP,9D4
```

Then, the function *DhcpOemToUnicode* is called with the following arguments:

```
8D85 2CF6FFFF LEA EAX, DWORD PTR SS:[EBP-9D4]
50          PUSH EAX          ; PUNICODE_STRING destination buffer
56          PUSH ESI          ; POEM_STRING buffer received
E8 3E010000  CALL DhcpOemToUnicode
```

In this function some values are calculated and passed to the *DhcpOemToUnicodeN* function:

```
0FB745 F8     MOVZX EAX,WORD PTR SS:[EBP-8] ; POEM_STRING->Length
8D4400 02     LEA EAX,DWORD PTR DS:[EAX+EAX+2] ; LEN = (POEM_STRING->Length+1)*2
50          PUSH EAX
FF75 0C     PUSH DWORD PTR SS:[EBP+C] ; PUNICODE_STRING buffer
FF75 08     PUSH DWORD PTR SS:[EBP+8] ; POEM_STRING buffer
E8 04000000  CALL DhcpOemToUnicodeN
```

In *DhcpOemToUnicodeN*, the following code is executed:

```
8B45 10      MOX EAX,DWORD PTR SS:[EBP+10] ; Argument 3 (LEN)
8B75 0C      MOV ESI,DWORD PTR SS:[EBP+C]   ; PUNICODE_STRING->MaxLength
03C0        ADD EAX, EAX
85F6        TEST ESI,ESI
66:8945 FA   MOV WORD PTR SS:[EBP-6], EAX  ; PUNICODE_STRING->MaxLength = LEN*2
```

As can be seen, the *MaxLength* member of the `UNICODE_STRING` structure is calculated based on the length of the OEM string provided (which, by the way, has already been multiplied by two).

Finally, the following call to *ntdll.RtlOemStringToUnicodeString* is made:

```
8D45 F0      LEA EAX,DWORD PTR SS:[EBP-10] ; POEM_STRING
6A 00        PUSH 0                          ; 0 = Do NOT Allocate
50          PUSH EAX
8D45 F8      LEA EAX,DWORD PTR SS:[EBP-8]   ; PUNICODE_STRING
50          PUSH EAX
FF15 54103477 CALL ntdll.RtlOemStringToUnicodeString
```

The call to *ntdll.RtlOemStringToUnicodeString* specifies that the buffer is already allocated by the caller. Therefore, this function will simply translate the supplied *POEM\_STRING->Buffer*, storing the resulting characters in *PUNICODE\_STRING->Buffer*, having done no bound checking.

This introduces the possibility of overflowing the stack space reserved for this buffer and gain control of execution modifying the SEH handler.

#### **Impact:**

The *dhcpcsvc.dll* is loaded by *services.exe* in Windows 2000 and by *svchost.exe* in Windows XP/2003. As both these services run with SYSTEM privileges, an exploitation of this issue would allow Remote Code Execution as SYSTEM over vulnerable systems.

#### **Solutions:**

Microsoft has referred this vulnerability as MS06-036 and developed a patch to address it. Users should upgrade their system through Microsoft Windows Update facility.

More information can be found in Microsoft KB914388.

**Vendor Response:**

- 12/26/2005: Initial Vendor Contact.
- 01/19/2006: Vendor Confirmed Vulnerability.
- 07/11/2006: Vendor Releases Update.
- 07/11/2006: Pre-Advisory Public Disclosure.
- 08/29/2006: Advisory Public Disclosure.

**Thanks:**

Special thanks go to Andres Riancho.

**Contact Information:**

For more information regarding the vulnerability feel free to contact the author at [mnunez {at} cybsec.com](mailto:mnunez@cybsec.com).

For more information regarding CYBSEC: [www.cybsec.com](http://www.cybsec.com)