Windows XP SP3 Kernel Registry Handling Denial of Service

by Matthew "j00ru" Jurczyk and Gynvael Coldwind
Hispasec

1. Basic Information

Name	Windows XP SP3 Kernel Registry Handling Denial of Service
Class	Implementation error
Impact	Medium
Credits	Matthew "j00ru" Jurczyk, Gynvael Coldwind
Discovered	2008-12-15
Published	2010-05-29

2. Abstract

Microsoft Windows XP is a commonly used desktop operating system, released with Service Pack 3 at the time of writing this paper.
The vulnerable system component is heart of Windows NT family – the *ntoskrnl.exe* kernel executable itself (other variations of this executable, e.g. *ntoskrnlpa.exe* are also affected). It is responsible for performing nearly all the critical system operations – most of the requests come from user-mode applications using the SYSENTER mechanism. System call set describes the functions available for a low-level application programmer – if one finds a denial-of-service vulnerability in a syscall handler, it is very likely he will be able to completely destroy the system's workspace and cause a Blue Screen of Death followed by a hard machine reboot.

The vulnerability covered in this paper relies on the fact that some very internal kernel code makes unsafe assumptions about the registry contents. The situation causing a system crash would actually never occur in real life, though it is still possible to manually trigger the necessary conditions and take advantage of the fact that the high-level code doesn't fully validate the registry structures being operated on.

Every Windows NT version prior to Windows Vista is affected, including Windows XP SP3 with latest patches.

3. Vulnerability details

The vulnerability is strongly related to the registry "symbolic link" mechanism provided by Microsoft. It is commonly used to create invisible transitions between separate keys, thought the vendor itself does not provide any technical document about neither how could a normal programmer use it in his software nor how does it work internally. This is probably because Microsoft wants to keep this feature for his own purposes and not let people mess too much with the Windows Registry.

Despite presenting such approach, nothing has been done to prevent a restricted user's applications access these extra features and take advantage of unknown possibilities that haven't yet been publicly explained. The "symbolic link" functionality creates real enormous possibilities when it comes to vulnerability research. In this paper, I will show how to crash the newest Windows XP version using the mechanism in a way that the developers probably haven't ever thought of.

Since the link keys don't really differ from the standard ones, apart from a KEY_LINK flag set in the REGF file, they can also contain value records. To be specific, the only allowed value is *SymbolicLinkValue* of the REG_LINK type. This value does not possess its own structure – it is a plain UNICODE string, and could be marked as REG_SZ if it wasn't a part of the linking mechanism. As far as I know, this kind of values is the only one being referenced by the kernel itself. If we look deeper into the symbolic link implementation, curious things might appear.

A well known fact is that one standard Unicode character is supposed to be 2-byte long, but when setting the registry value, its size is not validated in any way. What is more, when referencing such a key, the internal hashing routine assumes the length is divisible by sizeof(WCHAR):

```
Count = user controlled;
while (Count!=0)
{
  /* … */
  Count -= sizeof(WCHAR);
}
```

(*vulnerable function pseudocode*)

As one can see, the loop condition works fine only for valid Length field – in case of the string size being an odd value, the loop will not end its execution till an Invalid Memory Access exception is generated. This leads us straight to a BSoD and a system crash. To sum everything up, the only thing to make the machine go down is to create a symbolic link of the 2n+1 length and reference it somehow (basically using *RegOpenKeyEx* etc.)

4. Impact

The vunerability allows an attacker to crash a local machine, provided he can access an active user account and launch an application in its context. No special user privileges are required to perform the attack, as every user is allowed to create keys marked as

“symbolic links” and operate on their values. The impact of a single attack, considering the above conditions, is rated as medium.

5. Disclaimer