

[UNIX] Linux Kernel do_mremap Local Privilege Escalation Vulnerability (Technical Details)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2004-01/0054.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 01/15/04

To: list@securiteam.com

Date: 15 Jan 2004 18:09:40 +0200

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

Linux Kernel do_mremap Local Privilege Escalation Vulnerability (Technical Details)

SUMMARY

A critical security vulnerability has been found in the Linux kernel memory management code in `mremap(2)` system call due to incorrect bound checks. The following advisory will try to shed more light into the issue, and provide a more accurate means of testing this vulnerability (via an exploit).

DETAILS

Vulnerable systems:

* Linux kernel 2.4 up to 2.4.23 and 2.6.0

The `mremap` system call provides functionality of resizing (shrinking or growing) as well as moving across process's addressable space of existing virtual memory areas (VMAs) or any of its parts.

A typical VMA covers at least one memory page (which is exactly 4kB on the i386 architecture). An incorrect bound check discovered inside the `do_mremap()` kernel code performing remapping of a virtual memory area may

Securiteam: [UNIX] Linux Kernel do_mremap Local Privilege Escalation Vulnerability (Technical Details)

lead to creation of a virtual memory area of 0 bytes in length.

The problem bases on the general mremap flaw that remapping of 2 pages from inside a VMA creates a memory hole of only one page in length but also an additional VMA of two pages. In the case of a zero sized remapping request no VMA hole is created but an additional VMA descriptor of 0 bytes in length is created.

Such a malicious virtual memory area may disrupt the operation of the other parts of the kernel memory management subroutines finally leading to unexpected behavior.

A typical process's memory layout showing invalid VMA created with mremap system call:

```
08048000-0804c000 r-xp 00000000 03:05 959142 /tmp/test
0804c000-0804d000 rw-p 00003000 03:05 959142 /tmp/test
0804d000-0804e000 rwxp 00000000 00:00 0
40000000-40014000 r-xp 00000000 03:05 1544523 /lib/ld-2.3.2.so
```