

[UNIX] Linux Kernel do_mremap VMA Limit Local Privilege Escalation Vulnerability

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2004-02/0051.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 02/18/04

To: list@securiteam.com

Date: 18 Feb 2004 16:37:06 +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 VMA Limit Local Privilege Escalation Vulnerability

SUMMARY

A critical security vulnerability has been found in the Linux kernel memory management code inside the `mremap(2)` system call due to missing function return value check. This bug is completely unrelated to the `mremap` bug disclosed on 05-01-2004 except concerning the same internal kernel function code.

DETAILS

Vulnerable Systems:

- * Linux version 2.2 up to 2.2.25
- * Linux version 2.4 up to 2.4.24
- * Linux version 2.6 up to 2.6.2

The Linux kernel manages a list of user addressable valid memory locations on a per process basis. Every process owns a single linked list of so called virtual memory area descriptors (called from now on just VMAs). Every VMA describes the start of a valid memory region, its length and moreover various memory flags like page protection.

Every VMA in the list corresponds to a part of the process's page table. The page table contains descriptors (in short page table entries PTEs) of physical memory pages seen by the process. The VMA descriptor can be thus understood as a high level description of a particular region of the process's page table storing PTE properties like page R/W flag and so on.

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

Moving a part of the virtual memory from inside a VMA area to a new location requires creation of a new VMA descriptor as well as copying the underlying page table entries described by the VMA from the old to the new location in the process's page table.

To accomplish this task the do_mremap code calls the do_munmap() internal kernel function to remove any potentially existing old memory mapping in the new location as well as to remove the old virtual memory mapping. Unfortunately the code doesn't test the return value of the do_munmap() function which may fail if the maximum number of available VMA descriptors has been exceeded. This happens if one tries to unmap middle part of an existing memory mapping and the process's limit on the number of VMAs has been reached (which is currently 65535).

One of the possible situations can be illustrated with the following picture. The corresponding page table entries (PTEs) have been marked with o and x:

Before mremap():
(oooooooooooooooooooooooooooo) (xxxxxxxxxxxxxx)
[-----VMA1-----] [----VMA2----]
 [REMAPPED-VMA] < -----|

After mremap() without VMA limit:
(oooo)(xxxxxxxxxxxxxx)(oooo)
[VMA3][REMAPPED-VMA][VMA4]

After mremap() but VMA limit:
(oooooooooooooooooooooooooooo)
[-----VMA1-----]
 [REMAPPED-VMA]

After the maximum number of VMAs in the process's VMA list has been reached do_munmap() will refuse to create the necessary VMA hole because it would split the original VMA in two disjoint VMA areas exceeding the VMA descriptor limit.

Due to the missing return value check after trying to unmap the middle of the VMA1 (this is the first invocation of do_munmap inside do_mremap code) the corresponding page table entries from VMA2 are still inserted into the page table location described by VMA1 thus being subject to VMA1 page

Securiteam: [UNIX] Linux Kernel do_mremap VMA Limit Local Privilege Escalation Vulnerability

protection flags. It must be also mentioned that the original PTEs in the VMA1 are lost thus leaving the corresponding page frames unusable for ever.

The kernel also tries to insert the overlapping VMA area into the VMA descriptor list but this fails due to further checks in the low level VMA manipulation code. The low level VMA list check in the 2.4 and 2.6 kernel versions just call BUG() therefore terminating the malicious process.

There are also two other unchecked calls to do_munmap() inside the do_mremap() code and we believe that the second occurrence of unchecked do_munmap is also exploitable. The second occurrence takes place if the VMA to be remapped is being truncated in place. Note that do_munmap can also fail on an exceptional low memory condition while trying to allocate a VMA descriptor.

We were able to create a robust proof-of-concept exploit code giving full super-user privileges on all vulnerable kernel versions. The exploit code will be released next week.

Impact:

Since no special privileges are required to use the mremap(2) system call any process may use its unexpected behavior to disrupt the kernel memory management subsystem.

Proper exploitation of this vulnerability leads to local privilege escalation giving an attacker full super-user privileges. The vulnerability may also lead to a denial-of-service attack on the available system memory.

Tested and known to be vulnerable kernel versions are all $\leq 2.2.25$, $\leq 2.4.24$ and $\leq 2.6.1$. The 2.2.25 version of Linux kernel does not recognize the MREMAP_FIXED flag but this does not prevent the bug from being successfully exploited. All users are encouraged to patch all vulnerable systems as soon as appropriate vendor patches are released. There is no hotfix for this vulnerability. Limited per user virtual memory still permits do_munmap() to fail.

ADDITIONAL INFORMATION

The information has been provided by <mailto:ihaquer@isec.pl> Paul Starzetz.

The original article can be found at:

<<http://isec.pl/vulnerabilities/isec-0014-mremap-unmap.txt>>
<http://isec.pl/vulnerabilities/isec-0014-mremap-unmap.txt>

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:

Securiteam: [UNIX] Linux Kernel do_mremap VMA Limit Local Privilege Escalation Vulnerability

list-unsubscribe@securiteam.com

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====
=====

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.