Subject: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Kirill Korotaev on Mon, 04 Sep 2006 12:14:20 GMT
View Forum Message <> Reply to Message

Linus,

this patch is already commited into -stable 2.6.17.y tree.
 http://www.kernel.org/git/?p=linux/kernel/git/stable/linux-2
.6.17.y.git;a=commit;h=8833ebaa3f4325820fe3338ccf6fae04f6669254

I only incorporated a small compilation fix from
Fernando Vazquez <fernando@oss.ntt.co.jp>.
please, commit it into next 2.6.18-rcX.

local DoS with corrupted ELFs

This patch prevents cross-region mappings
on IA64 and SPARC which could lead to system crash.

davem@ confirmed: "This looks fine to me." :)

Signed-Off-By: Pavel Emelianov <xemul@openvz.org>
Signed-Off-By: Kirill Korotaev <dev@openvz.org>
Signed-off-by: Greg Kroah-Hartman <gregkh@suse.de>

```
diff --git a/arch/ia64/kernel/sys_ia64.c b/arch/ia64/kernel/sys_ia64.c
index 40722d8..9ef62a3 100644
--- a/arch/ia64/kernel/sys_ia64.c
+++ b/arch/ia64/kernel/sys_ia64.c
@@ -163,10 +163,25 @@ sys_pipe (void)
  return retval;
 }

+int ia64_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags)
+{
+ unsigned long roff;
+
+ /*
+  * Don't permit mappings into unmapped space, the virtual page table
+  * of a region, or across a region boundary.  Note: RGN_MAP_LIMIT is
+  * equal to 2^n-PAGE_SIZE (for some integer n <= 61) and len > 0.
+  */
+ roff = REGION_OFFSET(addr);
```

```
+ if ((len > RGN_MAP_LIMIT) || (roff > (RGN_MAP_LIMIT - len)))
+  return -EINVAL;
+ return 0;
+}
+
 static inline unsigned long
 do_mmap2 (unsigned long addr, unsigned long len, int prot, int flags, int fd, unsigned long pgoff)
 {
- unsigned long roff;
  struct file *file = NULL;

  flags &= ~(MAP_EXECUTABLE | MAP_DENYWRITE);
@@ -188,17 +203,6 @@ do_mmap2 (unsigned long addr, unsigned l
   goto out;
  }

- /*
-  * Don't permit mappings into unmapped space, the virtual page table of a region,
-  * or across a region boundary.  Note: RGN_MAP_LIMIT is equal to 2^n-PAGE_SIZE
-  * (for some integer n <= 61) and len > 0.
-  */
- roff = REGION_OFFSET(addr);
- if ((len > RGN_MAP_LIMIT) || (roff > (RGN_MAP_LIMIT - len))) {
-  addr = -EINVAL;
-  goto out;
- }
-
  down_write(&current->mm->mmap_sem);
  addr = do_mmap_pgoff(file, addr, len, prot, flags, pgoff);
  up_write(&current->mm->mmap_sem);
diff --git a/arch/sparc/kernel/sys_sparc.c b/arch/sparc/kernel/sys_sparc.c
index a41c8a5..94ff58c 100644
--- a/arch/sparc/kernel/sys_sparc.c
+++ b/arch/sparc/kernel/sys_sparc.c
@@ -219,6 +219,21 @@ out:
  return err;
 }

+int sparc_mmap_check(unsigned long addr, unsigned long len, unsigned long flags)
+{
+ if (ARCH_SUN4C_SUN4 &&
+     (len > 0x20000000 ||
+      ((flags & MAP_FIXED) &&
+       addr < 0xe0000000 && addr + len > 0x20000000)))
+  return -EINVAL;
+
+ /* See asm-sparc/uaccess.h */
+ if (len > TASK_SIZE - PAGE_SIZE || addr + len > TASK_SIZE - PAGE_SIZE)
```

```
+  return -EINVAL;
+
+ return 0;
+}
+
 /* Linux version of mmap */
 static unsigned long do_mmap2(unsigned long addr, unsigned long len,
  unsigned long prot, unsigned long flags, unsigned long fd,
@@ -233,25 +248,13 @@ static unsigned long do_mmap2(unsigned l
    goto out;
  }

- retval = -EINVAL;
  len = PAGE_ALIGN(len);
- if (ARCH_SUN4C_SUN4 &&
-     (len > 0x20000000 ||
-      ((flags & MAP_FIXED) &&
-       addr < 0xe0000000 && addr + len > 0x20000000)))
-  goto out_putf;
-
- /* See asm-sparc/uaccess.h */
- if (len > TASK_SIZE - PAGE_SIZE || addr + len > TASK_SIZE - PAGE_SIZE)
-  goto out_putf;
-
  flags &= ~(MAP_EXECUTABLE | MAP_DENYWRITE);

  down_write(&current->mm->mmap_sem);
  retval = do_mmap_pgoff(file, addr, len, prot, flags, pgoff);
  up_write(&current->mm->mmap_sem);

-out_putf:
  if (file)
   fput(file);
 out:
diff --git a/arch/sparc64/kernel/sys_sparc.c b/arch/sparc64/kernel/sys_sparc.c
index 054d0ab..bf5f14e 100644
--- a/arch/sparc64/kernel/sys_sparc.c
+++ b/arch/sparc64/kernel/sys_sparc.c
@@ -548,6 +548,26 @@ asmlinkage long sparc64_personality(unsi
  return ret;
 }

+int sparc64_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags)
+{
+ if (test_thread_flag(TIF_32BIT)) {
+  if (len >= STACK_TOP32)
+   return -EINVAL;
```

```
+
+  if ((flags & MAP_FIXED) && addr > STACK_TOP32 - len)
+   return -EINVAL;
+ } else {
+  if (len >= VA_EXCLUDE_START)
+   return -EINVAL;
+
+  if ((flags & MAP_FIXED) && invalid_64bit_range(addr, len))
+   return -EINVAL;
+ }
+
+ return 0;
+}
+
 /* Linux version of mmap */
 asmlinkage unsigned long sys_mmap(unsigned long addr, unsigned long len,
  unsigned long prot, unsigned long flags, unsigned long fd,
@@ -563,27 +583,11 @@ asmlinkage unsigned long sys_mmap(unsign
  }
  flags &= ~(MAP_EXECUTABLE | MAP_DENYWRITE);
  len = PAGE_ALIGN(len);
- retval = -EINVAL;
-
- if (test_thread_flag(TIF_32BIT)) {
-  if (len >= STACK_TOP32)
-   goto out_putf;
-
-  if ((flags & MAP_FIXED) && addr > STACK_TOP32 - len)
-   goto out_putf;
- } else {
-  if (len >= VA_EXCLUDE_START)
-   goto out_putf;
-
-  if ((flags & MAP_FIXED) && invalid_64bit_range(addr, len))
-   goto out_putf;
- }

  down_write(&current->mm->mmap_sem);
  retval = do_mmap(file, addr, len, prot, flags, off);
  up_write(&current->mm->mmap_sem);

-out_putf:
  if (file)
   fput(file);
 out:
diff --git a/include/asm-generic/mman.h b/include/asm-generic/mman.h
index 3b41d2b..010ced7 100644
--- a/include/asm-generic/mman.h
```

```
+++ b/include/asm-generic/mman.h
@@ -39,4 +39,10 @@ #define MADV_DOFORK 11  /* do inherit ac
 #define MAP_ANON MAP_ANONYMOUS
 #define MAP_FILE 0

+#ifdef __KERNEL__
+#ifndef arch_mmap_check
+#define arch_mmap_check(addr, len, flags) (0)
+#endif
+#endif
+
 #endif
diff --git a/include/asm-ia64/mman.h b/include/asm-ia64/mman.h
index 6ba179f..a42a3e6 100644
--- a/include/asm-ia64/mman.h
+++ b/include/asm-ia64/mman.h
@@ -8,6 +8,14 @@ #define _ASM_IA64_MMAN_H
  * David Mosberger-Tang <davidm@hpl.hp.com>, Hewlett-Packard Co
  */

+#ifdef __KERNEL__
+#define arch_mmap_check ia64_mmap_check
+#ifndef __ASSEMBLY__
+int ia64_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags);
+#endif
+#endif
+
 #include <asm-generic/mman.h>

 #define MAP_GROWSDOWN 0x00100  /* stack-like segment */
diff --git a/include/asm-sparc/mman.h b/include/asm-sparc/mman.h
index 88d1886..cf8e002 100644
--- a/include/asm-sparc/mman.h
+++ b/include/asm-sparc/mman.h
@@ -2,6 +2,14 @@
 #ifndef __SPARC_MMAN_H__
 #define __SPARC_MMAN_H__

+#ifdef __KERNEL__
+#define arch_mmap_check sparc_mmap_check
+#ifndef __ASSEMBLY__
+int sparc_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags);
+#endif
+#endif
+
 #include <asm-generic/mman.h>
```

```
 /* SunOS'ified... */
diff --git a/include/asm-sparc64/mman.h b/include/asm-sparc64/mman.h
index 6fd878e..900e2ac 100644
--- a/include/asm-sparc64/mman.h
+++ b/include/asm-sparc64/mman.h
@@ -2,6 +2,14 @@
 #ifndef __SPARC64_MMAN_H__
 #define __SPARC64_MMAN_H__

+#ifdef __KERNEL__
+#define arch_mmap_check sparc64_mmap_check
+#ifndef __ASSEMBLY__
+int sparc64_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags);
+#endif
+#endif
+
 #include <asm-generic/mman.h>

 /* SunOS'ified... */
diff --git a/mm/mmap.c b/mm/mmap.c
index c1868ec..02ecef8 100644
--- a/mm/mmap.c
+++ b/mm/mmap.c
@@ -913,6 +913,10 @@ unsigned long do_mmap_pgoff(struct file
  if (!len)
   return -EINVAL;

+ error = arch_mmap_check(addr, len, flags);
+ if (error)
+  return error;
+
  /* Careful about overflows.. */
  len = PAGE_ALIGN(len);
  if (!len || len > TASK_SIZE)
@@ -1859,6 +1863,7 @@ unsigned long do_brk(unsigned long addr,
  unsigned long flags;
  struct rb_node ** rb_link, * rb_parent;
  pgoff_t pgoff = addr >> PAGE_SHIFT;
+ int error;

  len = PAGE_ALIGN(len);
  if (!len)
@@ -1867,6 +1872,12 @@ unsigned long do_brk(unsigned long addr,
  if ((addr + len) > TASK_SIZE || (addr + len) < addr)
   return -EINVAL;
```

```
+ flags = VM_DATA_DEFAULT_FLAGS | VM_ACCOUNT | mm->def_flags;
+
+ error = arch_mmap_check(addr, len, flags);
+ if (error)
+  return error;
+
  /*
   * mlock MCL_FUTURE?
   */
@@ -1907,8 +1918,6 @@ unsigned long do_brk(unsigned long addr,
  if (security_vm_enough_memory(len >> PAGE_SHIFT))
   return -ENOMEM;

- flags = VM_DATA_DEFAULT_FLAGS | VM_ACCOUNT | mm->def_flags;
-
  /* Can we just expand an old private anonymous mapping? */
  if (vma_merge(mm, prev, addr, addr + len, flags,
      NULL, NULL, pgoff, NULL))
```

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Kyle McMartin on Tue, 05 Sep 2006 11:39:39 GMT
View Forum Message <> Reply to Message

On Mon, Sep 04, 2006 at 04:17:00PM +0400, Kirill Korotaev wrote:
> --- a/include/asm-generic/mman.h
> +++ b/include/asm-generic/mman.h
> @@ -39,4 +39,10 @@ #define MADV_DOFORK 11 /* do inherit ac
> #define MAP_ANON MAP_ANONYMOUS
> #define MAP_FILE 0
>
> +#ifdef __KERNEL__
> +#ifndef arch_mmap_check
> +#define arch_mmap_check(addr, len, flags) (0)
> +#endif
> +#endif
> +
> #endif

This breaks all arches that don't use asm-generic/mman.h, and that you
didn't add arch_mmap_check to asm/mman.h for.

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by dev on Tue, 05 Sep 2006 13:06:57 GMT
View Forum Message <> Reply to Message

Andrew, Kyle,

> On Mon, Sep 04, 2006 at 04:17:00PM +0400, Kirill Korotaev wrote:
>
>>--- a/include/asm-generic/mman.h
>>+++ b/include/asm-generic/mman.h
>>@@ -39,4 +39,10 @@ #define MADV_DOFORK 11 /* do inherit ac
>>#define MAP_ANON MAP_ANONYMOUS
>>#define MAP_FILE 0
>>
>>+#ifdef __KERNEL__
>>+#ifndef arch_mmap_check
>>+#define arch_mmap_check(addr, len, flags) (0)
>>+#endif
>>+#endif
>>+
>>#endif
>
>
> This breaks all arches that don't use asm-generic/mman.h, and that you
> didn't add arch_mmap_check to asm/mman.h for.

oops... You are right.

is define
#define arch_mmap_check(addr, len, flags) (0)
ok for you in such mman.h headers which do not include asm-generic/mman.h?
If yes, the following patch should help.

Though I didn't get the idea of include/asm-um/mman.h:
#include "asm/arch/mman.h"

This patch adds define of arch_mmap_check() to
archs which do not include include/asm-generic/mman.h

---

diff --git a/include/asm-alpha/mman.h b/include/asm-alpha/mman.h
index 5f24c75..51cf354 100644
--- a/include/asm-alpha/mman.h
+++ b/include/asm-alpha/mman.h
@@ -52,4 +52,10 @@ #define MADV_DOFORK 11 /* do inherit ac
 #define MAP_ANON MAP_ANONYMOUS
 #define MAP_FILE 0

+#ifdef __KERNEL__
+#ifndef arch_mmap_check
+#define arch_mmap_check(addr, len, flags) (0)

```
+#endif
+#endif
+
 #endif /* __ALPHA_MMAN_H__ */
diff --git a/include/asm-mips/mman.h b/include/asm-mips/mman.h
index 046cf68..f19e858 100644
--- a/include/asm-mips/mman.h
+++ b/include/asm-mips/mman.h
@@ -75,4 +75,10 @@ #define MADV_DOFORK 11  /* do inherit ac
 #define MAP_ANON MAP_ANONYMOUS
 #define MAP_FILE 0

+#ifdef __KERNEL__
+#ifndef arch_mmap_check
+#define arch_mmap_check(addr, len, flags) (0)
+#endif
+#endif
+
 #endif /* _ASM_MMAN_H */
diff --git a/include/asm-parisc/mman.h b/include/asm-parisc/mman.h
index 0ef15ee..9829b31 100644
--- a/include/asm-parisc/mman.h
+++ b/include/asm-parisc/mman.h
@@ -59,4 +59,10 @@ #define MAP_ANON MAP_ANONYMOUS
 #define MAP_FILE 0
 #define MAP_VARIABLE 0

+#ifdef __KERNEL__
+#ifndef arch_mmap_check
+#define arch_mmap_check(addr, len, flags) (0)
+#endif
+#endif
+
 #endif /* __PARISC_MMAN_H__ */
```

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Linus Torvalds on Wed, 06 Sep 2006 18:24:05 GMT
View Forum Message <> Reply to Message

On Mon, 4 Sep 2006, Kirill Korotaev wrote:
>
> this patch is already commited into -stable 2.6.17.y tree.

I don't like it. Apparently the patch was bad, and broken on MIPS and parisc, and it was applied to the stable tree without being in the standard tree.

If MIPS and parisc don't matter for the stable tree (very possible - there
are no big commercial distributions for them), then dammit, neither should
ia64 and sparc (there are no big commercial distros for them either).
Either way, it seems this didn't happen the way it should have.

The proper fix would _seem_ to have the whole

```
#ifndef arch_mmap_check
#define arch_mmap_check(addr, len, flags) (0)
#endif
```

in the only file that actually _uses_ this, namely mm/mmap.c. Rather than
pollute lots of architecture-specific files with this macro that nobody
really is interested in except for ia64 and sparc.

Linus

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Matthew Wilcox on Wed, 06 Sep 2006 18:27:33 GMT
View Forum Message <> Reply to Message

On Wed, Sep 06, 2006 at 11:24:05AM -0700, Linus Torvalds wrote:
> If MIPS and parisc don't matter for the stable tree (very possible - there
> are no big commercial distributions for them), then dammit, neither should
> ia64 and sparc (there are no big commercial distros for them either).

Erm, RHEL and SLES both support ia64.

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Arjan van de Ven on Wed, 06 Sep 2006 19:06:19 GMT
View Forum Message <> Reply to Message

On Wed, 2006-09-06 at 12:27 -0600, Matthew Wilcox wrote:
> On Wed, Sep 06, 2006 at 11:24:05AM -0700, Linus Torvalds wrote:
> > If MIPS and parisc don't matter for the stable tree (very possible - there
> > are no big commercial distributions for them), then dammit, neither should
> > ia64 and sparc (there are no big commercial distros for them either).
>
> Erm, RHEL and SLES both support ia64.

but neither use -stable.

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs

Posted by Linus Torvalds on Wed, 06 Sep 2006 20:20:59 GMT
View Forum Message <> Reply to Message

On Mon, 4 Sep 2006, Kirill Korotaev wrote:
>
> +#ifdef __KERNEL__
> +#define arch_mmap_check ia64_mmap_check
> +#ifndef __ASSEMBLY__
> +int ia64_mmap_check(unsigned long addr, unsigned long len,
> + unsigned long flags);
> +#endif
> +#endif

Btw, is there some reason for the __ASSEMBLY__ check?

I'm not seeing any kernel users that could care, a quick

 git grep 'mman\.h' -- '*.[sS]'

doesn't trigger anything, and the other header files that include this
seem to all either be mman.h themselves, or have things like structure
declarations etc that wouldn't work for any non-C source anyway.

But maybe I missed some.

I'd rather not have more of those '#ifndef __ASSEMBLY__' than necessary.

  Linus

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by tony.luck on Wed, 06 Sep 2006 21:27:54 GMT
View Forum Message <> Reply to Message

On Wed, Sep 06, 2006 at 01:20:59PM -0700, Linus Torvalds wrote:
> Btw, is there some reason for the __ASSEMBLY__ check?

On ia64 entry.S includes asm/pgtable.h, which includes asm/mman.h

-Tony

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by fernando on Wed, 06 Sep 2006 23:23:11 GMT
View Forum Message <> Reply to Message

On Wed, 2006-09-06 at 13:20 -0700, Linus Torvalds wrote:
>

> On Mon, 4 Sep 2006, Kirill Korotaev wrote:
> >
> > +#ifdef __KERNEL__
> > +#define arch_mmap_check ia64_mmap_check
> > +#ifndef __ASSEMBLY__
> > +int ia64_mmap_check(unsigned long addr, unsigned long len,
> > + unsigned long flags);
> > +#endif
> > +#endif
>
> Btw, is there some reason for the __ASSEMBLY__ check?
>
> I'm not seeing any kernel users that could care, a quick
>
> git grep 'mman\.h' -- '*.[sS]'
>
> doesn't trigger anything, and the other header files that include this
> seem to all either be mman.h themselves, or have things like structure
> declarations etc that wouldn't work for any non-C source anyway.
>
> But maybe I missed some.
>
> I'd rather not have more of those '#ifndef __ASSEMBLY__' than necessary

The problem is that "asm/mman.h" is being included from entry.S indirectly
through "asm/pgtable.h" (see code snips below).

* arch/ia64/kernel/entry.S:
...
#include <asm/pgtable.h>
...

* include/asm-ia64/pgtable.h:
...
#include <asm/mman.h>
...

* include/asm-ia64/mman.h
...
#ifdef __KERNEL__
#define arch_mmap_check ia64_map_check_rgn
int ia64_map_check_rgn(unsigned long addr, unsigned long len,
		unsigned long flags);
#endif
...

Without this fix compilation is broken:

```
  gcc -Wp,-MD,arch/ia64/kernel/.entry.o.d  -nostdinc -isystem
/usr/lib/gcc/ia64-linux-gnu/4.1.2/include -D__KERNEL__ -Iinclude  -include
include/linux/autoconf.h -DHAVE_WORKING_TEXT_ALIGN
-DHAVE_MODEL_SMALL_ATTRIBUTE -DHAVE_SERIALIZE_DIRECTIVE -D__ASSEMBLY__
-mconstant-gp -c -o arch/ia64/kernel/entry.o arch/ia64/kernel/entry.S
include/asm/mman.h: Assembler messages:
include/asm/mman.h:13: Error: Unknown opcode `int ia64_map_check_rgn(unsigned long
addr,unsigned long len,'
include/asm/mman.h:14: Error: Unknown opcode `unsigned long flags)'
make[1]: *** [arch/ia64/kernel/entry.o] Error 1
make: *** [arch/ia64/kernel] Error 2
```

Regards,

Fernando

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Kirill Korotaev on Thu, 07 Sep 2006 10:14:02 GMT
View Forum Message <> Reply to Message

> On Mon, 4 Sep 2006, Kirill Korotaev wrote:
>>this patch is already commited into -stable 2.6.17.y tree.
>
>
> I don't like it. Apparently the patch was bad, and broken on MIPS and
> parisc, and it was applied to the stable tree without being in the
> standard tree.
it is locally exploitable DoS IMHO.
I'm not sure what is the policy about security fixes, since Tony Luck didn't
show much interest in this fix and I kept pushing it myself.
Probably I should have published an exploit to make it moving direct way :)

And I really sorry for patch sent being broken a bit :/

> If MIPS and parisc don't matter for the stable tree (very possible - there
> are no big commercial distributions for them), then dammit, neither should
> ia64 and sparc (there are no big commercial distros for them either).
> Either way, it seems this didn't happen the way it should have.
>
> The proper fix would _seem_ to have the whole
>
> #ifndef arch_mmap_check
> #define arch_mmap_check(addr, len, flags) (0)
> #endif
>
> in the only file that actually _uses_ this, namely mm/mmap.c. Rather than
> pollute lots of architecture-specific files with this macro that nobody

> really is interested in except for ia64 and sparc.

Does the patch below looks better?
(compile tested on IA64)

```
diff --git a/arch/ia64/kernel/sys_ia64.c b/arch/ia64/kernel/sys_ia64.c
index 40722d8..5b190c9 100644
--- a/arch/ia64/kernel/sys_ia64.c
+++ b/arch/ia64/kernel/sys_ia64.c
@@ -163,10 +163,25 @@ sys_pipe (void)
  return retval;
 }

+int ia64_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags)
+{
+ unsigned long roff;
+
+ /*
+  * Don't permit mappings into unmapped space, the virtual page table
+  * of a region, or across a region boundary.  Note: RGN_MAP_LIMIT is
+  * equal to 2^n-PAGE_SIZE (for some integer n <= 61) and len > 0.
+  */
+ roff = REGION_OFFSET(addr);
+ if ((len > RGN_MAP_LIMIT) || (roff > (RGN_MAP_LIMIT - len)))
+  return -EINVAL;
+ return 0;
+}
+
 static inline unsigned long
 do_mmap2 (unsigned long addr, unsigned long len, int prot, int flags, int fd, unsigned long pgoff)
 {
- unsigned long roff;
  struct file *file = NULL;

  flags &= ~(MAP_EXECUTABLE | MAP_DENYWRITE);
@@ -188,17 +203,6 @@ do_mmap2 (unsigned long addr, unsigned l
   goto out;
  }

- /*
-  * Don't permit mappings into unmapped space, the virtual page table of a region,
-  * or across a region boundary.  Note: RGN_MAP_LIMIT is equal to 2^n-PAGE_SIZE
-  * (for some integer n <= 61) and len > 0.
-  */
- roff = REGION_OFFSET(addr);
```

```
- if ((len > RGN_MAP_LIMIT) || (roff > (RGN_MAP_LIMIT - len))) {
-  addr = -EINVAL;
-  goto out;
- }
-
  down_write(&current->mm->mmap_sem);
  addr = do_mmap_pgoff(file, addr, len, prot, flags, pgoff);
  up_write(&current->mm->mmap_sem);
diff --git a/arch/sparc/kernel/sys_sparc.c b/arch/sparc/kernel/sys_sparc.c
index a41c8a5..94ff58c 100644
--- a/arch/sparc/kernel/sys_sparc.c
+++ b/arch/sparc/kernel/sys_sparc.c
@@ -219,6 +219,21 @@ out:
  return err;
 }

+int sparc_mmap_check(unsigned long addr, unsigned long len, unsigned long flags)
+{
+ if (ARCH_SUN4C_SUN4 &&
+     (len > 0x20000000 ||
+      ((flags & MAP_FIXED) &&
+       addr < 0xe0000000 && addr + len > 0x20000000)))
+  return -EINVAL;
+
+ /* See asm-sparc/uaccess.h */
+ if (len > TASK_SIZE - PAGE_SIZE || addr + len > TASK_SIZE - PAGE_SIZE)
+  return -EINVAL;
+
+ return 0;
+}
+
 /* Linux version of mmap */
 static unsigned long do_mmap2(unsigned long addr, unsigned long len,
  unsigned long prot, unsigned long flags, unsigned long fd,
@@ -233,25 +248,13 @@ static unsigned long do_mmap2(unsigned l
    goto out;
  }

- retval = -EINVAL;
  len = PAGE_ALIGN(len);
- if (ARCH_SUN4C_SUN4 &&
-     (len > 0x20000000 ||
-      ((flags & MAP_FIXED) &&
-       addr < 0xe0000000 && addr + len > 0x20000000)))
-  goto out_putf;
-
- /* See asm-sparc/uaccess.h */
- if (len > TASK_SIZE - PAGE_SIZE || addr + len > TASK_SIZE - PAGE_SIZE)
```

```
-  goto out_putf;
-
  flags &= ~(MAP_EXECUTABLE | MAP_DENYWRITE);

  down_write(&current->mm->mmap_sem);
  retval = do_mmap_pgoff(file, addr, len, prot, flags, pgoff);
  up_write(&current->mm->mmap_sem);

-out_putf:
  if (file)
   fput(file);
 out:
diff --git a/arch/sparc64/kernel/sys_sparc.c b/arch/sparc64/kernel/sys_sparc.c
index 054d0ab..bf5f14e 100644
--- a/arch/sparc64/kernel/sys_sparc.c
+++ b/arch/sparc64/kernel/sys_sparc.c
@@ -548,6 +548,26 @@ asmlinkage long sparc64_personality(unsi
  return ret;
 }

+int sparc64_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags)
+{
+ if (test_thread_flag(TIF_32BIT)) {
+  if (len >= STACK_TOP32)
+   return -EINVAL;
+
+  if ((flags & MAP_FIXED) && addr > STACK_TOP32 - len)
+   return -EINVAL;
+ } else {
+  if (len >= VA_EXCLUDE_START)
+   return -EINVAL;
+
+  if ((flags & MAP_FIXED) && invalid_64bit_range(addr, len))
+   return -EINVAL;
+ }
+
+ return 0;
+}
+
 /* Linux version of mmap */
 asmlinkage unsigned long sys_mmap(unsigned long addr, unsigned long len,
  unsigned long prot, unsigned long flags, unsigned long fd,
@@ -563,27 +583,11 @@ asmlinkage unsigned long sys_mmap(unsign
  }
  flags &= ~(MAP_EXECUTABLE | MAP_DENYWRITE);
  len = PAGE_ALIGN(len);
- retval = -EINVAL;
```

```
-
- if (test_thread_flag(TIF_32BIT)) {
-  if (len >= STACK_TOP32)
-   goto out_putf;
-
-  if ((flags & MAP_FIXED) && addr > STACK_TOP32 - len)
-   goto out_putf;
- } else {
-  if (len >= VA_EXCLUDE_START)
-   goto out_putf;
-
-  if ((flags & MAP_FIXED) && invalid_64bit_range(addr, len))
-   goto out_putf;
- }

  down_write(&current->mm->mmap_sem);
  retval = do_mmap(file, addr, len, prot, flags, off);
  up_write(&current->mm->mmap_sem);

-out_putf:
  if (file)
   fput(file);
 out:
diff --git a/include/asm-generic/mman.h b/include/asm-generic/mman.h
diff --git a/include/asm-ia64/mman.h b/include/asm-ia64/mman.h
index 6ba179f..936fa9c 100644
--- a/include/asm-ia64/mman.h
+++ b/include/asm-ia64/mman.h
@@ -22,4 +22,12 @@ #define MAP_NONBLOCK 0x10000  /* do not
 #define MCL_CURRENT 1  /* lock all current mappings */
 #define MCL_FUTURE 2  /* lock all future mappings */

+#ifdef __KERNEL__
+#ifndef __ASSEMBLY__
+#define arch_mmap_check ia64_mmap_check
+int ia64_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags);
+#endif
+#endif
+
 #endif /* _ASM_IA64_MMAN_H */
diff --git a/include/asm-sparc/mman.h b/include/asm-sparc/mman.h
index 88d1886..b7dc40b 100644
--- a/include/asm-sparc/mman.h
+++ b/include/asm-sparc/mman.h
@@ -35,4 +35,12 @@ #define MC_UNLOCKAS     6  /* Unlock ent

 #define MADV_FREE 0x5  /* (Solaris) contents can be freed */
```

```
+#ifdef __KERNEL__
+#ifndef __ASSEMBLY__
+#define arch_mmap_check sparc_mmap_check
+int sparc_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags);
+#endif
+#endif
+
 #endif /* __SPARC_MMAN_H__ */
diff --git a/include/asm-sparc64/mman.h b/include/asm-sparc64/mman.h
index 6fd878e..8cc1860 100644
--- a/include/asm-sparc64/mman.h
+++ b/include/asm-sparc64/mman.h
@@ -35,4 +35,12 @@ #define MC_UNLOCKAS     6  /* Unlock ent

 #define MADV_FREE 0x5  /* (Solaris) contents can be freed */

+#ifdef __KERNEL__
+#ifndef __ASSEMBLY__
+#define arch_mmap_check sparc64_mmap_check
+int sparc64_mmap_check(unsigned long addr, unsigned long len,
+  unsigned long flags);
+#endif
+#endif
+
 #endif /* __SPARC64_MMAN_H__ */
diff --git a/mm/mmap.c b/mm/mmap.c
index c1868ec..e66a0b5 100644
--- a/mm/mmap.c
+++ b/mm/mmap.c
@@ -30,6 +30,10 @@ #include <asm/uaccess.h>
 #include <asm/cacheflush.h>
 #include <asm/tlb.h>

+#ifndef arch_mmap_check
+#define arch_mmap_check(addr, len, flags) (0)
+#endif
+
 static void unmap_region(struct mm_struct *mm,
   struct vm_area_struct *vma, struct vm_area_struct *prev,
   unsigned long start, unsigned long end);
@@ -913,6 +917,10 @@ unsigned long do_mmap_pgoff(struct file
  if (!len)
   return -EINVAL;

+ error = arch_mmap_check(addr, len, flags);
+ if (error)
```

```
+  return error;
+
  /* Careful about overflows.. */
  len = PAGE_ALIGN(len);
  if (!len || len > TASK_SIZE)
@@ -1859,6 +1867,7 @@ unsigned long do_brk(unsigned long addr,
  unsigned long flags;
  struct rb_node ** rb_link, * rb_parent;
  pgoff_t pgoff = addr >> PAGE_SHIFT;
+ int error;

  len = PAGE_ALIGN(len);
  if (!len)
@@ -1867,6 +1876,12 @@ unsigned long do_brk(unsigned long addr,
  if ((addr + len) > TASK_SIZE || (addr + len) < addr)
   return -EINVAL;

+ flags = VM_DATA_DEFAULT_FLAGS | VM_ACCOUNT | mm->def_flags;
+
+ error = arch_mmap_check(addr, len, flags);
+ if (error)
+  return error;
+
  /*
   * mlock MCL_FUTURE?
   */
@@ -1907,8 +1922,6 @@ unsigned long do_brk(unsigned long addr,
  if (security_vm_enough_memory(len >> PAGE_SHIFT))
   return -ENOMEM;

- flags = VM_DATA_DEFAULT_FLAGS | VM_ACCOUNT | mm->def_flags;
-
  /* Can we just expand an old private anonymous mapping? */
  if (vma_merge(mm, prev, addr, addr + len, flags,
      NULL, NULL, pgoff, NULL))
```

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Linus Torvalds on Thu, 07 Sep 2006 15:17:04 GMT
View Forum Message <> Reply to Message

On Thu, 7 Sep 2006, Kirill Korotaev wrote:
>
> Does the patch below looks better?

Yes.

Apart from the whitespace corruption, that is.

I don't know how to get mozilla to not screw up whitespace.

Linus

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Willy Tarreau on Thu, 07 Sep 2006 20:07:14 GMT
View Forum Message <> Reply to Message

On Thu, Sep 07, 2006 at 08:17:04AM -0700, Linus Torvalds wrote:
>
>
> On Thu, 7 Sep 2006, Kirill Korotaev wrote:
> >
> > Does the patch below looks better?
>
> Yes.
>
> Apart from the whitespace corruption, that is.
>
> I don't know how to get mozilla to not screw up whitespace.

maybe by using it to download mutt or something saner ? :-)

More seriously, while we don't like email attachments because they make
it impossible to comment on a patch, maybe we should encourage people
with broken mailers to post small patches in both forms :
  - pure text for human review (spaces are not much of a problem here)
  - MIME to apply the patch.

At least when the do so, they should insist on it at the top of the
patch, or even manually mangle the patch header so that GIT (or any
other tool) does not use it.

> Linus

Regards
Willy

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Andrew Morton on Thu, 07 Sep 2006 23:42:07 GMT
View Forum Message <> Reply to Message

On Thu, 7 Sep 2006 22:07:14 +0200
Willy Tarreau <w@1wt.eu> wrote:

> On Thu, Sep 07, 2006 at 08:17:04AM -0700, Linus Torvalds wrote:
> >
> >
> > On Thu, 7 Sep 2006, Kirill Korotaev wrote:
> > >
> > > Does the patch below looks better?
> >
> > Yes.
> >
> > Apart from the whitespace corruption, that is.
> >
> > I don't know how to get mozilla to not screw up whitespace.

Me either. I've had a bug report in the mozilla system for maybe four
years concerning space-stuffing. Occasionally it comes to life but afaict
nothing ever changes.

I expect it'd be pretty easy to undo the space-stuffing in git.
In extremis I just do s/^ /^ / and it works. An automated solution would
need to recognise the appropriate headers (Format=Flowed, iirc).

> maybe by using it to download mutt or something saner ? :-)
>
> More seriously, while we don't like email attachments because they make
> it impossible to comment on a patch, maybe we should encourage people
> with broken mailers to post small patches in both forms :
> - pure text for human review (spaces are not much of a problem here)
> - MIME to apply the patch.

argh. That means that email contains two copies of the patch. So it
applies with `patch --dry-run' then causes havoc with `patch'

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Willy Tarreau on Fri, 08 Sep 2006 04:34:12 GMT
View Forum Message <> Reply to Message

On Thu, Sep 07, 2006 at 04:42:07PM -0700, Andrew Morton wrote:
> On Thu, 7 Sep 2006 22:07:14 +0200
> Willy Tarreau <w@1wt.eu> wrote:
>
> > On Thu, Sep 07, 2006 at 08:17:04AM -0700, Linus Torvalds wrote:
> > >
> > >
> > > On Thu, 7 Sep 2006, Kirill Korotaev wrote:
> > > >
> > > > Does the patch below looks better?

> > >
> > > Yes.
> > >
> > > Apart from the whitespace corruption, that is.
> > >
> > > I don't know how to get mozilla to not screw up whitespace.
>
> Me either. I've had a bug report in the mozilla system for maybe four
> years concerning space-stuffing. Occasionally it comes to life but afaict
> nothing ever changes.
>
> I expect it'd be pretty easy to undo the space-stuffing in git.

Perhaps, but it should not be up to the versionning system to decide to
change the contents of the patches which get merged. Otherwise, we will
not be able to trust it as much as today.

> In extremis I just do s/^ /^ / and it works. An automated solution would
> need to recognise the appropriate headers (Format=Flowed, iirc).

perhaps for this case, but then what will prevent us from trying to
implement dirtier features such as line un-wrapping ?

> > maybe by using it to download mutt or something saner ? :-)
> >
> > More seriously, while we don't like email attachments because they make
> > it impossible to comment on a patch, maybe we should encourage people
> > with broken mailers to post small patches in both forms :
> >   - pure text for human review (spaces are not much of a problem here)
> >   - MIME to apply the patch.
>
> argh. That means that email contains two copies of the patch. So it
> applies with `patch --dry-run' then causes havoc with `patch'

except if the text version is mangled in order not to be detected as
a patch. I suspect that inserting a space in front of "---" is enough
for patch not to find it. Don't get me wrong, I know this is dirty.
But as long as some people will use broken mailers, we'll get broken
patches. Some people occasionnaly switch to attachments stating they
have broken mailers, and others even post links to their patches,
which is annoying for potential reviewers. If we could give them
strict rules on how to proceed when they have such problems, it would
make the job easier for others.

willy

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Jes Sorensen on Fri, 08 Sep 2006 09:12:21 GMT
View Forum Message <> Reply to Message

>>>>> "Arjan" == Arjan van de Ven <arjan@infradead.org> writes:

Arjan> On Wed, 2006-09-06 at 12:27 -0600, Matthew Wilcox wrote:
>> On Wed, Sep 06, 2006 at 11:24:05AM -0700, Linus Torvalds wrote: >
>> If MIPS and parisc don't matter for the stable tree (very possible
>> - there > are no big commercial distributions for them), then
>> dammit, neither should > ia64 and sparc (there are no big
>> commercial distros for them either).
>>
>> Erm, RHEL and SLES both support ia64.

Arjan> but neither use -stable.

And getting a patch into -stable tends to be a really good argument
for a backport into the next vendor kernel :)

Cheers,
Jes

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by dev on Fri, 08 Sep 2006 15:12:31 GMT
View Forum Message <> Reply to Message

Linus Torvalds wrote:

>>Does the patch below looks better?
>
>
> Yes.
>
> Apart from the whitespace corruption, that is.
>
> I don't know how to get mozilla to not screw up whitespace.

What is funny is that mozilla doesn't screw up whitespaces.
2 people checked that patch from the email applies to the kernel.

I even checked the email myself and the only difference between "good"
patches and mine is that mine has "format=flowed" in
Content-Type: text/plain; charset=us-ascii; format=flowed

It looks like some mailers replace TABs with spaces when format=flowed
is specified. So are you sure that the problem is in mozilla?

Thanks,
Kirill

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Linus Torvalds on Fri, 08 Sep 2006 15:35:03 GMT
View Forum Message <> Reply to Message

On Fri, 8 Sep 2006, Kirill Korotaev wrote:
>
> I even checked the email myself and the only difference between "good"
> patches and mine is that mine has "format=flowed" in
> Content-Type: text/plain; charset=us-ascii; format=flowed
>
> It looks like some mailers replace TABs with spaces when format=flowed
> is specified. So are you sure that the problem is in mozilla?

Hey, what do you know? Good call. I can actually just "S"ave the message
to a file, and it is a perfectly fine patch. But when I view it in my mail
reader, your "format=flowed" means that it _shows_ it as being corrupted
(ie word wrapping and missing spaces at the beginning of lines).

Will apply, thanks. It would be better if your mailer didn't lie about the
format though (treating the text as "flowed" definitely isn't right, and
some mail gateways might actually find it meaningful, for all I know).

Linus

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by Dave Jones on Fri, 08 Sep 2006 15:49:45 GMT
View Forum Message <> Reply to Message

On Fri, Sep 08, 2006 at 08:35:03AM -0700, Linus Torvalds wrote:
 >
 >
 > On Fri, 8 Sep 2006, Kirill Korotaev wrote:
 > >
 > > I even checked the email myself and the only difference between "good"
 > > patches and mine is that mine has "format=flowed" in
 > > Content-Type: text/plain; charset=us-ascii; format=flowed
 > >
 > > It looks like some mailers replace TABs with spaces when format=flowed
 > > is specified. So are you sure that the problem is in mozilla?
 >
 > Hey, what do you know? Good call. I can actually just "S"ave the message

> to a file, and it is a perfectly fine patch. But when I view it in my mail
> reader, your "format=flowed" means that it _shows_ it as being corrupted
> (ie word wrapping and missing spaces at the beginning of lines).
>
> Will apply, thanks. It would be better if your mailer didn't lie about the
> format though (treating the text as "flowed" definitely isn't right, and
> some mail gateways might actually find it meaningful, for all I know).

I got bitten by this myself a while ago. Since then I added this
hack to my .procmailrc

```
:0fw
| /usr/bin/perl -pe 's/^(Content-Type: .*)format=flowed/\1format=flawed/'
```

now I see patches as they were intended..

Dave

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by dev on Fri, 08 Sep 2006 16:06:06 GMT
View Forum Message <> Reply to Message

Linus Torvalds wrote:
>
> On Fri, 8 Sep 2006, Kirill Korotaev wrote:
>
>>I even checked the email myself and the only difference between "good"
>>patches and mine is that mine has "format=flowed" in
>>Content-Type: text/plain; charset=us-ascii; format=flowed
>>
>>It looks like some mailers replace TABs with spaces when format=flowed
>>is specified. So are you sure that the problem is in mozilla?
>
>
> Hey, what do you know? Good call. I can actually just "S"ave the message
> to a file, and it is a perfectly fine patch. But when I view it in my mail
> reader, your "format=flowed" means that it _shows_ it as being corrupted
> (ie word wrapping and missing spaces at the beginning of lines).
Oh, I finally found how to tune Mozilla and fix it:

One need to edit defaults/pref/mailnews.js file to have:
pref("mailnews.send_plaintext_flowed", false); // RFC 2646=======
pref("mailnews.display.disable_format_flowed_support", true);

This makes Mozilla to send emails w/o "format=flowed".

Thanks a lot for your patience :)

Kirill

Subject: Re: [PATCH] IA64,sparc: local DoS with corrupted ELFs
Posted by rdunlap on Wed, 13 Sep 2006 15:45:20 GMT
View Forum Message <> Reply to Message

On Fri, 08 Sep 2006 20:09:22 +0400 Kirill Korotaev wrote:

> Linus Torvalds wrote:
> >
> > On Fri, 8 Sep 2006, Kirill Korotaev wrote:
> >
> >>I even checked the email myself and the only difference between "good"
> >>patches and mine is that mine has "format=flowed" in
> >>Content-Type: text/plain; charset=us-ascii; format=flowed
> >>
> >>It looks like some mailers replace TABs with spaces when format=flowed
> >>is specified. So are you sure that the problem is in mozilla?
> >
> >
> > Hey, what do you know? Good call. I can actually just "S"ave the message
> > to a file, and it is a perfectly fine patch. But when I view it in my mail
> > reader, your "format=flowed" means that it _shows_ it as being corrupted
> > (ie word wrapping and missing spaces at the beginning of lines).
> Oh, I finally found how to tune Mozilla and fix it:
>
> One need to edit defaults/pref/mailnews.js file to have:
> pref("mailnews.send_plaintext_flowed", false); // RFC 2646=======
> pref("mailnews.display.disable_format_flowed_support", true);
>
> This makes Mozilla to send emails w/o "format=flowed".
>
> Thanks a lot for your patience :)

Here is some (similar) info for thunderbird:
  http://mbligh.org/linuxdocs/Email/Clients/Thunderbird

~Randy