

# [PATCH 6/7] uml: cleanup unprofile expression and build infrastructure

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-04/msg06710.html>

---

- *From:* Paolo 'Blaisorblade' Giarrusso <[blaisorblade@xxxxxxxx](mailto:blaisorblade@xxxxxxxx)>
  - *Date:* Sun, 30 Apr 2006 16:16:22 +0200
- 

From: Paolo 'Blaisorblade' Giarrusso <[blaisorblade@xxxxxxxx](mailto:blaisorblade@xxxxxxxx)>

\*) Rather than duplicate in various buggy ways the application of CFLAGS\_NO\_HARDENING and UNPROFILE (which apply to the same files), centralize it in Makefile.rules. UNPROFILE\_OBJS mustn't be listed in USER\_OBJS but are compiled as such.

I've also verified that unprofile didn't work in the current form, because we set `_c_flags` directly (using CFLAGS and not USER\_CFLAGS, which is wrong), which is normally used by `c_flags`, but we also override `c_flags` for all USER\_OBJS, and there we don't call unprofile.

Instead it only worked for `unmap.o`, the only one which wasn't a USER\_OBJ.

We need to set `c_flags` (which is not a public Kbuild API) to clear a lot of compilation flags like `-nostdinc` which Kbuild forces on everything.

\*) Rather than `$(CFLAGS_$(notdir $@))`, which expands to `CFLAGS_anObj.s` when building "`anObj.s`", use `$(CFLAGS_$(*)F.o)` which always accesses `CFLAGS_anObj.o`, like done by Kbuild.

\*) Make `c_flags` apply to all targets having the same basename, rather than listing `.s`, `.i`, `.lst` and `.o`, with the use (which I tested) of `$(USER_OBJS:.o=.%): c_flags = ...`

and of

```
- $(obj)/unmap.c: _c_flags = ...
+ $(obj)/unmap.%: _c_flags = ...
```

Signed-off-by: Paolo 'Blaisorblade' Giarrusso <[blaisorblade@xxxxxxxx](mailto:blaisorblade@xxxxxxxx)>

---

```
arch/um/kernel/skas/Makefile | 9 ++-----
arch/um/scripts/Makefile.rules | 12 ++++++++--
arch/um/sys-i386/Makefile | 10 +++++-----
arch/um/sys-x86_64/Makefile | 10 +++++-----
4 files changed, 22 insertions(+), 19 deletions(-)
```

[PATCH 6/7] uml: cleanup unprofile expression and build infrastructure

```

diff --git a/arch/um/kernel/skas/Makefile b/arch/um/kernel/skas/Makefile
index ad84296..ea3a8e4 100644
--- a/arch/um/kernel/skas/Makefile
+++ b/arch/um/kernel/skas/Makefile
@@ -6,16 +6,11 @@
obj-y := clone.o exec_kern.o mem.o mmu.o process_kern.o \
syscall.o tlb.o uaccess.o

-USER_OBJS := clone.o
-
-include arch/um/scripts/Makefile.rules
-
# clone.o is in the stub, so it can't be built with profiling
# GCC hardened also auto-enables -fpic, but we need %ebx so it can't work ->
# disable it

CFLAGS_clone.o := $(CFLAGS_NO_HARDENING)
+UNPROFILE_OBJS := clone.o

-# since we're setting c_flags we _must_ add $(CFLAGS_$(*)F.o).
-
-$(obj)/clone.o : c_flags = -Wp,-MD,$(depfile) $(call unprofile,$(USER_CFLAGS)) $(CFLAGS_$(*)F.o)
+include arch/um/scripts/Makefile.rules
diff --git a/arch/um/scripts/Makefile.rules b/arch/um/scripts/Makefile.rules
index 5e7a9c3..1347dc6 100644
--- a/arch/um/scripts/Makefile.rules
+++ b/arch/um/scripts/Makefile.rules
@@ -7,11 +7,19 @@
USER_SINGLE_OBJS := \
USER_OBJS += $(filter %_user.o,$(obj-y)) $(obj-m) $(USER_SINGLE_OBJS)
USER_OBJS := $(foreach file,$(USER_OBJS),$(obj)/$(file))

-$(USER_OBJS) $(USER_OBJS:.o=.i) $(USER_OBJS:.o=.s) $(USER_OBJS:.o=.lst): \
- c_flags = -Wp,-MD,$(depfile) $(USER_CFLAGS) $(CFLAGS_$(notdir $@))
+$(USER_OBJS:.o=.%): \
+ c_flags = -Wp,-MD,$(depfile) $(USER_CFLAGS) $(CFLAGS_$(*)F.o)
$(USER_OBJS) : CHECKFLAGS := -D__linux__ -Dlinux -D__STDC__ \
-Dunix -D__unix__ -D__$$(SUBARCH)__

+# These are like USER_OBJS but filter USER_CFLAGS through unprofile instead of
+# using it directly.
+UNPROFILE_OBJS := $(foreach file,$(UNPROFILE_OBJS),$(obj)/$(file))
+
+$(UNPROFILE_OBJS:.o=.%): \
+ c_flags = -Wp,-MD,$(depfile) $(call unprofile,$(USER_CFLAGS)) $(CFLAGS_$(*)F.o)
+$(UNPROFILE_OBJS) : CHECKFLAGS := -D__linux__ -Dlinux -D__STDC__ \
+ -Dunix -D__unix__ -D__$$(SUBARCH)__

# The stubs and unmap.o can't try to call mcount or update basic block data
define unprofile
diff --git a/arch/um/sys-i386/Makefile b/arch/um/sys-i386/Makefile
index 3734c3e..374d61a 100644

```

[PATCH 6/7] uml: cleanup unprofile expression and build infrastructure

```
--- a/arch/um/sys-i386/Makefile
+++ b/arch/um/sys-i386/Makefile
@@ -8,16 +8,16 @@ subarch-obj-y = lib/bitops.o kernel/sema
subarch-obj-$(CONFIG_HIGHMEM) += mm/highmem.o
subarch-obj-$(CONFIG_MODULES) += kernel/module.o

-USER_OBJS := bugs.o ptrace_user.o sigcontext.o fault.o stub_segv.o
+USER_OBJS := bugs.o ptrace_user.o sigcontext.o fault.o

USER_OBJS += user-offsets.s
extra-y += user-offsets.s

-CFLAGS_stub_segv.o := $(CFLAGS_NO_HARDENING)
-
extra-$(CONFIG_MODE_TT) += unmap.o

+UNPROFILE_OBJS := stub_segv.o
+CFLAGS_stub_segv.o := $(CFLAGS_NO_HARDENING)
+
include arch/um/scripts/Makefile.rules

-$(obj)/stub_segv.o $(obj)/unmap.o: \
- _c_flags = $(call unprofile,$(CFLAGS))
+$(obj)/unmap.o: _c_flags = $(call unprofile,$(CFLAGS))
diff --git a/arch/um/sys-x86_64/Makefile b/arch/um/sys-x86_64/Makefile
index 6d3b29c..c19794d 100644
--- a/arch/um/sys-x86_64/Makefile
+++ b/arch/um/sys-x86_64/Makefile
@@ -16,16 +16,16 @@ subarch-obj-$(CONFIG_MODULES) += kernel/

ldt-y = ../sys-i386/ldt.o

-USER_OBJS := ptrace_user.o sigcontext.o stub_segv.o
+USER_OBJS := ptrace_user.o sigcontext.o

USER_OBJS += user-offsets.s
extra-y += user-offsets.s

-CFLAGS_stub_segv.o := $(CFLAGS_NO_HARDENING)
-
extra-$(CONFIG_MODE_TT) += unmap.o

+UNPROFILE_OBJS := stub_segv.o
+CFLAGS_stub_segv.o := $(CFLAGS_NO_HARDENING)
+
include arch/um/scripts/Makefile.rules

-$(obj)/stub_segv.o $(obj)/unmap.o: \
- _c_flags = $(call unprofile,$(CFLAGS))
+$(obj)/unmap.o: _c_flags = $(call unprofile,$(CFLAGS))
-
```

[PATCH 6/7] uml: cleanup unprofile expression and build infrastructure

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in  
the body of a message to majordomo@xxxxxxxxxxxxxxxxx  
More majordomo info at <http://vger.kernel.org/majordomo-info.html>  
Please read the FAQ at <http://www.tux.org/lkml/>