

[PATCH 1/9] x86-64 dont use r10 in AES

Source: <http://linux.derkeiler.com/Mailing-Lists/Kernel/2005-11/9632.html>

From: Benjamin LaHaise (*bcr1_at_kvack.org*)

Date: 11/30/05

Date: Tue, 29 Nov 2005 23:21:26 -0500

To: Andi Kleen <ak@suse.de>

In preparation for using r10 as current on x86-64, the AES crypto asm must be changed to use a different register for its temporary. Use r12 instead of r10.

```

---
 arch/x86_64/crypto/aes-x86_64-asm.S | 27 ++++++-----
 1 files changed, 14 insertions(+), 13 deletions(-)
 applies-to: 9604b37c2530212b42e06391eefefc09d758a418
 d042c714e30b53a4ed443f4a224e8030ebd63a73
 diff --git a/arch/x86_64/crypto/aes-x86_64-asm.S b/arch/x86_64/crypto/aes-x86_64-asm.S
 index 483cbb2..5304e74 100644
 --- a/arch/x86_64/crypto/aes-x86_64-asm.S
 +++ b/arch/x86_64/crypto/aes-x86_64-asm.S
 @@ -43,39 +43,40 @@
  #define R7E    %ebp
  #define R8     %r8
  #define R9     %r9
 -#define R10    %r10
  #define R11    %r11
 +#define R12    %r12

 -#define prologue(FUNC,BASE,B128,B192,r1,r2,r3,r4,r5,r6,r7,r8,r9,r10,r11) \
 +#define prologue(FUNC,BASE,B128,B192,r1,r2,r3,r4,r5,r6,r7,r8,r9,r11,r12) \
  .global FUNC; \
  .type    FUNC,@function; \
  .align  8; \
 -FUNC:   movq   r1,r2; \
 +FUNC:   pushq  r12; \
 +       movq   r1,r2; \
  movq   r3,r4; \
  leaq   BASE+52(r8),r9; \
 -       movq   r10,r11; \
  movl   (r7),r5 ## E; \
  movl   4(r7),r1 ## E; \
  movl   8(r7),r6 ## E; \
  movl   12(r7),r7 ## E; \
 -       movl   (r8),r10 ## E; \
 +       movl   (r8),r12 ## E; \
  xorl   -48(r9),r5 ## E; \
  xorl   -44(r9),r1 ## E; \
  xorl   -40(r9),r6 ## E; \
  xorl   -36(r9),r7 ## E; \
 -       cmpl   $24,r10 ## E; \
 +       cmpl   $24,r12 ## E; \
  jb     B128; \

```

Linux-Kernel: [PATCH 1/9] x86-64 dont use r10 in AES

```

    leaq    32(r9),r9;           \
    je     B192;                \
    leaq    32(r9),r9;

-#define epilogue(r1,r2,r3,r4,r5,r6,r7,r8,r9) \
+#define epilogue(r1,r2,r3,r4,r5,r6,r7,r8,r9,r12) \
    movq    r1,r2;             \
    movq    r3,r4;             \
    movl    r5 ## E,(r9);      \
    movl    r6 ## E,4(r9);     \
    movl    r7 ## E,8(r9);     \
    movl    r8 ## E,12(r9);    \
+    popq   r12;               \
    ret;

#define round(TAB,OFFSET,r1,r2,r3,r4,r5,r6,r7,r8,ra,rb,rc,rd) \
@@ -129,23 +130,23 @@ FUNC:      movq    r1,r2;           \
    movl    r4 ## E,r2 ## E;

#define entry(FUNC,BASE,B128,B192) \
-    prologue(FUNC,BASE,B128,B192,R2,R8,R7,R9,R1,R3,R4,R6,R10,R5,R11)
+    prologue(FUNC,BASE,B128,B192,R2,R8,R7,R9,R1,R3,R4,R6,R5,R11,R12)

-#define return_epilogue(R8,R2,R9,R7,R5,R6,R3,R4,R11)
+#define return_epilogue(R8,R2,R9,R7,R5,R6,R3,R4,R11,R12)

#define encrypt_round(TAB,OFFSET) \
-    round(TAB,OFFSET,R1,R2,R3,R4,R5,R6,R7,R10,R5,R6,R3,R4) \
+    round(TAB,OFFSET,R1,R2,R3,R4,R5,R6,R7,R12,R5,R6,R3,R4) \
    move_regs(R1,R2,R5,R6)

#define encrypt_final(TAB,OFFSET) \
-    round(TAB,OFFSET,R1,R2,R3,R4,R5,R6,R7,R10,R5,R6,R3,R4)
+    round(TAB,OFFSET,R1,R2,R3,R4,R5,R6,R7,R12,R5,R6,R3,R4)

#define decrypt_round(TAB,OFFSET) \
-    round(TAB,OFFSET,R2,R1,R4,R3,R6,R5,R7,R10,R5,R6,R3,R4) \
+    round(TAB,OFFSET,R2,R1,R4,R3,R6,R5,R7,R12,R5,R6,R3,R4) \
    move_regs(R1,R2,R5,R6)

#define decrypt_final(TAB,OFFSET) \
-    round(TAB,OFFSET,R2,R1,R4,R3,R6,R5,R7,R10,R5,R6,R3,R4)
+    round(TAB,OFFSET,R2,R1,R4,R3,R6,R5,R7,R12,R5,R6,R3,R4)

/* void aes_encrypt(void *ctx,u8 *out, const u8 *in) */

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

```