

## [PATCH 06/40] KVM: x86 emulator: Group decoding for groups 4 and 5

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2008-03/msg12004.html>

---

- *From:* Avi Kivity <[avi@xxxxxxxxxxxxx](mailto:avi@xxxxxxxxxxxxx)>
  - *Date:* Mon, 31 Mar 2008 17:36:50 +0300
- 

Add group decoding support for opcode 0xfe (group 4) and 0xff (group 5).

Signed-off-by: Avi Kivity <[avi@xxxxxxxxxxxxx](mailto:avi@xxxxxxxxxxxxx)>

---  
arch/x86/kvm/x86\_emulate.c | 40 ++++++-----  
1 files changed, 10 insertions(+), 30 deletions(-)

diff --git a/arch/x86/kvm/x86\_emulate.c b/arch/x86/kvm/x86\_emulate.c  
index 52e65ae..7310368 100644

--- a/arch/x86/kvm/x86\_emulate.c

+++ b/arch/x86/kvm/x86\_emulate.c

@@ -70,7 +70,7 @@

#define GroupMask 0xff /\* Group number stored in bits 0:7 \*/

enum {

- Group1A, Group3\_Byte, Group3,

+ Group1A, Group3\_Byte, Group3, Group4, Group5,

};

static u16 opcode\_table[256] = {

@@ -174,7 +174,7 @@ static u16 opcode\_table[256] = {

ImplicitOps, ImplicitOps, Group | Group3\_Byte, Group | Group3,

/\* 0xF8 - 0xFF \*/

ImplicitOps, 0, ImplicitOps, ImplicitOps,

- 0, 0, ByteOp | DstMem | SrcNone | ModRM, DstMem | SrcNone | ModRM

+ 0, 0, Group | Group4, Group | Group5,

};

static u16 twobyte\_table[256] = {

@@ -246,6 +246,12 @@ static u16 group\_table[] = {

DstMem | SrcImm | ModRM | SrcImm, 0,

DstMem | SrcNone | ModRM, ByteOp | DstMem | SrcNone | ModRM,

0, 0, 0, 0,

+ [Group4\*8] =

+ ByteOp | DstMem | SrcNone | ModRM, ByteOp | DstMem | SrcNone | ModRM,

+ 0, 0, 0, 0, 0, 0,

+ [Group5\*8] =

+ DstMem | SrcNone | ModRM, DstMem | SrcNone | ModRM, 0, 0,

[PATCH 06/40] KVM: x86 emulator: Group decoding for groups 4 and 5

```
+ SrcMem | ModRM, 0, SrcMem | ModRM | Stack, 0,  
};
```

```
static u16 group2_table[] = {  
@@ -1097,7 +1103,6 @@ static inline int emulate_grp45(struct x86_emulate_ctxt *ctxt,  
struct x86_emulate_ops *ops)  
{  
struct decode_cache *c = &ctxt->decode;  
- int rc;  
  
switch (c->modrm_reg) {  
case 0: /* inc */  
@@ -1107,36 +1112,11 @@ static inline int emulate_grp45(struct x86_emulate_ctxt *ctxt,  
emulate_1op("dec", c->dst, ctxt->eflags);  
break;  
case 4: /* jmp abs */  
- if (c->b == 0xff)  
- c->eip = c->dst.val;  
- else {  
- DPRINTF("Cannot emulate %02x\n", c->b);  
- return X86EMUL_UNHANDLEABLE;  
- }  
+ c->eip = c->src.val;  
break;  
case 6: /* push */  
-  
- /* 64-bit mode: PUSH always pushes a 64-bit operand. */  
-  
- if (ctxt->mode == X86EMUL_MODE_PROT64) {  
- c->dst.bytes = 8;  
- rc = ops->read_std((unsigned long)c->dst.ptr,  
- &c->dst.val, 8, ctxt->vcpu);  
- if (rc != 0)  
- return rc;  
- }  
- register_address_increment(c->regs[VCPU_REGS_RSP],  
- -c->dst.bytes);  
- rc = ops->write_emulated(register_address(ctxt->ss_base,  
- c->regs[VCPU_REGS_RSP]), &c->dst.val,  
- c->dst.bytes, ctxt->vcpu);  
- if (rc != 0)  
- return rc;  
- c->dst.type = OP_NONE;  
+ emulate_push(ctxt);  
break;  
- default:  
- DPRINTF("Cannot emulate %02x\n", c->b);  
- return X86EMUL_UNHANDLEABLE;  
}  
return 0;  
}
```

—  
1.5.4.5

—  
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/>