header.S
@512
_start

→ main.c

↓
start_of_setup

- bootloader heap
- stack
- cmp 5a5a aa55 sig
- call main

main.c

- copy boot_params
- console init
- init_heap()
- validate_cpu
- detect_memory e820
- keyboard
- :
- go_to_protected_mode();

pm.c

- a20
- mask ints PIC
- setup_idt()
- setup_gdt() (boot_gdt[])

- prot_mode_jump(code32_start, boot_params)
  0x100.000 gdb: arch i386

↳ pmjump.S

- code16 → code32 : flag for protectedmode in CR0
- jump with boot_gdt_CS to 32 bit code
- → jump to code32_start 0x100.000

↳ head_64.S
in boot/compressed

86/boot/compressed/head_64.S :

- ebp ← loaded at address
- ebx ← LOAD_PHYSICAL 0x1000 000 = 16 MiB   +BP_init_size

decompress uses boot_heap 0x10.000 = 64 KiB
boot_stack 16 KiB

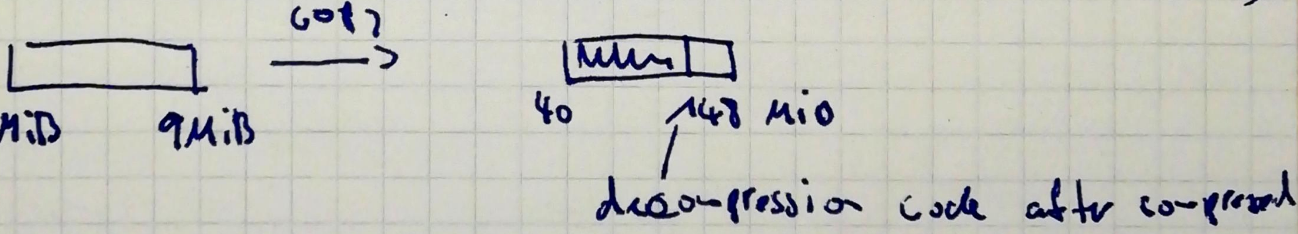
- GDT with 64bit segments
- PAE in CR4
- boot_pagetables in CR3 ; L
- LONG-Mode in EFER @ MSR LME

"Enter paged & protected mode ACTIVATING LONG MODE"
PG in CR0

- code64 @ offset 512

%rbp decompressed kernel start 0x1000 000

%rbx +BP_init_size − compressed size (_end vmlinux.lds.S)



call extract_kernel with parameters for heap_,

input_len, output_l_
8MiB 32MiB

→ returns kernel_start in %RAX

jmp *%rax

misc.c KASLR enabled

→ choose_random output before decompress

→ initialize_identity_maps();

mem_avoid_init