邊界和型態檢查

邊界檢查

邊界檢查是用來確保所有對 segment 的存取動作，都在 segment 的有效範圍內。一個 segment 的有效範圍，由 segment descriptor 中的參數來決定（參考「記憶體管理」的「分段架構」）。決定有效範圍的方式如下：

- 若 B = 0，則範圍的最大值 MAX 為 FFFFH；若 B = 1，則 MAX 為 FFFFFFFFH。
- 若 G = 0，則有效邊界 LIMIT 即為 descriptor 中的邊界值，即有效邊界可以從 0 到 FFFFFH。若 G = 1，則有效邊界的最左端 12 bit 不被檢查，即有效邊界可以從 FFFH 到 FFFFFFFFH；也就是說，若邊界值是 0，有效邊界 LIMIT 則是 FFFH，若邊界值是 1，則 LIMIT 是 1FFFH。
- 若 segment 是資料 segment，而且其 E = 1，則有效範圍是 LIMIT + 1 到 MAX。若 E = 0 或 segment 是程式碼的 segment，則有效範圍是 0 到 LIMIT。

任何試圖存取在有效範圍之外的動作，都會導致例外（exception）。所以，試圖在 LIMIT - 1 的地方讀取一個 word（16 bit），也會導致例外。
除了對一般的 segment 有邊界檢查之外，對 GDT 和 IDT 也會進行邊界檢查。在 GDTR 和 IDTR 中存放的邊界值可以避免存取到 GDT 和 IDT 外面的值。同樣的，在 LDTR 和工作暫存器（task register）中也有存放從 segment descriptor 中讀取的邊界值，所以對 LDT 和 TSS 也會進行邊界檢查。

型態檢查

型態檢查是用來避免對 segment（或 gate，gate 是一種系統 segment）進行不適當的動作，例如把資料 segment 當成程式執行。Segment 的型態由 S 旗標和型態位元決定（參考「記憶體管理」的「分段架構」）。
型態檢查有下面幾種（這裡只是舉一些例子）：

- 分段暫存器的檢查：
  - 只有程式碼的 segment selector 能被載入到 CS 中。
  - 不能讀取的程式 segment selector（R = 0）不能載入到 DS、ES、FS、GS 中。
  - 只有能寫入的資料 segment selector（W = 1）被載入到 SS 中。
  - 不能把 null segment（GDT 的第 0 個 segment descriptor）載入到 CS 或 SS 中。
- LDTR 或工作暫存器的檢查：
  - 只有 LDT 的 segment selector 能被載入到 LDTR 中。
  - 只有 TSS 的 segment selector 能被載入到工作暫存器中。
- 存取 segment 中的資料時的檢查：
  - 不能把資料寫入一個可執行的 segment（程式碼 segment）。
  - 不能把資料寫入 W = 0（不允許寫入）的資料 segment。
  - 不能讀取 R = 0（不允許讀取）的程式碼 segment 中的資料。
  - 不能對 null segment 進行存取。
- 指令中有分段暫存器時的檢查：
  - 跨段的（far）CALL 或 JMP 指令，只能到程式碼 segment、call gate、task gate、或是 TSS。
  - 載入一些系統暫存器的指令（如 LLDT、LTR、LAR、LSL）所參考的 segment descriptor 的型態必須相符。
  - IDT 的 entry 必須是 interrupt gate、trap gate、或是 task gate。