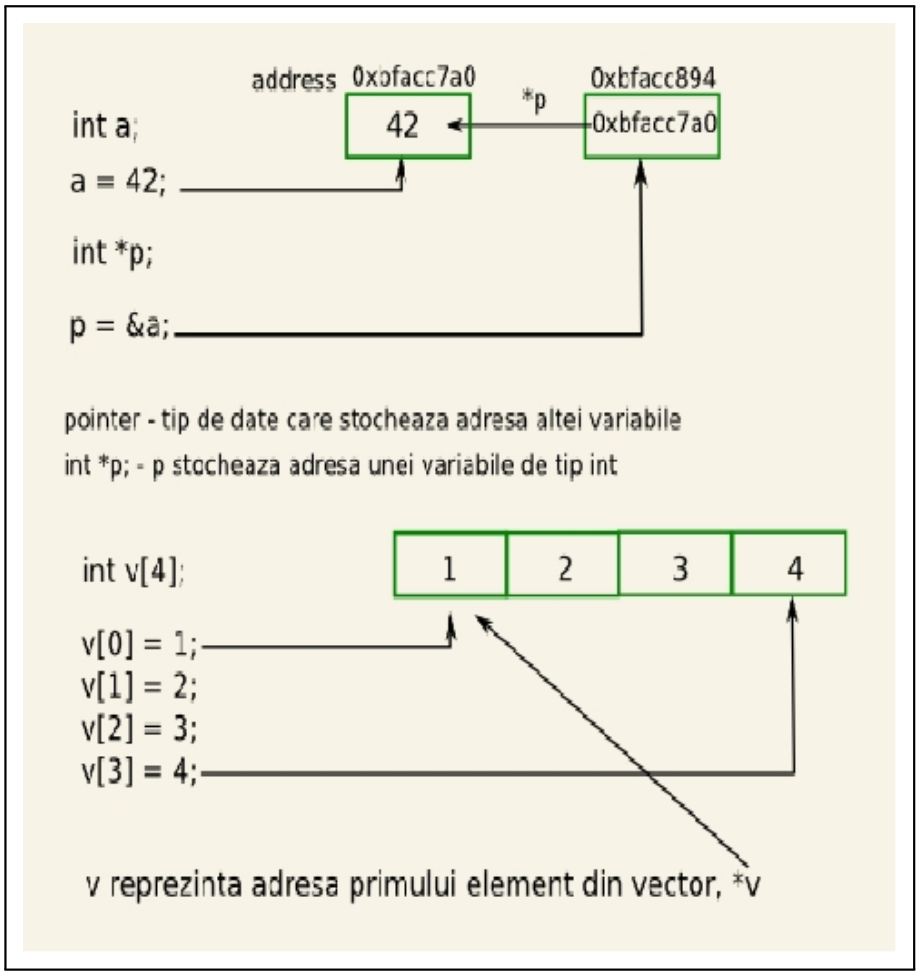
Laborator 4

Gestiunea Memoriei

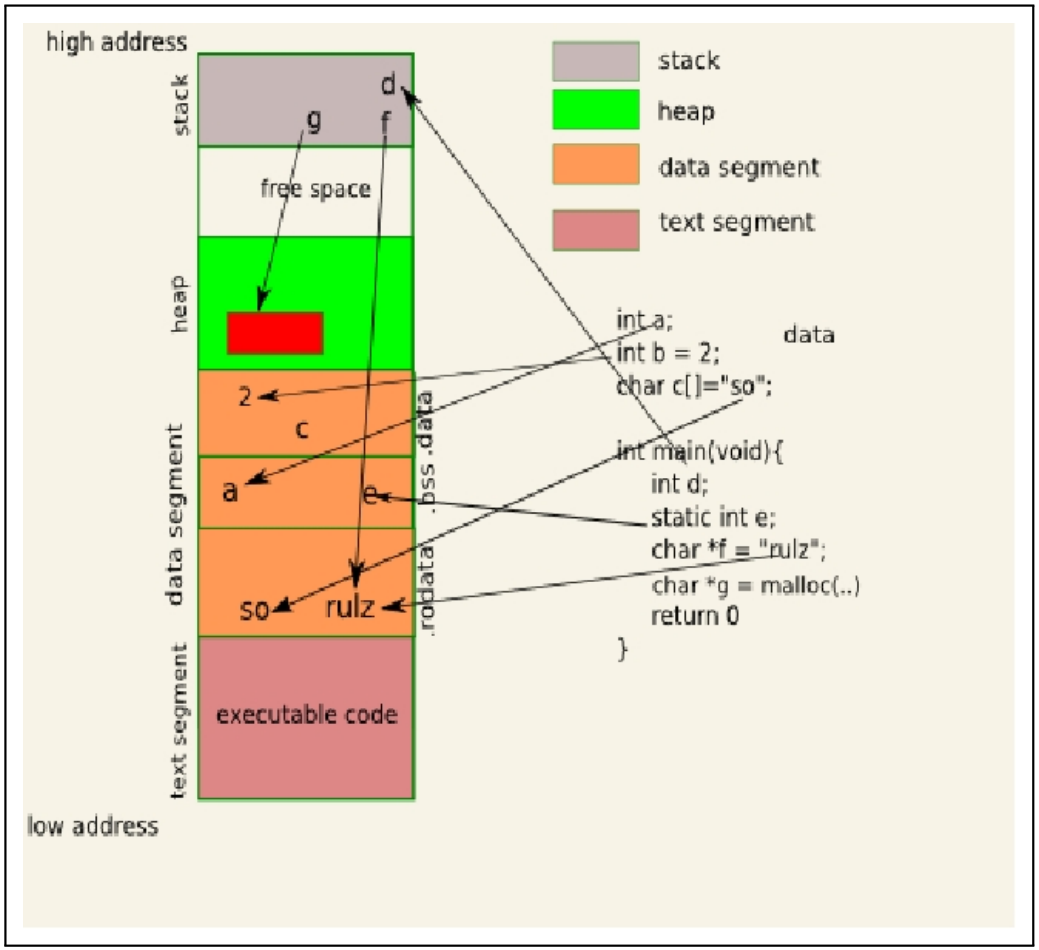
Sisteme de Operare

10 -16 Martie 2011

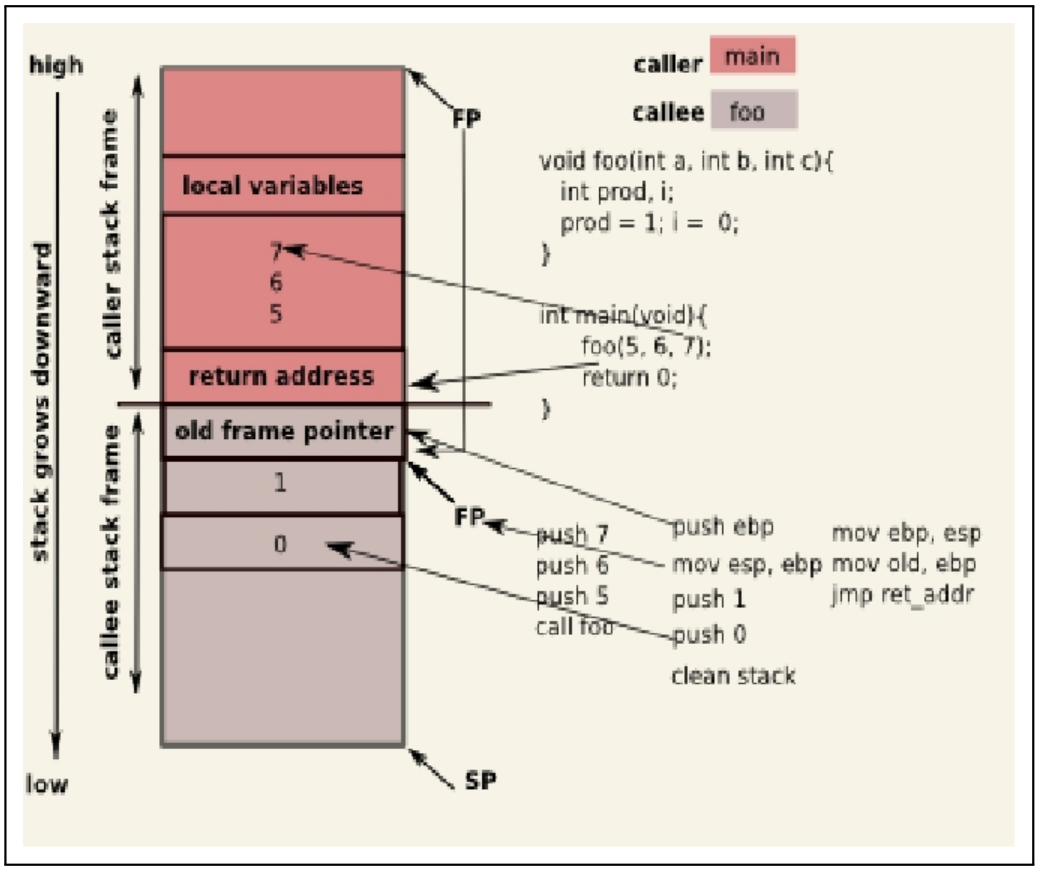


- Primitive (char , int)
- Pointer
- Array , Struct

Spațiul de adresă al unui proces



Pasarea parametrilor pe stivă



Alocarea memoriei

- Linux
  - void *malloc(size_t size);
  - void *calloc(size_t nmemb, size_t size);
  - void *realloc(void *ptr, size_t size);
  - void free(void *ptr);
- Windows
  - HANDLE HeapCreate(flOptions , dwInitialSize, dwMaximumSize);
  - BOOL HeapDestroy(hHeap);
  - LPVOID HeapAlloc(hHeap, dwFlags, dwBytes);
  - HeapReAlloc(hHeap, dwFlags, lpMem, dwBytes);
  - HeapFree(hHeap, dwFlags, lpMem);

- acces invalid

```
char s[4]; sprintf(s,"%s","so_rulz");
```

- memory leak
  - pierderea referintei la zona de memorie

```
for(i = 0; i < 10; i++)
        a = malloc(16*sizeof(int));
free(a);
```

- dangling reference
  - accesul la o zona de memorie care a fost anterior eliberata

```
a = malloc(16*sizeof(int));
b = a; free(b);
printf("%d", a[i]);
```

  - memoria alocata pentru a a fost eliberata prin intermediul lui b

GDB

- fișierele trebuie compilate cu opțiunea -g
- se transmite ca argument numele executabilului

```
gdb ./a.out
```

- comenzi GDB utile
  - `bt` - backtrace
  - `run` - rulare
  - `step, next` - următoarea instrucțiune
  - `quit` - părăsirea depanatorului
  - `set args` - stabilirea argumentelor de rulare
  - `disassamble` - afișează codul mașină generat de compilator
  - `info reg` - afișează conținutul registrilor
  - `man gdb` - pentru mai multe detalii

mcheck, mtrace

- mcheck
  - verifică consistența heap-ului.
  - MALLOC_CHECK_=1 ./executabil
- mtrace
  - detectează memory leak-urile
  - mtrace(), muntrace() , pe regiunea inspectată.

valgrind

- suită de utilitare pentru debugging și profiling
- memcheck, callgrind, helgrind
- memcheck
  - valgrind –tool=memcheck ./executabil
  - detectează
    - folosirea de memorie neinițializată
    - citire/scriere din/in memorie după ce regiunea respectivă a fost eliberată
    - memory leak-uri
    - citirea/scriere dincolo de sfârșitul zonei alocate
    - folosirea necorespunzatoare a apelurilor malloc/new și free/delete
    - citirea/scrierea pe stivă în zone necorespunzătoare

Keywords

- Spațiu de adresă
  - .text
  - .data .rodata .bss
  - stivă
  - heap
- Alocarea memoriei
  - malloc / calloc / realloc
  - HeapAlloc / HeapReAlloc
- Dezalocarea memoriei
  - free
  - HeapFree
- accesul invalid
  - gdb
  - mcheck
- memory leak
  - valgrind
  - mtrace

- Care este diferența între char a[]="hello" și char *a = "hello";
- Cum ați implementa o funcție care aloca N octeți pe stivă? Cum arată funcția free în acest caz?
- Cum putem determina direcția de creștere a stivei? (de la adrese mari la adrese mici sau invers)