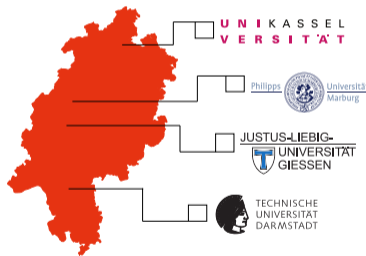


Excursion 1: Program Memory Layout

Hessisches Kompetenzzentrum für Hochleistungsrechnen (HKHLR)

Dr. Christian Iwainsky

V1.0



HKHLR is funded by the Hessian Ministry of Sciences and Arts



This segment's contents:

- ▶ Memory fundamentals, such as:
 - ▶ Head,
 - ▶ Stack (aka call-stack),
 - ▶ Variables and arguments, and
 - ▶ Pointers and allocated memory.
- ▶ Stack in context of multi-threading or OpenMP.



Two categories of variables:

▶ Direct access:

▶ `double i=pi;`

▶ `int j=1;`

▶ Indirect access:

▶ `double a[10];`

`a[j]=i;`

▶ `long int * list;`

`list=(long int*) malloc(sizeof(long int)*100);`

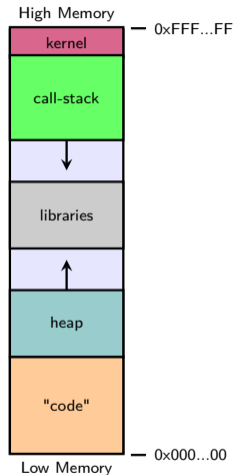
Out of bound access:

▶ `a[11]=0;`

▶ `list[-1]=0;`

The virtual memory of any process is segmented with a structured defined by conventions:

- ▶ Call-stack,
- ▶ Heap,
- ▶ Static variables, constants, and code (called text-segment),
- ▶ Space for dynamic libraries,
- ▶ Kernel space,
- ▶ Other "reserved" memory as required.

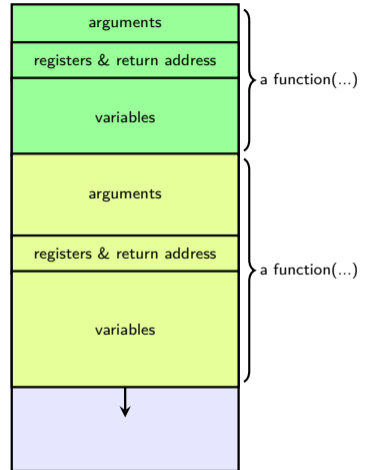


Stack-frame contents:

- ▶ Function arguments in reverse order,
- ▶ Return address,
- ▶ Saved registers, and
- ▶ Local variables.

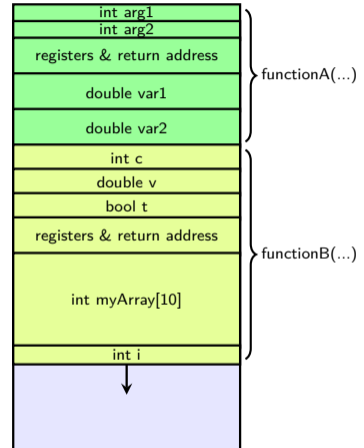
Local variables and fields can be:

- ▶ Rearranged,
- ▶ Padded,
- ▶ and aligned.



"Code example"

```
1 functionA(int arg1, int arg2){  
2     double var1, var2;  
3     functionB(var1,10.0,true);  
4 }  
5  
6 functionB(int c, double v, bool t){  
7     int myArray[10];  
8     for (int i=0;i<15;++)  
9         myArray[i]=0;  
10 }
```

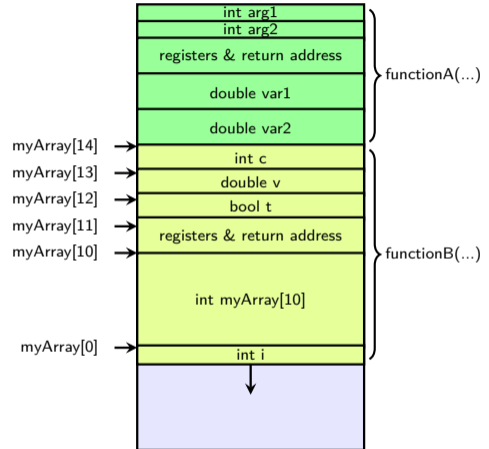


"Code example"

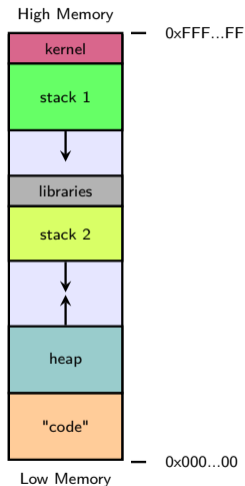
```

1 functionA(int arg1, int arg2){
2     double var1, var2;
3     functionB(var1,10.0,true);
4 }
5
6 functionB(int c, double v, bool t){
7     int myArray[10];
8     for (int i=0;i<15;++i)
9         myArray[i]=0;
10 }
    
```

The out of bounds access of `myArray[]` writes to return address and saved registers.



- ▶ Multi-threading and OpenMP require a call-stack for each thread,
- ▶ Additional call-stacks usually located next to mapped memory, such as libraries,
- ▶ PThreads have a default stack size of 2 MB on 64bit systems,
- ▶ PThread stack size can be modified by `RLIMIT_STACK` or `ulimit -s`,
- ▶ Default OpenMP stack size is 4 MB (2 MB) on 64bit (32bit) systems,
- ▶ OpenMP thread stack size is specified by `OMP_STACKSIZE`.
- ▶ In **Multi-threading** there are multiple stacks.



This segments contents:

- ▶ Memory fundamentals, such as:
 - ▶ Head,
 - ▶ Stack (aka call-stack),
 - ▶ variables and arguments, and
 - ▶ pointers and allocated memory.
- ▶ Stack in context of multi-threading or OpenMP.

