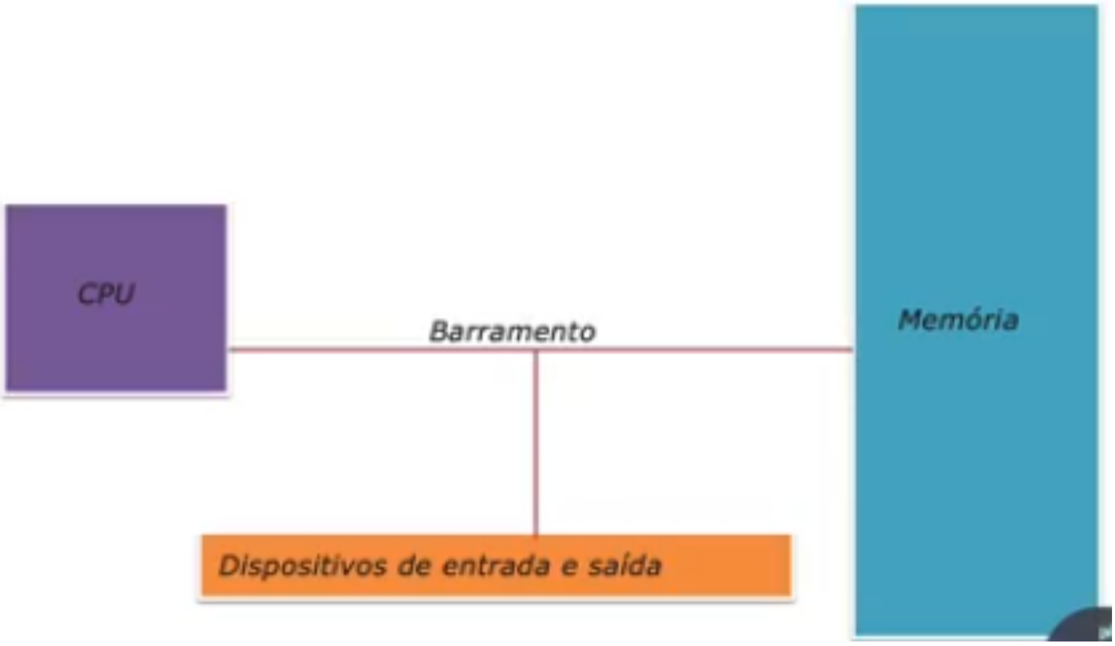
Arquitetura de Computadores

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Organização Básica de Computadores

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~



~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

CPU - Central Processing Unit

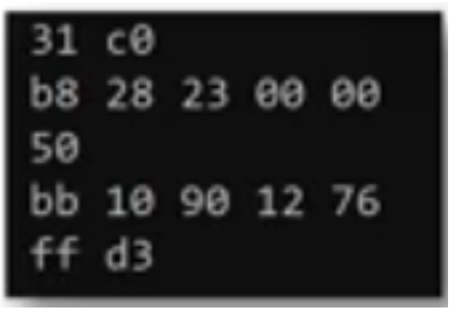
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

CPU - É o "cérebro" de um computador e é responsável por realizar operações lógicas aritméticas, processamento de dados etc.

O processador é responsável por executar o código de máquina de um programa de computador.

Machine Code é um conjunto de instuções que a CPU processa, essas instruções movem dados, realizam operações lógicas, aritméticas etc.

Essas instuções são representadas em formato hexadecimal.



código de máquina é difícil de ser lido por humanos

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Assembly

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

O código de máquina pode ser traduzido em um código mnemônico conhecido como assembly. (ASM)

Código Assembly

```
31 c0                   xor    eax,eax
b8 28 23 00 00          mov    eax,0x2328
50                      push   eax
bb 10 90 12 76          mov    ebx,0x76129010
ff d3                   call   ebx
```

Código de máquina

Cada CPU tem seu próprio conjunto de instruções conhecido como: Instruction Set Architecture (ISA)

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Arquiteturas

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Cada CPU tem um conjunto de registradores que são pequenos locais para ler e manipular dados de uma forma extremamente rápida.

x86 - Processadores de 32 bits
x64 - Processadores de 64 bits (x86_64 / AMD64)

32 bits ou 4 bytes de largura
64 bits ou 8 bytes de largura

(lembrando: 1 byte = 8 bits)

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Registradores

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

64 bits	32 bits	16 bits	8 bits
RAX	EAX - Accumulator	AX	AH | AL
RBX	EBX - Base	BX	BH | BL
RCX	ECX - Counter	CX	CH | CL
RDX	EDX - Data	DX	DH | DL
RSI	ESI - Source Index	SI	
RDI	EDI - Destination Index	DI	
RSP	**ESP** - Stack Pointer	SP	
RBP	**EBP** - Base Pointer	BP	
RIP	**EIP - Instruction Pointer**	IP	
R8-R15 8 bytes	4 bytes	2 bytes	1 byte

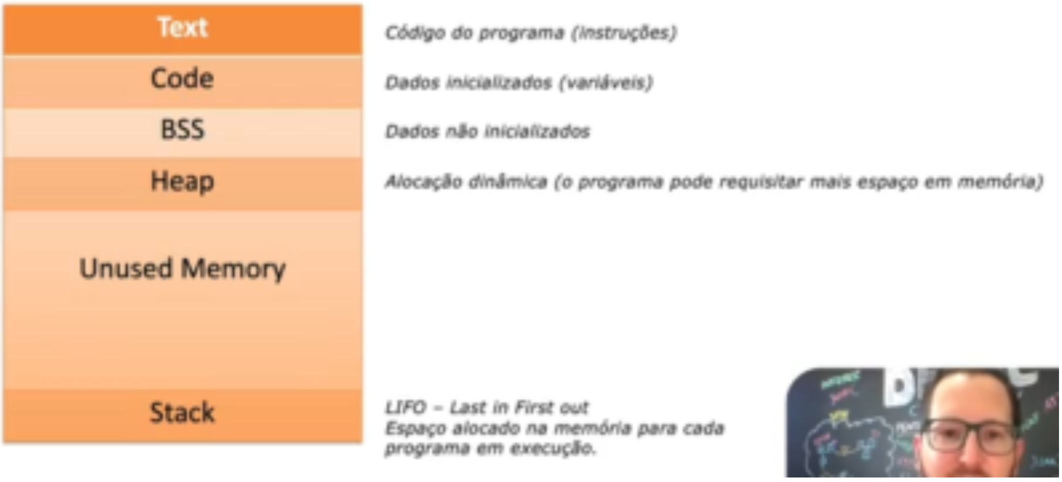
8 bits
H = HIGH | L = LOW
FFFF

→ Instruction Pointer = aponta para o próximo endereço a ser executado
→ o EIP é interessante para o desenvolvimento de um exploit pois ele quem determina o fluxo de um programa (de repente podemos direcionar o fluxo para uma shellcode)

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Processo em Memória
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~
Quando um processo executa ele é carregado e organizado na memória


~~~~~~