

C基础：内存布局

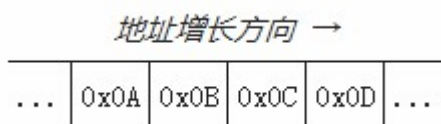
大端与小端

大端和小端是表示字节顺序的方式，又称端序或尾序(Endianness)

- 大端(big-endian)

- 高字节存储在低地址上，低字节存储在高地址上

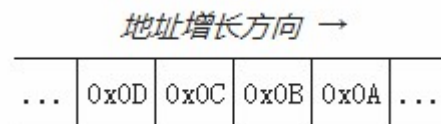
- 以8bit为单位的数据 0x0A0B0C0D 为例，



- 小端(little-endian)

- 高字节存储在高地址上，低字节存储在低地址上

- 以8bit为单位的数据 0x0A0B0C0D 为例，



现代PC机，以x86架构为基础的机器上，大多数都是采用了小端模式

C的内存模型和运行时内存布局

- BSS段(bss segment)

- 通常是指用来存放程序中**未初始化的**全局变量的一块内存区域，属于静态内存分配

- 数据段(data segment)

- 通常是指用来存放程序中**已初始化的**全局变量 的一块内存区域，属于静态内存分配

- 代码段(code segment/text segment)

- 通常代码段和只读数据段合成为文本段(text segment)，包含实际要执行的代码（机器指令）和常量

- 堆(heap)

- 堆是用于存放进程运行中被动态分配的内存段，它的大小并不固定，可动态扩张或缩减

- 栈(stack)

- 栈内存只在程序运行时出现，在函数内部使用的变量、函数的参数以及返回值将使用栈空间，栈空间由编译器自动分配和释放。其操作方式类似于数据结构中的栈

00000E51240	FF FF 43 89 9D D8 FB FF FF 83 FB 03 74 12 8B 9D	ÿÿC% Øÿÿÿfû t <
00000E51250	D4 FB FF FF E9 37 FF FF FF B9 01 00 00 00 EB 02	Ûÿÿÿé7ÿÿÿ¹ ë
00000E51260	33 C9 85 C9 BA 08 AA E9 00 B8 1C AA E9 00 0F 45	3É...É° ºé , ºé E
00000E51270	C2 50 E8 99 FD FF FF FF B5 D4 FB FF FF E8 8E F6	ÂPè"ÿÿÿÿÿÿÛÛÿÿÿèŽö
00000E51280	00 00 FF B5 D0 FB FF FF E8 73 F2 00 00 68 2C AA	ÿÿÛÛÿÿÿèsò h, º
00000E51290	E9 00 E8 62 F6 00 00 8B 4D FC 83 C4 10 33 CD 33	é èbö <MüfÄ 3í3
00000E512A0	C0 5F 5E 5B E8 37 00 00 00 8B E5 5D C3 68 EC A9	Ä_^[è7 <â]Ähi©
00000E512B0	E9 00 E8 D9 F3 00 00 83 C4 04 6A 01 E8 AF ED 00	é èÜó fÄ j è-í
00000E512C0	00 68 D8 A9 E9 00 E8 C5 F3 00 00 83 C4 04 6A 01	hØœé èÄó fÄ í

3. 看到引用字符串地址前有BA和B8，这是 mov edx, xxx 和 mov eax, xxx 的格式，在往后看看。传送了两个地址后面有个50表示 push eax 的操作，那么考虑改下这个push操作，因为最后打印成功的结果肯定有函数调用，那么这个push操作很有可能就是这次的调用。我们改成 push edx

00000E51240	FF FF 43 89 9D D8 FB FF FF	Login: user1
00000E51250	D4 FB FF FF E9 37 FF FF FF	Password: secret1
00000E51260	33 C9 85 C9 BA 08 AA E9 00	Login: 1
00000E51270	C2 52 E8 99 FD FF FF FF B5	Password: 1
00000E51280	00 00 FF B5 D0 FB FF FF E8	Login: 1
00000E51290	E9 00 E8 62 F6 00 00 8B 4D	Password: 1
00000E512A0	C0 5F 5E 5B E8 37 00 00 00	Login successful!
00000E512B0	E9 00 E8 D9 F3 00 00 83 C4	请按任意键继续. . .