**IDA逆向代码段说明**

.text段：text段是代码段，包含了程序的执行指令，即程序的可执行机器代码。 .idata段：idata段是导入数据段，包含了程序需要从外部动态链接库导入的函数和变量的信息。 .rdata段：rdata段是只读数据段，包含了程序中使用的常量数据，如字符串、静态初始化的数组 等，这些数据在程序运行期间不会被修改。 .data段：data段是数据段，包含了程序中使用的可读写数据，如全局变量和静态变量。

知识点：如果一个程序的入口地址为.text段大多不带壳，因为壳会将原来的EP（入口点）给覆盖。

**汇编知识 常见数据类型：**

- 字节型（DB，8个字节） - 字型（DW，16个字节） - 双字型（DD，32个字节） - 四字型（DQ，64个字节） - 十字节型

- 字符串型

**寄存器知识点以及常见汇编代码：**

CS:IP：任意时刻，CPU将CS:IP指向的内容当作指令执行 DS：通常存放要访问数据的段地址 SS:SP：任意时刻，SS:SP指向栈顶元素，SP - 2 loop：cx中存放循环的次数 offset：取得标号的偏移地址

函数调用：

push ebp mov ebp,esp sub esp,N

push xxx ret 就相当于 jmp xxx

**常见寄存器及作用：**

**寄存器** **描述**

EAX 通用寄存器，用于存放函数返回值、算术运算结果等。 EBX 通用寄存器，用于存放内存地址或其他通用数据。 ECX 通用寄存器，用于存放函数参数。

EDX 通用寄存器，用于存放内存地址或其他通用数据。

ESI 通用寄存器，用于存放源数据的地址，例如字符串拷贝等操作。 EDI 通用寄存器，用于存放目标数据的地址，例如字符串拷贝等操作。 EBP 基址指针，用于存放当前栈帧的基址。

ESP 堆栈指针，用于存放当前堆栈顶部的地址。

EIP 指令指针，用于存放当前正在执行的指令的地址。

EFLAGS 标志寄存器，用于存放CPU的状态标志，例如进位标志、零标志、符号标志 等。

**常用的宏：**

LOWORD：取32位数的低16位 HIWORD：取32位数的高16位 LOBYTE：取16位数的低8位 HIBYTE：去16位数的高8位

**函数通过参数输入：**

在C语言中，main 函数是程序执行的入口点。int \_\_cdecl main(int argc, const char \*\*argv, const char \*\*envp) 中的参数具有以下含义：

int argc：这是 “argument count” 的缩写，代表传递给程序的命令行参数的数量。这包括程序 名称本身，因此如果有一个命令行参数，argc 将是2。

const char \*\*argv：这是 “argument vector” 的缩写，代表一个指向字符串的指针数组，这 些字符串包含了传递给程序的命令行参数。argv[0] 通常包含程序的名称或者一个指向程序名称的 指针，argv[1] 是第一个命令行参数，依此类推。

const char \*\*envp：这是一个指向环境变量的指针数组。每个元素都是一个以 null 结尾的字符 串，形式为 “变量=值”，表示程序的环境设置。例如，envp 可能包含像 PATH=/usr/bin 或 HOME=/home/user 这样的字符串。

**最后**

更多逆向知识，在bilibili上有视频教程，欢迎大家参考学习。