DT\_STRTAB 🡪.dynstr (字符串表)

DT\_SYMTAB 🡪.dynsym (符号表)

DT\_JMPREL 🡪.rel.plt (PLT表)

**readelf -S 文件名** 来分析各个段(section)

**.dynsym(**符号表**)**里面记录了各种符号的信息，每个结构体对应一个符号。我们这里只关心函数符号

<符号表里面记录了目标文件所用到的所以符号，每个定义的符号又一个对应的值，叫做符号值。对于变量和函数来说，符号值就是他们的地址---我们将变量和函数统称为符号，函数名和变量名就是符号名>

typedef struct

{

Elf32\_Word st\_name; //符号名，是相对.dynstr起始的偏移，这种引用字符串的方式在前面说过了

Elf32\_Addr st\_value;

Elf32\_Word st\_size;

unsigned char st\_info; //对于导入函数符号而言，它是0x12

unsigned char st\_other;

Elf32\_Section st\_shndx;

}Elf32\_Sym; //对于导入函数符号而言，其他字段都是0

.rel.plt结构体数组，每个项对应一个导入函数

typedef struct

{

Elf32\_Addr r\_offset; //指向GOT表的指针

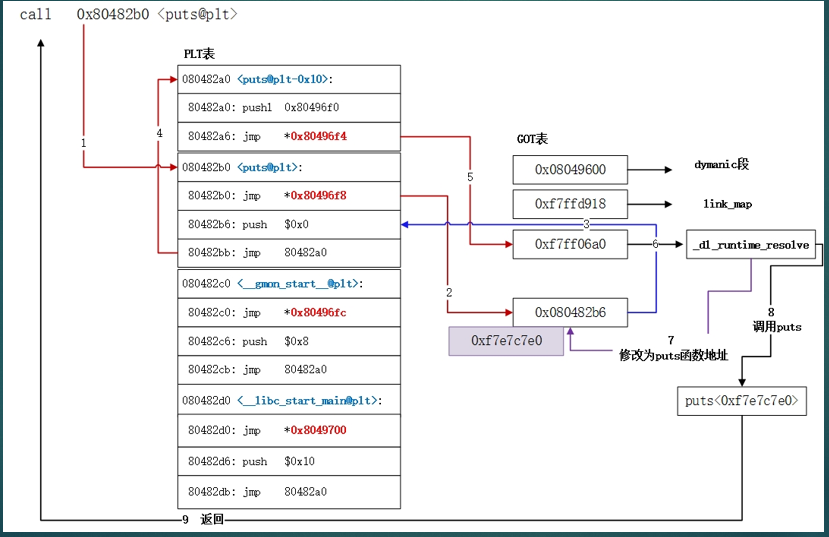
Elf32\_Word r\_info;

//一些关于导入符号的信息，我们只关心从第二个字节开始的值((val)>>8)，忽略那个07

//1和3是这个导入函数的符号在.dynsym中的下标，

//如果往回看的话你会发现1和3刚好和.dynsym的puts和\_\_libc\_start\_main对应

} Elf32\_Rel;



link\_map的指针，它包含了.dynamic的指针，通过这个link\_map，\_dl\_runtime\_resolve函数可以访问到.dynamic这个section。

**.dynamic**动态链接的关键信息

0x00000107>>8（即0x1）个元素

因为一个16进制数代表4个bit 刚好2个16

进制数代表8个bit，就相当于这个数向右移动两位数

(>>8表示将数字化作2进制后向右移动8位)