作者：代兴　邮箱：503268771@qq.com　Vx 公众号：东方架构师

最小化开发环境

工欲善其事，必先利其器。把开发环境搭建好，方便后续的学习。
本书使用最小化环境，把更多注意力集中在技术学习，减少干扰。

CPU 类型：64 位 x86 架构 CPU。
CPU 厂商包括 intel、AMD。

操作系统：Linux
推荐 centos7 。
如果使用 windows 或 mac，可以装个虚拟机，推荐 vmware。

```
[root@localhost x86-asm]# uname -a
Linux localhost.localdomain 3.10.0-1160.el7.x86_64 #1 SMP Mon Oct 19 16:18:59 UTC 2020 x86_64 x86_64
x86_64 GNU/Linux
```

编译器：gcc

```
[root@localhost x86-asm]# gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-redhat-linux/4.8.5/lto-wrapper
Target: x86_64-redhat-linux
Configured with: ../configure --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info
--with-bugurl=http://bugzilla.redhat.com/bugzilla --enable-bootstrap --enable-shared
--enable-threads=posix --enable-checking=release --with-system-zlib --enable-__cxa_atexit
--disable-libunwind-exceptions --enable-gnu-unique-object --enable-linker-build-id
--with-linker-hash-style=gnu --enable-languages=c,c++,objc,obj-c++,java,fortran,ada,go,lto
--enable-plugin --enable-initfini-array --disable-libgcj
--with-isl=/builddir/build/BUILD/gcc-4.8.5-20150702/obj-x86_64-redhat-linux/isl-install
--with-cloog=/builddir/build/BUILD/gcc-4.8.5-20150702/obj-x86_64-redhat-linux/cloog-install
--enable-gnu-indirect-function --with-tune=generic --with-arch_32=x86-64 --build=x86_64-redhat-linux
Thread model: posix
gcc version 4.8.5 20150623 (Red Hat 4.8.5-44) (GCC)
```

调试器：gdb

```
[root@localhost x86-asm]# gdb -v
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-120.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
```

代码编辑器：
VS code，安装 ssh 插件，连接 linux。
Vim，熟练掌握，用命令行小范围编辑更方便。

反汇编工具：objdump
从程序反向生成汇编代码。

```
[root@localhost x86-asm]# objdump -v
GNU objdump version 2.27-44.base.el7
Copyright (C) 2016 Free Software Foundation, Inc.
This program is free software; you may redistribute it under the terms of
the GNU General Public License version 3 or (at your option) any later version.
This program has absolutely no warranty.
```

解析 ELF 文件：readelf

```
[root@localhost x86-asm]# readelf -v
GNU readelf version 2.27-44.base.el7
Copyright (C) 2016 Free Software Foundation, Inc.
This program is free software; you may redistribute it under the terms of
the GNU General Public License version 3 or (at your option) any later version.
This program has absolutely no warranty.
```

用 hello 程序测试开发环境

编写代码：dev.c

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

// 变量
char *num_str = "hello , num = %d \n";

// 方法
void print_num(int num)
{
    printf(num_str, num);
}

// 主函数
int main()
{
    print_num(7766);
    return 0;
}
```

编译代码：

```
# 编译为可执行程序
```

```
gcc dev.c -o dev

# 编译为汇编文件
gcc dev.c -S -o dev.s

# 查看 ELF 信息
readelf -a dev > dev.elf.txt

# 查看反汇编信息
objdump -D dev > dev.dump.txt
```

运行代码：

```
[root@localhost dev]# ./dev
hello , num = 7766
```

分析结果：
后续章节做深入讨论。
源文件 dev.c ，有 20 行。
汇编文件 dev.s ，有 58 行。
ELF 信息 dev.elf.txt ，有 260 行。
反汇编信息 dev.dump.txt ，有 785 行。

函数 print_num 的源代码

```
// 变量
char *num_str = "hello , num = %d \n";

// 方法
void print_num(int num)
{
    printf(num_str, num);
}
```

函数 print_num 的汇编代码

```
print_num:
.LFB2:
    .cfi_startproc
    pushq   %rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq    %rsp, %rbp
    .cfi_def_cfa_register 6
    subq    $16, %rsp
    movl    %edi, -4(%rbp)
    movq    num_str(%rip), %rax
    movl    -4(%rbp), %edx
    movl    %edx, %esi
    movq    %rax, %rdi
    movl    $0, %eax
    call    printf
```

```
    leave
    .cfi_def_cfa 7, 8
    ret
```

函数 print_num 的 ELF 信息

```
Symbol table '.symtab' contains 65 entries:
   Num:    Value          Size Type    Bind   Vis      Ndx Name
    55: 0000000000601038     8 OBJECT  GLOBAL DEFAULT   24 num_str
    62: 000000000040052d    38 FUNC    GLOBAL DEFAULT   13 print_num
```

函数 print_num 的反汇编信息

```
000000000040052d <print_num>:
  40052d:   55                      push   %rbp
  40052e:   48 89 e5                mov    %rsp,%rbp
  400531:   48 83 ec 10             sub    $0x10,%rsp
  400535:   89 7d fc                mov    %edi,-0x4(%rbp)
  400538:   48 8b 05 f9 0a 20 00    mov    0x200af9(%rip),%rax        # 601038 <num_str>
  40053f:   8b 55 fc                mov    -0x4(%rbp),%edx
  400542:   89 d6                   mov    %edx,%esi
  400544:   48 89 c7                mov    %rax,%rdi
  400547:   b8 00 00 00 00          mov    $0x0,%eax
  40054c:   e8 bf fe ff ff          callq  400410 <printf@plt>
  400551:   c9                      leaveq
  400552:   c3                      retq
```