

**汇编语言与逆向技术**

**课程实验报告**

**实验五：ARM平台-HelloWorld**

****

学 院 网络空间安全

专 业 信息安全

学 号 2212998

姓 名 胡博浩

班 级­­­­ 2022级信息安全

1. **实验目的**
   1. 理解GNU ARM 汇编代码运行环境的搭建、配置及编译运行，掌握在华为鲲鹏云服务器上进行环境配置
   2. 命令行输出“HelloWorld”
2. **实验步骤**
3. **创建 hello 目录**

执行以下命令，创建 hello 目录，存放该程序的所有文件, 并进入 hello 目录。

mkdir hello

cd hello

1. **创建示例程序代码hello.s**

执行以下命令，创建程序源码 hello.s。

vim hello.s

代码及解析如下：

.text

.global \_start

\_start:

mov x0, #0 // 将文件描述符设置为0（表示标准输入）

ldr x1, =msg // 将消息字符串的地址加载到寄存器x1中

mov x2, len // 将消息的长度加载到寄存器x2中

mov x8, 64 // 用于写入的系统调用号（64 对应 sys\_write）

svc #0 // 发起系统调用以将消息写入标准输出

mov x0, 123 // 将一个任意值加载到寄存器x0中

mov x8, 93 // 用于退出的系统调用号（93 对应 sys\_exit）

svc #0 // 发起系统调用以退出程序

.data

msg:

.ascii "Hello World!\n" // 实际的消息字符串

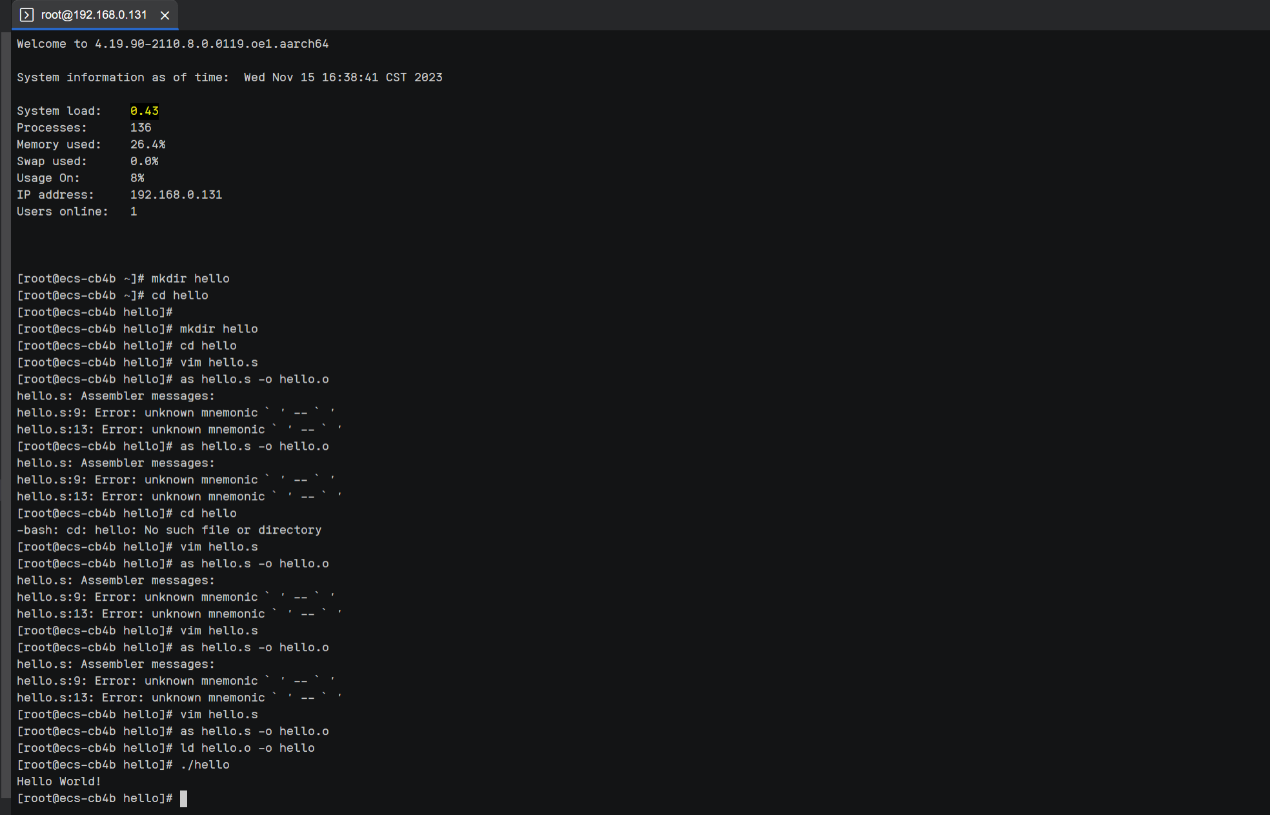
len = . - msg // 计算消息的长度

这段代码的主要作用是：

通过系统调用号为 64（sys\_write）将 "Hello World!\n" 消息写入标准输出。

使用系统调用号为 93（sys\_exit）退出程序，寄存器 x0 中的值（在这里是123）被用作退出码。

1. **进行编译运行**

保存示例源码文件，然后退出 vim 编辑器。在当前目录中依次执行以下命令，进行代码编译运行。

as hello.s -o hello.o

ld hello.o -o hello

./hello

通过上述代码运行，可以看出，编写的 hello-wolrd 示例程序已经在华为鲲鹏云服务器上通过编译和运行，并成功输出结果。

1. **思考题：同样的代码能否在x86平台运行，为什么？**

这段代码是为ARM64架构编写的，不能直接在x86平台上运行。主要原因是两种架构的指令集和系统调用接口是不同的。

在x86平台上，通常使用的是x86或x86-64（也称为AMD64）架构。这两者的指令集与ARM64是不同的，因此需要相应调整代码以适应不同的架构。

另外，不同的操作系统也可能有不同的系统调用接口。在Linux上，x86平台和ARM64平台的系统调用编号和调用方式也是不同的。

如果想在x86平台上运行相似的程序，需要重新编写代码，使用x86或x86-64的指令集，并遵循相应平台的系统调用接口。