GDB调试相关

调试相关

gdb 可执行程序

gdb --args 可执行程序 传入参数

gdb

file 可执行程序路径

b 函数名

b 文件名：行号

b 本文件行号

条件断点

{ b 相关语句} {if 语句}

cl 行号 //消除该行断点

命令 作用

l 列出本行上下文代码

s 逐步调试代码

n 逐行调试代码

r 开始运行

c 继续向下执行

p 打印

print xxx //打印变量

p /x xxx //16进制显示

p str@str\_len //打印字符串

还有个更灵活强大的是直接打印%esp开始的前N个元素，例如打印栈上前10个元素就是：x/10x $sp

格式: x /nfu <addr> x 是 examine 的缩写   
a.n表示要显示的内存单元的个数   
b.f表示显示方式, 可取如下值  
(1).x 按十六进制格式显示变量。  
(2).d 按十进制格式显示变量。  
(3).u 按十进制格式显示无符号整型。  
(4).o 按八进制格式显示变量。  
(5).t 按二进制格式显示变量。  
(6).a 按十六进制格式显示变量。  
(7).i 指令地址格式  
(8).c 按字符格式显示变量。  
(9).f 按浮点数格式显示变量。  
c.u表示一个地址单元的长度  
(1).b表示单字节，  
(2).h表示双字节，  
(3).w表示四字节，  
(4).g表示八字节

比如：x/3xh buf   
表示从内存地址buf读取内容，3表示三个单位，x表示按十六进制显示，h表示以双字节为一个单位。

9.生成环境使用GDB场景

内核转储(coredump)

(1).配置产生core文件

前置条件：确保系统配置的core file size足够，一般设置成unlimited  
ulimit -c unlimited

配置corefile的参数：  
echo 2 > /proc/sys/fs/suid\_dumpable [程序中切换用户，也要产生corefile]  
mkdir /tmp/corefiles   
chmod 777 /tmp/corefiles  
echo "/tmp/corefiles/core">/proc/sys/kernel/core\_pattern //配置core文件产生的目录为/tmp/corefiles  
echo "1" > /proc/sys/kernel/core\_uses\_pid

注意：  
a.确保配置的目录有足够的磁盘空间，否则产生core文件可能不完整。  
b.对于mysqld而言，要保证正确产生core-file，需要加上—core-file，默认这个参数是不打开的。

c.kill -9 pid 是不能产生core文件的，因为SIGKILL信号不能被捕获。

(2).使用core文件

gdb /usr/mysql/bin/mysqld core.24556

关于堆栈

bt：bt是backtrace的缩写，打印当前所在函数的堆栈路径。   
info frame id：打印选中的栈帧的信息。   
info args：打印选中栈帧的参数。   
print variable：打印指定变量的值。   
list：列出相应的源代码。   
info registers：查看所有寄存器的值。

#### 查询运行信息

* where/bt：查看当前运行的堆栈列表
* up/down：改变堆栈显示的深度
* set args：参数:指定运行时的参数
* show args：查看设置好的参数
* info program： 来查看程序的是否在运行，进程号，被暂停的原因。

#### 分割窗口

* layout：用于分割窗口，可以一边查看代码，一边测试
* layout src：显示源代码窗口
* layout asm：显示反汇编窗口
* layout regs：显示源代码/反汇编和CPU寄存器窗口
* layout split：显示源代码和反汇编窗口
* Ctrl + L：刷新窗口