Linux 调试工具GDB入门

Version 1.0

西安電子科技大學

需要掌握的要点

- 理解软件调试技术的重要性:
  - 找出程序运行时错误：错误使人进步
  - 成为编程高手的必经之路（调试高手）
  - 加深对工作原理和工作机制的理解
- 了解GBD工具的基本原理
  - 本地调试
  - 远程调试
  - 可视化调试工具DDD、Eclipse
- 掌握GBD工具的基本用法

西安電子科技大學

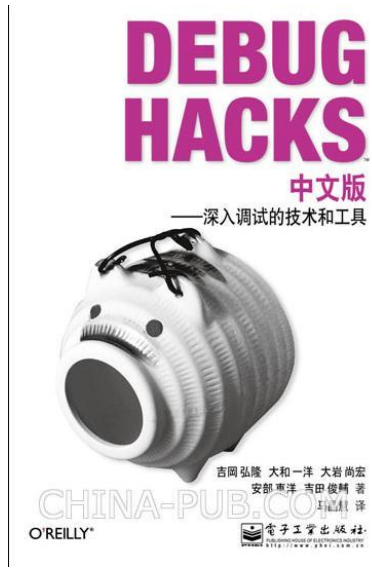
软件调试技术的重要性

- 找出程序运行时错误：错误使人进步
- 成为编程高手的必经之路（调试高手）
- 加深对工作原理和工作机制的理解：
  - 编程语言、编译原理、操作系统、微机原理等

软件调试技术参考书

软件调试技术参考书

软件调试技术参考书



软件调试技术参考书

GDB工具

- GDB = GNU debuger

功能：

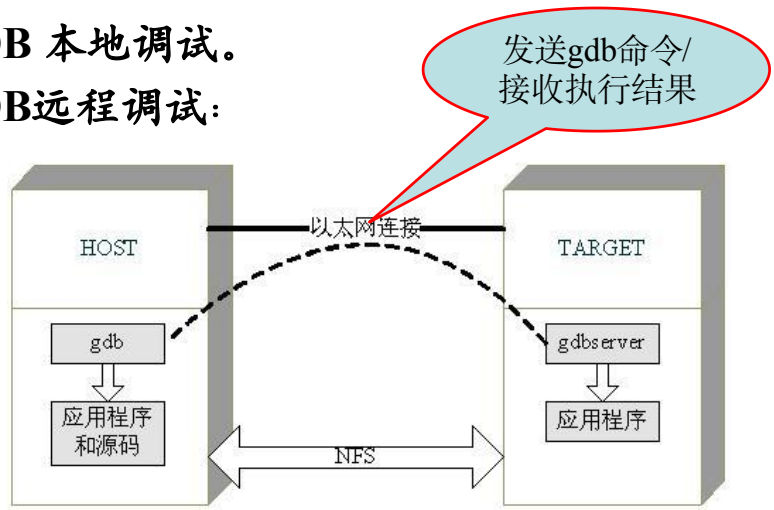
- 设置断点
- 更改程序流向
- 监视程序运行时数据（变量、内存等）
- 监视其他信息（寄存器、环境变量等）
- 程序运行时动态修改数据（变量值、信号量）
- 调用回溯
- core dump分析

GDB的工作原理

- GDB 本身也是一个应用程序。
- 当利用gdb program（program是被调试的目标程序）命令启动gdb程序后：
  - gdb首先将program装入内存；
  - 找到program的源程序并利用符号表信息建立内存进程镜像与源程序的关联
  - 为program创建进程上下文，但并不立即执行该进程
  - gdb进入命令行模式，等待用户输入命令
  - 根据用户命令进行插入断点、查看内存内容等操作
  - 根据用户命令运行program的全部或部分指令

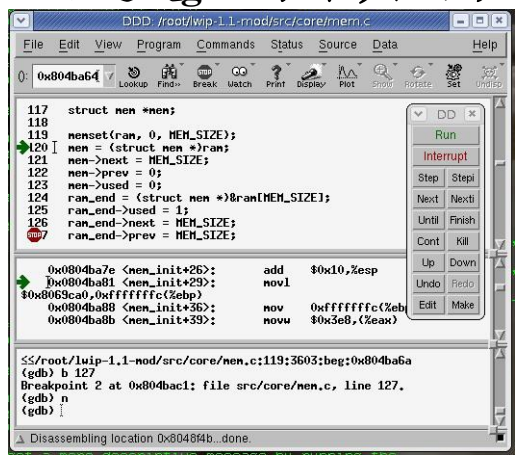
GDB的工作原理

- GDB 本地调试。
- GDB远程调试：



GDB的图形化调试

- DDD（Data Display Debuger）和Eclipse调试插件只不过是gdb的图形化客户端



gdb使用流程

- 将要调试的程序编译为调试版(gcc –g)
- 启动GDB ——gdb program
- 运行GDB命令：
  - 查看文件——list
  - 设置断点——b 6
  - 查看断点处情况——info b
  - 运行代码——r
  - 查看变量值——p n / p i
  - 单步运行——s或n
  - 恢复程序运行——c 或 f
  - 观察变量——watch n
- 退出GDB ——q

启动GDB开始调试

- A.准备工作

编译调试版本的可执行程序(gcc加上-g或-ggdb3参数即可,注意不要调试加-O相关的选项)

- B.冷启动

gdb *program* e.g., gdb ./cs

gdb –p *pid* e.g., gdb –p `pidof cs`

gdb *program core* e.g., gdb ./cs core.xxx

- C.热启动

(gdb) attach *pid* e.g., (gdb) attach 2313

- D. 命令行参数

gdb *program* --args *arglist*

(gdb) set args *arglist*

(gdb) run *arglist*

GDB常用命令——断点

- break <function>
- break <linenum>
- break +offset
- break -offset
- break filename:linenum
- break filename:function
- break *address
- break
- break ... if <condition>

GDB常用命令—— 断点

- tbreak 临时断点
- condition <bnum> <expression>断点附加条件
- watch <expr>
- rwatch <expr>
- awatch <expr>

GDB常用命令—— 断点

- clear
- delete
- disable
- enable

GDB常用命令—— 更改程序流向

- step/stepi
- next/nexti
- continue/continue <N>
- finish

- jump <linenum> /jump <addr>
- call <func>
- return / return <return value>

GDB常用命令—— 监视程序运行时数据

- 查看程序运行时对应表达式或变量的值

  print /[fmt] <expr>
- 设定在单步运行后其他情况下，自动显示的对应表达式的内容

  display /[fmt] <expr>

  display /[fmt] <addr>
- 查看内存变量内容

  examine [/n/f/u] <addr>

GDB常用命令—— 监视其他信息

- backtrace
- info registers
- info frame
- info args
- info locals

GDB常用命令—— 动态修改程序数据

- set val <valname>

GDB常用命令—— 其他

- list / list <N> / list <func>列出源程序
- disas <func> / disas <addr> 反汇编
- forward-search <regexpr>
- reverse-search <regexpr>

gdb基本命令

1. 工作环境相关命令

命 令 格 式	含 义
set args运行时的参数	指定运行时参数，如set args 2
show args	查看设置好的运行参数
path dir	设定程序的运行路径
show paths	查看程序的运行路径
set environment var [=value]	设置环境变量
show environment [var]	查看环境变量
cd dir	进入到dir目录，相当于shell中的cd命令
pwd	显示当前工作目录
shell command	运行shell的command命令

gdb基本命令

2. 设置断点与恢复命令

命令格式	含　义
info b	查看所设断点
break [文件名:]行号或函数名<条件表达式>	设置断点
tbreak [文件名:]行号或函数名<条件表达式>	设置临时断点，到达后被自动删除
delete [断点号]	删除指定断点，其断点号为“info b”中的第一栏。若缺省断点号则删除所有断点
disable [断点号]	停止指定断点，使用“info b”仍能查看此断点。同delete一样，若缺省断点号则停止所有断点
enable [断点号]	激活指定断点，即激活被disable停止的断点
condition [断点号] <条件表达式>	修改对应断点的条件
ignore [断点号]<num>	在程序执行中，忽略对应断点num次
step	单步恢复程序运行，且进入函数调用
next	单步恢复程序运行，但不进入函数调用
finish	运行程序，直到当前函数完成返回
c	继续执行函数，直到函数结束或遇到新的断点

gdb基本命令

3. gdb中源码查看相关命令

命令格式	含　义
list <行号>|<函数名>	查看指定位置代码
file [文件名]	加载指定文件
forward-search 正则表达式	源代码的前向搜索
reverse-search 正则表达式	源代码的后向搜索
dir DIR	将路径DIR添加到源文件搜索的路径的开头
show directories	显示源文件的当前搜索路径
info line	显示加载到gdb内存中的代码

gdb基本命令

4. gdb中查看运行数据相关命令

命令格式	含　义
print 表达式|变量	查看程序运行时对应表达式和变量的值
x <n/f/u> address	查看内存变量内容。其中n为整数表示显示内存的长度，f表示显示的格式，u表示从当前地址往后请求显示的字节数
display 表达式	设定在单步运行或其他情况中，自动显示的对应表达式的内容
backtrace或bt	查看当前栈侦的情况，即可以查到哪些被调用的函数尚未返回。
frame *n*	打印第*n*个栈侦
info reg/stack	查看寄存器/堆栈使用情况
up	调到上一层函数，即上移栈帧
down	与up相对，即下移栈帧

gdbserver远程调试

- 使用交叉调试工具实现远程调试。
- gdb调试器提供了两种不同的远程调试方法，即stub（插桩）方式和gdbserver方式。
- gdbserver本身的体积很小,能够在具有很小内存的目标系统上独立运行，因而非常适合嵌入式开发。
- stub方式则需要通过链接器把调试代理和要调试的程序链接成一个可执行的应用程序文件，而且stub需要修改异常处理和驱动程序等。
- gdbserver要求宿主机和目标系统采用同一系列的操作系统，而stub没有这种限制，甚至目标系统可以没有操作系统。gdbserver比较适合于调试嵌入式平台上的应用程序，而stub比较适合于调试bootloader和内核等系统程序。

gdbserver远程调试

- 用gdb+gdbserver的方式调试嵌入式平台上的Linux应用程序
  - 安装arm-linux-gdb
  - 安装gdbserver
  - 远程调试

- 上机过程中演示讲解