联航精英训练营

UNIGRESS ELITE TRAINING CAMP

Linux开发基础-GDB

Gdb手册

- http://sourceware.org/gdb/current/onlinedocs/gdb/

gdb 简介

- Linux系统中包含了GNU 调试程序gdb，它是一个用来调试C和 C++ 程序的调试器。可以使程序开发者在程序运行时观察**程序的内部结构**和**内存**的使用情况。

- gdb 所提供的一些功能如下所示：
  - 运行程序，设置所有的能影响程序运行的参数和环境；
  - 控制程序在指定的条件下停止运行；
  - 当程序停止时，可以检查程序的状态；
  - 修改程序的错误，并重新运行程序；
  - 动态监视程序中变量的值；
  - 可以单步逐行执行代码，观察程序的运行状态。

gdb的特点

- gdb的功能非常强大
  - 到目前为止，gdb已能够支持Moduls-2、Chill、Pascal和FORTRAN程序的调试，但是调试这些语言的源程序时有一些功能还不能使用。
  - 例如调试FORTRAN程序时还不支持表达式的输入、输出变量或类FORTRAN的词法。

- gdb程序调试的对象是**可执行文件**，而不是程序的源代码文件。
- 然而，并不是所有的可执行文件都可以用gdb调试。如果要让产生的可执行文件可以用来调试，需在执行gcc指令编译程序时，加上**-g**参数，指定程序在编译时包含调试信息。
- 调试信息包含程序里的每个变量的类型和在可执行文件里的地址映射以及源代码的行号。
- gdb 利用这些信息使源代码和机器码相关联。 gdb是一个用来调试C和C++程序的常用调试工具之一。

GDB 调试程序一般流程

1. 以产生含有可调试信息的方式编译应用程序 [-g]
2. 将含有调试信息的应用程序加载到gdb [gdb a.out]
3. gdb中查看源程序 [l/list]
4. 设置断点/观察点等[b 行号]
5. 设置运行时使用参数[]
6. 在gdb中运行应用程序[run]
7. 调试应用程序
8. 使用s[单步] n[逐过程] fin until调试跟踪
9. 查看变量值、数组等
10. 退出gdb，结束调试

gdb的启动

- 在命令行上输入gdb并按回车键就可以运行gdb了，如果一切正常的话，将启动gdb

  gdb [filename]

  出现

  (gdb)

  在这里，可以输入调试命令

- 在可以使用 gdb 调试程序之前，必须使用 -g 选项编译源文件。可在 makefile 中如下定义 CFLAGS 变量：
  CFLAGS = -g 运行

获取帮助信息

- 启动gdb后，可以在命令行上指定很多的选项。输入：

  help

  可以获得gdb的帮助信息。

- 如果想要了解某个具体命令（比如break）的帮助信息，在gdb提示符下输入下面的命令：

  help break

  屏幕上会显示关于break的帮助信息。从返回的信息可知，break是用于设置断点的命令。

- 另一个获得gdb帮助的方法是浏览gdb的手册页。在Linux Shell提示符输入：

  man gdb

  可以看到man的手册页

gdb命令的分类

- 在 gdb 提示符处键入help，将列出命令的分类，主要的分类有：
  aliases：命令别名
  breakpoints：断点定义；
  data：数据查看；
  files：指定并查看文件；
  internals：维护命令；
  running：程序执行；
  stack：调用栈查看；
  statu：状态查看；
  tracepoints：跟踪程序执行。

基本gdb命令（1/2）

file命令：装入想要调试的可执行文件。

cd命令：改变工作目录。

pwd命令：返回当前工作目录。

run命令：执行当前被调试的程序。

kill命令：停止正在调试的应用程序。

list命令：列出正在调试的应用程序的源代码。

break命令：设置断点。

watch命令：设置监视点，监视表达式的变化。

awatch命令：设置读写监视点。当要监视的表达式被读或写时将应用程序挂起。它的语法与watch命令相同。

rwatch命令：设置读监视点，当监视表达式被读时将程序挂起，等待调试。此命令的语法与watch相同。

next命令：执行下一条源代码，但是不进入函数内部。也就是说，将一条函数调用作为一条语句执行。执行这个命令的前提是已经run，开始了代码的执行。

基本gdb命令（2/2）

step命令：执行下一条源代码，进入函数内部。如果调用了某个函数，会跳到函数所在的代码中等候一步步执行。执行这个命令的前提是已经用run开始执行代码。

display命令：在应用程序每次停止运行时显示表达式的值。

info break命令：显示当前断点列表，包括每个断点到达的次数

info files命令：显示调试文件的信息。

info func命令：显示所有的函数名。

info local命令：显示当前函数的所有局部变量的信息。

info prog命令：显示调试程序的执行状态。

print命令：显示表达式的值。

delete命令：删除断点。指定一个断点号码，则删除指定断点。不指定参数则删除所有的断点。

Shell命令：执行Linux Shell命令。

make命令：不退出gdb而重新编译生成可执行文件。

Quit命令：退出gdb。

查看程序执行状态

- info program

查看源代码

list <linenum>
list <function>
list <filename:linenum>
list <filename:function>
list

停/恢复程序运行-设置断点

break <function>
break <linenum>
break +/-offset
break filename:linenum
break filename:function
i b 查看断点

停/恢复程序运行-设置断点

break

break [] if <condition>

查看断点

info breakpoints [n]

info break[n]

停/恢复程序运行-关停使能断点

- enable 断点号
- disable 断点号

停/恢复程序运行 – 删除断点

- delete:使用的是断点号
- clear: 使用的是行号

停/恢复程序执行-设置观察点

watch命令：设置监视点，监视表达式的变化。

awatch命令：设置读写监视点。当要监视的表达式被读或写时将应用程序挂起。它的语法与watch命令相同。

rwatch命令：设置读监视点，当监视表达式被读时将程序挂起，等待调试。此命令的语法与watch相同。

info wtachpoints: 查看当前设置了哪些观察点

GDB调试

step :单步调试
next 单步调试,但不进入要调用的函数
until: 运行程序直到退出循环体
finish:运行程序直到从函数中返回
return value:停止函数运行,返回value
stepi/nexti 机器指令的单步执行

查看运行时的表达式或变量值

- print
- “@” 用于数组显示
- “::” 用于指定一个函数中的变量的现实

print fun1::var0

查看数组

print array1

print *array@10

自动显示

display <expr>

undisplay <dnums>

推荐阅读材料

- GNU/Linux编程指南
- GDB手册
- Debugging with GDB
- Linux software debugging with GDB

专业铸就品质 梦想成就未来

The Specialty Casts Quality,the Dream Gains Future.