gdb调试器

1. gdb功能介绍

gdb（GNU debugger）是GNU开发组织发布的一个强大的UNIX/Linux下的调试工具。一般来说，gdb提供了以下3个方面的功能：

* 监视程序中变量的值
* 设置断点以使程序在指定的代码上停止执行
* 一行行地执行代码

实例：**test.c**

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  int func(int n) {  int sum = 0,i;  for(i=0;i<n;i++) {  sum+=i;  }  return sum;  }  main() {  int i;  long result = 0;  for(i=1;i<=100;i++)  {  result += 100;  }  printf("result[1-100] = %d\n",result);  printf("result[1-250] = %d\n",func(250));  } |

编译生成执行文件：gcc –g test.c –o test

启动调试：gdb test

列出源代码：list

设置断点，在源代码第16行：break 16

设置断点，在函数func()入口处：break func

查看断点信息：info break

运行程序：r（run）

单条语句执行：n（next）

继续运行程序：c（continue）

打印变量i的值：p（print） i

查看函数堆栈：bt

退出函数：finish

退出调试：q（quit）

总结一下，gdb的基本命令如表1所示：

表1 gdb的基本命令

|  |  |
| --- | --- |
| 命令 | 说明 |
| file | 装入想要调试的可执行文件 |
| kill | 终止正在调试的进程 |
| list | 列出产生执行文件的源代码的一部分 |
| next | 执行一行源代码，但是不进入函数内部 |
| step | 执行一行源代码，而且进入函数内部 |
| run | 执行当前被调试的程序 |
| quit | 退出gdb |
| watch | 动态监视一个变量的值 |
| make | 不退出gdb而重新产生可执行文件 |
| call name（args） | 调用并执行名为name，参数为args的函数 |
| return value | 停止执行当前函数，并将值value返回给调用者 |
| break | 在代码里设置断点，使程序执行到此处被挂起 |

1. gdb的调用

通常来说，调用gdb只需要使用一个参数。

gdb <可执行程序名>

如果程序运行时产生了段错误，会在当前目录下产生核心内存映像core文件，可以在指定执行文件的同时为可执行程序指定一个core文件。

gdb <可执行文件名> core

除此之外，还可以为要执行的文件指定一个进程号。

gdb <可执行文件名> <进程号>

例：使用gdb为test指定进程号

gdb test 2000

首先，gdb会去寻找一个进程号为2000的文件，如果找不到，则把调试程序的进程号设置为2000。

当gdb运行时，把任何一个不带选项前缀的参数都作为一个可执行文件或core文件，或者要与被调试的程序相关联的进程号。不带任何选项前缀的参数和前面加了-se或-c选项的参数效果一样。gdb把第一个前面没有选项说明的参数看作前面加了-se选项，也就是需要调试的可执行文件，并从此文件了读取符号表。如果有第二个前面没有选项说明的参数，将被看作是跟在-c选项后面，也就是需要调试的core文件名。

如果不希望看到gdb开始的提示信息，可以用gdb-silent执行调试工作，通过更多的选项，开发者可以按照自己的喜好定制gdb的行为。

输入gdb --help或者gdb –h可以得到gdb启动时的所有选项提示。gdb命令行中的所有参数都被按照排列的顺序传递给gdb，出发使用了-x参数。

gdb的许多选项都可以用缩写形式代表，可以用-h查看相关缩写。在gdb中也可以采用任意长度的字符串代表选项，只要保证gdb能唯一地识别此参数就行。gdb的常用参数选项如表2所示：

表2 gdb常用的参数选项

|  |  |
| --- | --- |
| 选项 | 说明 |
| -s filename | 从filename指定的文件中读取要调试的程序的符号表 |
| -e filename | 在合适的时候执行filename指定的文件，并通过与core文件进行比较来检查正确的数据 |
| -se filename | 从filename中读取符号表，并作为可执行文件进行调试 |
| -c filename | 把filename指定的文件作为一个core文件 |
| -c num | 把数字num作为进程号和调试的程序进行关联，与attach命令相似 |
| -command filename | 按照filename指定的文件中的命令执行gdb命令，在filename指定的文件中存放着一系列的gdb命令，就像是一个批处理 |
| -d path | 指定源文件的路径。把path加入到搜索源文件的路径中 |
| -r | 从符号文件中一次读取整个符号表，而不是使用默认的方式首先调入一部分符号，当需要时再读入其他一部分。这会使gdb的启动较慢，但可以加快以后的调试速度 |

1. gdb运行模式的选择

可以用许多模式来运行gdb，例如，采用“批模式”或“安静模式”。这些模式都是在gdb运行时在命令行中通过选项来指定的。

gdb运行模式的相关选项如表3所示:

表3 gdb运行模式选项

|  |  |
| --- | --- |
| 选项 | 说明 |
| -n | 不执行任何初始化文件中的命令（一般初始化文件为.gdbinit）。一般情况下，在这些文件中的命令会在所有的命令行参数都被传给gdb后执行 |
| -q | 设定gdb的运行模式为“安静模式”，可以不输出介绍和版权信息。这些信息在“批模式”中也不会显示 |
| -batch | 设定gdb的运行模式为“批模式”。gdb在“批模式”下运行时，会执行命令文件中的所有命令，当所有命令都被成功地执行后，gdb返回状态0，如果在执行过程中出错，gdb返回一个非零值。 |
| -cd dir | 把dir作为gdb的工作目录，而非当前目录（一般gdb默认把当前目录作为工作目录） |