Ferramentas de desenvolvimento

Breve Introdução

José Pedro Oliveira
(jpo@di.uminho.pt)

Grupo de Sistemas Distribuídos
Departamento de Informática
Escola de Engenharia
Universidade do Minho

Sistemas Operativos I
2006-2007

Conteúdo

Compilador de C - gcc

Synopsis

```
gcc [ opções ] ficheiro ...
```

Algumas opções

-E	- Executar apenas o passo de pré-processamento
-S	- Gerar o ficheiro assembly (.s)
-c	- Compilar apenas
-Wall	- Activa a grande maioria dos avisos
-Wextra	- Activa ainda mais avisos
-Werror	- Os avisos passam a ser considerados erros
-g	- Gerar informação de debugging
-On	- Nível de optimização (por omissão: -O0)
-Dmacro=valor	- Definir macro

Exemplo de um programa C

warnings.c

```c
#include <stdio.h>

int
main(int argc, char *argv[])
{
    printf("Sistemas Operativos.\n");

    return 0;
}

/* vim:set ai ts=4 sw=4 sts=4 et: */
```

Versão

$ gcc --version

```
gcc (GCC) 4.0.2 20051125 (Red Hat 4.0.2-8)
Copyright (C) 2005 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.
There is NO warranty; not even for MERCHANTABILITY or FITNESS
FOR A PARTICULAR PURPOSE.
```

$ gcc -v

```
Target: i386-redhat-linux
Configured with: ../configure --prefix=/usr
--mandir=/usr/share/man --infodir=/usr/share/info
--enable-shared --enable-threads=posix --enable-checking=release
--with-system-zlib --enable-__cxa_atexit
--disable-libunwind-exceptions --enable-libgcj-multifile
--enable-languages=c,c++,objc,java,f95,ada --enable-java-awt=gtk
--with-java-home=/usr/lib/jvm/java-1.4.2-gcj-1.4.2.0/jre
--host=i386-redhat-linux
Thread model: posix
gcc version 4.0.2 20051125 (Red Hat 4.0.2-8)
```

Nome do binário gerado

Opção

-o ficheiro - Especificar nome do binário

$ gcc warnings.c

Binário gerado: **a.out** (nome usado por omissão)

$ gcc -o warnings warnings.c

Binário gerado: **warnings**

Standard C

Opção

-std=*standard* - Especficar o standard da linguagem C

Alguns dos standards disponíveis

- c89 - ISO C90 (o mesmo que -ansi)
- c99 - ISO C99
- gnu89 - ISO C90 mais extensões GNU (por omissão)
- gnu99 - ISO C99 mais extensões GNU

Exemplo

```
gcc -std=c99 ...
```

Avisos - exemplos de algumas opções

$ gcc -Wall warnings.c

Binário gerado: **a.out**

$ gcc -Wall -Wextra warnings.c

Binário gerado: **a.out**
Avisos:

```
warnings.c:4: warning: unused parameter 'argc'
warnings.c:4: warning: unused parameter 'argv'
```

$ gcc -Wall -Wextra -Werror warnings.c

Os avisos são considerados erros. Nenhum binário é gerado.

Avisos do tipo *unused*

Solução 1 - alterar interface

```
int
main(void)
{
```

Solução 2 - utilizar atributos

```
int
main(int __attribute__((unused)) argc,
     char __attribute__((unused)) *argv[])
{
```

Símbolos pré-definidos

$ echo ”int main(void) { return 0; }” | gcc -E -dM -

```
...
#define __GNUC__ 4
#define __GNUC_MINOR__ 0
#define __GNUC_PATCHLEVEL__ 2
#define __GNUC_RH_RELEASE__ 8
#define __gnu_linux__ 1
#define __GXX_ABI_VERSION 1002
#define __i386 1
#define __i386__ 1
...
#define __linux 1
#define __linux__ 1
...
```

Ficheiros header (.h)

Opção

-I*dir* - Especficar directórios de ficheiros header adicionais

Especificação de ficheiros header

- **#include** <**header.h**> - Ficheiro header de uma biblioteca standard ou de sistema
- **#include ”header.h”** - Ficheiro header privado (do projecto)

Exemplos

```
gcc -I/usr/X11R6/include ...
gcc -I..  -I../modulo1 -I../modulo2 ...
```

Bibliotecas

Bibliotecas estáticas e dinâmicas

Estáticas - ficheiros com extensão .a

Dinâmicas - ficheiros com extensão .so

Opções

-L*dir* - Especificar directórios de bibliotecas adicionais

-l*lib* - Especficar biblioteca a linkar

Exemplos

```
gcc ... -lefence ...
gcc ... -lfl -lwrap ...
```

Exemplos

Exemplos

```
gcc -E ...
gcc -S ...
gcc -Wall -Wextra ...
gcc -Wall -Wextra -std=c99 ...
gcc -Wall -Wextra -std=c99 -O0 -g ...
gcc -O2 -D_FORTIFY_SOURCE=2 ...
gcc -Wall -Wextra -Werror -O2 -Wp,-D_FORTIFY_SOURCE=2 ...
gcc -Wall -Wextra -g -lefence ...
gcc -Wall -Wextra -g -fmudflap -lmudflap ...
```

Conteúdo

splint

splint

Uma ferramenta para análise estática de programas C (lint).

Alguns dos problemas detectados pelo splint incluem (1/2):

- Dereferencing a possibly null pointer;
- Using possibly undefined storage or returning storage that is not properly defined;
- Type mismatches, with greater precision and flexibility than provided by C compilers;
- Violations of information hiding;

splint

Alguns dos problemas detectados pelo splint incluem (2/2):

- Memory management errors including uses of dangling references and memory leaks;
- Dangerous aliasing;
- Modifications and global variable uses that are inconsistent with specified interfaces;
- Problematic control flow such as likely infinite loops, fall through cases or incomplete switches, and suspicious statements;
- Buffer overflow vulnerabilities;
- Dangerous macro implementations or invocations;
- Violations of customized naming conventions.

Exemplos - avisos

$ splint warnings.c

```
Splint 3.1.1 --- 28 Jul 2005

warnings.c:  (in function main)
warnings.c:4:10:  Parameter argc not used
  A function parameter is not used in the body of the
  function.  If the argument is needed for type
  compatibility or future plans, use /*@unused@*/ in
  the argument declaration.  (Use -paramuse to inhibit
  warning)
warnings.c:4:22:  Parameter argv not used

Finished checking --- 2 code warnings
```

Exemplos - alguns métodos para desactivar avisos

Utilizar opções de linha de comando

```
$ splint -paramuse warnings.c
```

Alterar o código fonte: comentários especiais

```
#include <stdio.h>

int
main(/*@unused@*/ int argc, /*@unused@*/ char *argv[])
{
    printf("Sistemas Operativos.\n");

    return 0;
}
```

Modos de operação

Modos de operação (apêndice B)

- weak - Weak checking
- standard - The default mode (por omissão)
- checks - Moderately strict checking
- strict - Absurdly strict checking

Listar avisos activos por modo de operação

```
$ splint -help modes
```

Exemplo

```
$ splint +checks ...
```

Especificação de flags

Flags

- -flag - desactivar
- +flag - activar

Ordem de avaliação de flags

1. Ficheiro de configuração do utilizador (`$HOME/.splintrc`)
2. Ficheiro de configuração no directório de trabalho (`./.splintrc`)
3. Linha de comando (incluindo a especificação de ficheiros de configuração adicionais através da opção -f)

Ficheiro de configuração

Ficheiro de configuração: **.splintrc**

```
### Mode selection flags
#       weak, standard (default), checks, strict

+checks

### Display Flags

#+showscan
+showsummary
+stats

-hints
#+forcehints
```

Exemplo - problema associado à gestão de memória

exemplo.c - analisar o output produzido pelo splint

```
1  #include <stdlib.h>
2
3  static void f(void) {
4      int *p1 = NULL;
5
6      p1 = (int *) malloc(sizeof(int));
7      free(p1);
8      *p1 = 0;
9  }
10
11 int main(void) {
12     f();
13     return 0;
14 }
```

Exemplo - problema associado à gestão de memória (output)

$ splint exemplo.c

```
Splint 3.1.1 --- 21 Apr 2006

memory_2.c: (in function f)
memory_2.c:8:3: Variable p1 used after being released
   memory_2.c:7:7: Storage p1 released
memory_2.c:8:3: Dereference of possibly null pointer p1: *p1
   memory_2.c:6:7: Storage p1 may become null

Error Type                Reported  Suppressed
==========================  ========  ==========
nullderef                    1           0
sizeoftype                   0           1
usereleased                  1           0
                          ========  ==========
Total                        2           1

Finished checking --- 2 code warnings
16 source lines in 0.03 s
```

Referências

Projecto

- **Homepage**
  `http://www.splint.org/`
- **Documentação**
  `http://www.splint.org/documentation/`