

IA32: Controlo de Fluxo e Variáveis

Guião V

Apresentação

Este guião tem vista abordar os temas relacionados com o controlo de fluxo de instruções e a manipulação de variáveis usando o jogo de instruções de IA32.

Questão 1 (Comparações)

Nas alíneas seguintes, nas funções C substituíram-se alguns operadores de comparação e declarações de tipo por “__”. A partir do código produzido pelo GCC, determine os operadores em falta.

a)

<pre>char ctest(int a, int b) { char t1 = a __ b; return t1; }</pre>	<pre>movl 12(%ebp), %eax cmpl %eax, 8(%ebp) setl %al movsbl %al, %eax</pre>
--	---

b)

<pre>char ctest(__a, int b) { char t1 = a __ b; return t1; }</pre>	<pre>movl 12(%ebp), %eax cmpl %eax, 8(%ebp) setb %al movsbl %al, %eax</pre>
--	---

c)

<pre>char ctest(int a) { char t1 = a __ 0; return t1; }</pre>	<pre>movl 8(%ebp), %eax testl %eax, %eax setg %al movsbl %al, %eax</pre>
---	--

d)

<pre>char ctest(int a, int b) { char t1 = a __ b; char t2 = a __ b; char t3 = t1 + t2; return t3; }</pre>	<pre>movl 12(%ebp), %eax movl 8(%ebp), %ecx cmpl %eax, %ecx setne %dl cmpl %eax, %ecx setge %al addb %al, %dl movsbl %dl, %eax</pre>
---	--

Questão 2 (Controlo de fluxo)

Nos excertos de código desmontado, a seguir, alguns itens de informação foram substituídos por X's.

a) Qual o endereço destino especificado na instrução **jbe**?

```
8048d1c: 76 da                jbe XXXXXXXX  
8048d1e: eb 24                jmp 8048d44
```

b) Qual o endereço em que se encontra o início da instrução **mov**?

```
XXXXXXXX: eb 54                jmp 8048d44  
XXXXXXXX: c7 45 f8 10 00          mov $0x10,0xffffffff8(%ebp)
```

c) Nesta alínea, o endereço da instrução de salto é especificado no modo relativo ao IP/PC, em 4 bytes, codificado em complemento para 2. Qual o endereço especificado na instrução **jmp**?

```
8048902: e9 cb 00 00 00        jmp XXXXXXXX  
8048907: 90 nop
```

Questão 3 (vetores de tipo simples)

Use o comando `gcc -s` para criar o código de montagem referente ao programa, abaixo.

- Identifique e explique as instruções responsáveis pelo ciclo `for (...)`
- Identifique e explique as instruções responsáveis pelo cálculo do endereço de `vector[i]`

vectorInt.c	
<pre>#include <stdio.h> typedef int tipoInt; tipoInt vector[100]; int sum, i; void ini (void) { for (i=0 ; i<100 ; i++) vector[i] = i; }</pre>	<pre>main () { ini (); sum = 0; for (i=0 ; i<100 ; i++) sum += vector[i]; printf ("Sum=%d\n", sum); }</pre>

Questão 4 (vetores de estruturas)

Analise o código de montagem *main.S* criado com comando `gcc -s` para o programa, abaixo.

- Identifique e explique as instruções responsáveis pelo cálculo do endereço de `vector[i].a`. Compare com a resposta à questão 3, alínea b).
- Modifique no código em C o tamanho do campo **s** da estrutura para 8 caracteres. Identifique e explique as instruções responsáveis pelo cálculo do endereço de `vector[i].a`. Compare com a resposta à alínea anterior.

vectorEstrutura.c	
<pre>#include <stdio.h> typedef struct { char s[4]; int a; } tipoEstrutura; tipoEstrutura vector[100]; int sum, i; void ini (void) { for (i=1 ; i<100 ; i++) { vector[i].s[0] = '\0'; vector[i].a = i; } }</pre>	<pre>main () { ini (); sum = 0; for (i=0 ; i<100 ; i++) sum += vector[i].a; printf ("Sum=%d\n", sum); }</pre>