Architektury systemów komputerowych 2017

Lista zadań nr 5

Na zajęcia 29 i 30 marca 2017

Przy tłumaczeniu kodu w asemblerze `x86-64` do języka `C` należy trzymać się następujących wytycznych:

- Używaj złożonych wyrażeń minimalizując liczbę zmiennych tymczasowych.
- Nazwy wprowadzonych zmiennych muszą opisywać ich zastosowanie, np. `result` zamiast `rax`.
- Instrukcja `goto` jest zabroniona. Należy używać instrukcji sterowania `if`, `for`, `while` i `switch`.
- Jeśli to ma sens, pętle `while` należy przetłumaczyć do pętli `for`.

Zadanie 1. Zapisz w języku `C` funkcję o sygnaturze «`int puzzle(long x, unsigned n)`», której kod w asemblerze podano niżej. Przedstaw jednym zdaniem co ta procedura robi.

```
1  puzzle:
2          testl %esi, %esi
3          je    .L4
4          xorl  %edx, %edx
5          xorl  %eax, %eax
6  .L3:
7          movl  %edi, %ecx
8          andl  $1, %ecx
9          addl  %ecx, %eax
10         sarq  %rdi
11         incl  %edx
12         cmpl  %edx, %esi
13         jne   .L3
14         ret
15 .L4:
16         movl  %esi, %eax
17         ret
```

Zadanie 2. Wykonaj polecenia z poprzedniego zadania dla funkcji «`long puzzle2(char *s, char *d)`».

```
1  puzzle2:
2          movq  %rdi, %rax
3  .L3:
4          leaq  1(%rax), %r8
5          movb  -1(%r8), %r9b
6          movq  %rsi, %rdx
7  .L2:
8          incq  %rdx
9          movb  -1(%rdx), %cl
10         testb %cl, %cl
11         je    .L7
12         cmpb  %cl, %r9b
13         jne   .L2
14         movq  %r8, %rax
15         jmp   .L3
16 .L7:
17         subq  %rdi, %rax
18         ret
```

Zadanie 3. Przeczytaj poniższy kod w języku C i odpowiadający mu kod w asemblerze, a następnie wywnioskuj jakie są wartości stałych A i B.

```
typedef struct {
  int x[A][B];
  long y;
} str1;

typedef struct {
  char array[B];
  int t;
  short s[A];
  long u;
} str2;
```

```
void set_val(str1 *p, str2 *q) {
  long v1 = q->t;
  long v2 = q->u;
  p->y = v1 + v2;
}

set_val:
  movslq 8(%rsi),%rax
  addq   32(%rsi),%rax
  movq   %rax,184(%rdi)
  ret
```

Zadanie 4. Przeczytaj poniższy kod w języku C i odpowiadający mu kod w asemblerze, a następnie wywnioskuj jakie są wartości stałych R, S i T.

```
long A[R][S][T];

long store_elem(long i, long j,
                long k, long *dest)
{
  *dest = A[i][j][k];
  return sizeof(A);
}
```

```
store_elem:
  leaq (%rsi,%rsi,2),%rax
  leaq (%rsi,%rax,4),%rax
  movq %rdi,%rsi
  salq $6,%rsi
  addq %rsi,%rdi
  addq %rax,%rdi
  addq %rdi,%rdx
  movq A(,%rdx,8),%rax
  movq %rax,(%rcx)
  movq $3640,%rax
  ret
```

Zadanie 5. Przeczytaj poniższy kod w języku C i odpowiadający mu kod w asemblerze, a następnie wywnioskuj jaka jest wartość stałej CNT i jak wygląda definicja struktury a_struct.

```
typedef struct {
  int first;
  a_struct a[CNT];
  int last;
} b_struct;

void test (long i, b_struct *bp) {
  int n = bp->first + bp->last;
  a_struct *ap = &bp->a[i];
  ap->x[ap->idx] = n;
}
```

```
test:
  movl   0x120(%rsi),%ecx
  addl   (%rsi),%ecx
  leaq   (%rdi,%rdi,4),%rax
  leaq   (%rsi,%rax,8),%rax
  movq   0x8(%rax),%rdx
  movslq %ecx,%rcx
  movq   %rcx,0x10(%rax,%rdx,8)
  retq
```

Zadanie 6. Przeczytaj definicję unii elem oraz kod procedury proc i odpowiedz na poniższe pytania.

```
union elem {
  struct {
    long *p;
    long y;
  } e1;
  struct {
    long x;
    union elem *next;
  } e2;
};
```

```
proc:
  movq 8(%rdi),%rax
  movq (%rax),%rdx
  movq (%rdx),%rdx
  subq 8(%rax),%rdx
  movq %rdx,(%rdi)
  ret
```

- Jaki jest rozmiar unii elem w bajtach?
- Napisz funkcję w języku C, która odpowiada procedurze proc.

Zadanie 7. Poniższy kod w asemblerze otrzymano w wyniku deasemblacji funkcji zadeklarowanej jako «`long switch_prob(long x, long n)`». Zapisz w języku `C` kod odpowiadający tej funkcji.

```
400590 <switch_prob>:
400590: 48 83                   subq  $0x3c,%rsi
400594: 48 83 fe 05             cmpq  $0x5,%rsi
400598: 77 29                   ja    *0x4005c3 <switch_prob+0x33>
40059a: ff 24 f5 f8 06 40 00    jmpq  *0x4006f8(,%rsi,8)
4005a1: 48 8d 04 fd 00 00 00 00 lea   0x0(,%rdi,8),%rax
4005a9: c3                      retq
4005aa: 48 89 f8                movq  %rdi,%rax
4005ad: 48 c1 f8 03             sarq  $0x3,%rax
4005b1: c3                      retq
4005b2: 48 89 f8                movq  %rdi,%rax
4005b5: 48 c1 e0 04             shlq  $0x4,%rax
4005b9: 48 29 f8                subq  %rdi,%rax
4005bc: 48 89 c7                movq  %rax,%rdi
4005bf: 48 0f af ff             imulq %rdi,%rdi
4005c3: 48 8d 47 4b             leaq  0x4b(%rdi),%rax
4005c7: c3                      retq
```

Zrzut pamięci przechowującej tablicę skoków:

```
(gdb) x/6gx 0x4006f8
0x4006f8: 0x4005a1 0x4005a1
0x400708: 0x4005b2 0x4005c3
0x400718: 0x4005aa 0x4005bf
```

Zadanie 8. Przeczytaj definicje struktur `SA` i `SB`, a następnie przeanalizuj kod w asemblerze funkcji zadeklarowanych jako «`SB eval(SA s)`» i «`long wrap(long x, long y, long z)`».

```
typedef struct A {
  long u[2];
  long *v;
} SA;

typedef struct B {
  long p[2];
  long q;
} SB;
```

```
eval:
  movq  %rdi, %rax
  movq  16(%rsp), %rcx
  movq  24(%rsp), %rdx
  movq  (%rdx), %rsi
  movq  %rcx, %rdx
  imulq %rsi, %rdx
  movq  %rdx, (%rdi)
  movq  8(%rsp), %rdx
  movq  %rdx, %rdi
  subq  %rsi, %rdi
  movq  %rdi, 8(%rax)
  subq  %rcx, %rdx
  movq  %rdx, 16(%rax)
  ret
```

```
wrap:
  subq  $72, %rsp
  movq  %rdx, (%rsp)
  movq  %rsp, %rdx
  leaq  8(%rsp), %rax
  pushq %rdx
  pushq %rsi
  pushq %rdi
  movq  %rax, %rdi
  call  eval
  movq  40(%rsp), %rax
  addq  32(%rsp), %rax
  imulq 48(%rsp), %rax
  addq  $96, %rsp
  ret
```

Zapisz w języku `C` kod odpowiadający funkcjom `eval` i `wrap`. Narysuj diagram przedstawiający zawartość ramki stosu funkcji `wrap` w momencie wywołania funkcji `eval`.