Architektury systemów komputerowych 2016

Lista zadań nr 5

Na zajęcia 30 i 31 marca oraz 4 i 5 kwietnia 2016

Zadanie 1. Zapisz w języku `C` funkcję zadeklarowaną jako `long loop(long x, int n)`, której kod w asemblerze jest przedstawiony na poniższym listingu:

```
loop:
  movl  %esi,%ecx
  xorq  %rax,%rax
  leaq  1(%rax),%rdx
  jmp   .L2
.L3:
  movq  %rdi,%r8
  andq  %rdx,%r8
  orq   %r8,%rax
  salq  %cl,%rdx
.L2:
  testq %rdx,%rdx
  jne   .L3
  rep; retq
```

Zadanie 2. 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 3. 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 4. 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 5. Przeczytaj poniższą 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 6. 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
```

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

Zadanie 7. Przeczytaj definicje struktur `strA` i `strB`, a następnie przeanalizuj kod w asemblerze funkcji zadeklarowanych jako: `strB process(strA s)` i `long eval(long x, long y, long z)`.

```
typedef struct {
  long a[2];
  long *p;
} strA;

typedef struct {
  long u[2];
  long q;
} strB;
```

```
process:
  movq %rdi,%rax
  movq 24(%rsp),%rdx
  movq (%rdx),%rdx
  movq 16(%rsp),%rcx
  movq %rcx,(%rdi)
  movq 8(%rsp),%rcx
  movq %rcx,8(%rdi)
  movq %rdx,16(%rdi)
  retq
```

```
eval:
  subq $104,%rsp
  movq %rdx,24(%rsp)
  leaq 24(%rsp),%rax
  movq %rdi,(%rsp)
  movq %rsi,8(%rsp)
  movq %rax,16(%rsp)
  leaq 64(%rsp),%rdi
  call process
  movq 72(%rsp),%rax
  addq 64(%rsp),%rax
  addq 80(%rsp),%rax
  addq $104,%rsp
  retq
```

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