1) b tem 8 bytes, porque é um vetor de int com tamanho 2 e cada int tem 4 bytes.

a tem 12 bytes, porque é um array de 2 arrays de tamanho 3 e cada short tem 2 bytes.

2) o buraco ocorre porque os ints são alocados em endereços de memória múltiplos de 4. Ao declarar a struct X, a ocupou os primeiros 4 bytes da memoria, b ocupou os 2 seguintes e é preciso pular 2 campos para que c fique seja alocado em uma posição múltipla de 4.

3)

a. tam = 9 bytes, organização:

C2

i

i

i

i

C1

^múltiplo de 4

b. tam = 16 bytes, organização:

l

l

l

c

l

l

l

l

l

múltiplo de 8

c. tam = 8 bytes, organização:

C1

i

C2

i

i

i

múltiplo de 4 ^

d. tam = 24 bytes, organização:

l

l

l

c

l

l

l

l

l

c

múltiplo de 8

struct X2

e. tam = 3 bytes, organização:

C3

C1

C2

f. tam = 16 bytes, organização:

S1

S1

i

i

i

i

S2

S2

C[0]

C[1]

C[2]]

múltiplo de 4 ^ múltiplo de 2 ^

os dois paddings no final dessa struct são para garantir que ao alinhar ela com outra struct X6 o int i fique em um endereço múltiplo de 4.

g. tam = 8 bytes, organização:

C[2]

C[1]

C[0]

C[4]

C[3]

ou

i

i

i

i

múltiplo de 4

h. tam = 6 bytes, organização:

C[2]

C[1]

C[0]

C[4]

C[3]

ou múltiplo de 2

s

s