Piacente Cristian 866020

Homework H5 – Symbolic execution

Software Quality, Academic Year 2023-2024, University of Milan – Bicocca

Exercise 1) Given the following function

```
1   int foo (int a, int b) {
2     int k = 1;
3     int res = a;
4     while (k < b) {
5       res = res -10;
6       k = k + 5;
7     }
8     res = res + 10;
9     if (res > 0) {
10      System.out.println("Res is positive");
11    }
12    return res;
13  }
```

Compute the path condition which executes two iterations of the loop and prints the string "Res is positive". Provide a program input which satisfies the path condition.

To execute two iterations of the loop and print the string "Res is positive", we have to execute the following path: 1-2-3-4-5-6-4-5-6-4-8-9-10.

Let's compute the path condition with symbolic execution.

After 1: a=X, b=Y, PC=true
After 2: a=X, b=Y, k=1, PC=true
After 3: a=X, b=Y, k=1, res=X, PC=true
After 4 (jump to 5): a=X, b=Y, k=1, res=X, PC=1<Y
After 5: a=X, b=Y, k=1, res=X-10, PC=1<Y
After 6: a=X, b=Y, k=6, res=X-10, PC=1<Y
After 4 (jump to 5): a=X, b=Y, k=6, res=X-10, PC=1<Y && 6<Y
After 5: a=X, b=Y, k=6, res=X-20, PC=1<Y && 6<Y
After 6: a=X, b=Y, k=11, res=X-20, PC=1<Y && 6<Y
After 4 (jump to 8): a=X, b=Y, k=11, res=X-20, PC=1<Y && 6<Y && 11>=Y
After 8: a=X, b=Y, k=11, res=X-10, PC=1<Y && 6<Y && 11>=Y
After 9 (jump to 10): a=X, b=Y, k=11, res=X-10, PC=1<Y && 6<Y && 11>=Y && X-10>0

Path condition: 1 < Y && 6 < Y && 11 >= Y && X − 10 > 0.

We must consider the following inequalities:
Y > 1
Y > 6
Y <= 11
X > 10

That means 7 <= Y <= 11 and X > 10.

The path is **feasible**, since the path condition is satisfiable.

Since a=X and b=Y, an example of input is **a=11, b=11**.

Piacente Cristian 866020

Homework H5 – Symbolic execution

Software Quality, Academic Year 2023-2024, University of Milan – Bicocca

Exercise 2) Symbolically execute the following program, and determine whether the instruction "a[k] = 0;" (line 16) can or cannot generate a buffer overflow.

Motivate your answer and show the path conditions you generated during the analysis also stating for each of them whether it is satisfiable or not.

If a buffer overflow is possible, provide concrete inputs that trigger the overflow, by solving the execution condition computed with the symbolic execution.

```
1   int f(int x, int y) {
2     if (x < 0) {
3       x = abs( x );
4     } else if (y < 0) {
5       y = abs( y );
6     }
7
8     int[] a = new int[max(x,y) + 10];
9
10    int k = 0;
11    if (x > y) {
12      k = x − y;
13    } else {
14      k = y − x;
15    }
16    a[k] = 0;
17    return a[0];
18  }
```

To determine if line 16 can generate a buffer overflow, we have to compute every possible path conditions which reaches that line and, after making sure the path condition is satisfiable, check if the value of k can be greater or equal to the length of the array a.

Therefore, we can have an overflow iff **PC** **&&** **value of k** **>=** **length of a** is satisfiable.

There are 6 possible paths:

1) 1-2-4-8-10-11-12-16
2) 1-2-4-8-10-11-14-16
3) 1-2-3-8-10-11-12-16
4) 1-2-3-8-10-11-14-16
5) 1-2-4-5-8-10-11-12-16
6) 1-2-4-5-8-10-11-14-16

Let's perform symbolic execution on each one of them.

1)
After 1: x=A, y=B, PC=true
After 2 (jump to 4): x=A, y=B, PC=A>=0
After 4 (jump to 8): x=A, y=B, PC=A>=0 && B>=0
After 8: x=A, y=B, a=new int[max(A,B) + 10], PC=A>=0 && B>=0
After 10: x=A, y=B, a=new int[max(A,B) + 10], k=0, PC=A>=0 && B>=0
After 11 (jump to 12): x=A, y=B, a=new int[A + 10], k=0, PC=A>=0 && B>=0 && A>B we replace max(A,B) with A
After 12: x=A, y=B, a=new int[A + 10], k=A-B, PC=A>=0 && B>=0 && A>B

Consider A>=0 && B>=0 && A>B && A-B >= A + 10

We must consider the following inequalities:
A >= 0
B >= 0
A > B
-B >= 10 that is B <= -10

Piacente Cristian 866020

Homework H5 – Symbolic execution

Software Quality, Academic Year 2023-2024, University of Milan – Bicocca

There is a **contradiction**: B can't be both greater or equal to 0 and less or equal to -10.

If we only consider the path condition, then we have a **feasible path**, but **no possible overflow** because of the contradiction when considering PC && A-B >= A + 10.

2)
After 1: x=A, y=B, PC=true
After 2 (jump to 4): x=A, y=B, PC=A>=0
After 4 (jump to 8): x=A, y=B, PC=A>=0 && B>=0
After 8: x=A, y=B, a=new int[max(A,B) + 10], PC=A>=0 && B>=0
After 10: x=A, y=B, a=new int[max(A,B) + 10], k=0, PC=A>=0 && B>=0
After 11 (jump to 14): x=A, y=B, a=new int[B + 10], k=0, PC=A>=0 && B>=0 && A<=B
After 14: x=A, y=B, a=new int[B + 10], k=B-A, PC=A>=0 && B>=0 && A<=B

Consider A>=0 && B>=0 && A<=B && B-A >= B + 10

We must consider the following inequalities:
A >= 0
B >= 0
A <= B
-A >= 10 that is A <= -10

There is a **contradiction**: A can't be both greater or equal to 0 and less or equal to -10.

If we only consider the path condition, then we have a **feasible path**, but **no possible overflow** because of the contradiction when considering PC && B-A >= B + 10.

3)
After 1: x=A, y=B, PC=true
After 2 (jump to 3): x=A, y=B, PC=A<0
After 3: x=|A|, y=B, PC=A<0
After 8: x=|A|, y=B, a=new int[max(|A|,B) + 10], PC=A<0
After 10: x=|A|, y=B, a=new int[max(|A|,B) + 10], k=0, PC=A<0
After 11 (jump to 12): x=|A|, y=B, a=new int[|A| + 10], k=0, PC=A<0 && |A|>B
After 12: x=|A|, y=B, a=new int[|A| + 10], k=|A|-B, PC=A<0 && |A|>B

Consider A<0 && |A|>B && |A|-B >= |A| + 10

We must consider the following inequalities:
A < 0
|A| > B
-B >= 10 that is B <= -10

It is **satisfiable: there is a possible buffer overflow** and of course the path is **feasible**.

Since x=A and y=B, an example of input is **x=-1, y=-10**.

4)
After 1: x=A, y=B, PC=true
After 2 (jump to 3): x=A, y=B, PC=A<0
After 3: x=|A|, y=B, PC=A<0
After 8: x=|A|, y=B, a=new int[max(|A|,B) + 10], PC=A<0
After 10: x=|A|, y=B, a=new int[max(|A|,B) + 10], k=0, PC=A<0
After 11 (jump to 14): x=|A|, y=B, a=new int[B + 10], k=0, PC=A<0 && |A|<=B
After 14: x=|A|, y=B, a=new int[B + 10], k=B-|A|, PC=A<0 && |A|<=B

Consider A<0 && |A|<=B && B-|A| >= B + 10

Piacente Cristian 866020

Homework H5 – Symbolic execution

Software Quality, Academic Year 2023-2024, University of Milan – Bicocca

We must consider the following inequalities:
A < 0
|A| <= B
-|A| >= 10 that is |A| <= -10 **but this is impossible**.

If we only consider the path condition, then we have a **feasible path**, but **no possible overflow** because PC && B-|A| >= B + 10 is not satisfiable.

5)
After 1: x=A, y=B, PC=true
After 2 (jump to 4): x=A, y=B, PC=A>=0
After 4 (jump to 5): x=A, y=B, PC=A>=0 && B<0
After 5: x=A, y=|B|, PC=A>=0 && B<0
After 8: x=A, y=|B|, a=new int[max(A,|B|) + 10], PC=A>=0 && B<0
After 10: x=A, y=|B|, a=new int[max(A,|B|) + 10], k=0, PC=A>=0 && B<0
After 11 (jump to 12): x=A, y=|B|, a=new int[A + 10], k=0, PC=A>=0 && B<0 && A>|B|
After 12: x=A, y=|B|, a=new int[A + 10], k=A-|B|, PC=A>=0 && B<0 && A>|B|

Consider A>=0 && B<0 && A>|B| && A-|B| >= A + 10

We must consider the following inequalities:
A >= 0
B < 0
A > |B|
-|B| >= 10 that is |B| <= -10 **but this is impossible**.

If we only consider the path condition, then we have a **feasible path**, but **no possible overflow** because PC && A-|B| >= A + 10 is not satisfiable.

6)
After 1: x=A, y=B, PC=true
After 2 (jump to 4): x=A, y=B, PC=A>=0
After 4 (jump to 5): x=A, y=B, PC=A>=0 && B<0
After 5: x=A, y=|B|, PC=A>=0 && B<0
After 8: x=A, y=|B|, a=new int[max(A,|B|) + 10], PC=A>=0 && B<0
After 10: x=A, y=|B|, a=new int[max(A,|B|) + 10], k=0, PC=A>=0 && B<0
After 11 (jump to 14): x=A, y=|B|, a=new int[|B| + 10], k=0, PC=A>=0 && B<0 && A<=|B|
After 14: x=A, y=|B|, a=new int[|B| + 10], k=|B|-A, PC=A>=0 && B<0 && A<=|B|

Consider A>=0 && B<0 && A<=|B| && |B|-A >= |B| + 10

We must consider the following inequalities:
A >= 0
B < 0
A <= |B|
-A >= 10 that is A <= -10

There is a **contradiction**: A can't be both greater or equal to 0 and less or equal to -10.

If we only consider the path condition, then we have a **feasible path**, but **no possible overflow** because of the contradiction when considering PC && |B|-A >= |B| + 10.

Conclusion: **yes**, line 16 can generate a **buffer overflow** (see path **3)**).