**SE 421 Software Analysis & Verification for Safety and Security EXAM**

November 17, 2021

Name (last, first): Ogbondah, ChimzimScore out of 50:

PLEASE READ BEFORE YOU START

In compliance with the ISU Academic Misconduct Policy expectations, you will not give or receive aid on this examination.  This includes discussing or sharing the exam with anyone, uploading material online (e.g., share sources chegg.com and others) and using unauthorized sources while working on the EXAM.  If you are aware of cheating on this examination, you have an obligation to inform the instructor immediately.  By taking the exam you have agreed to comply with the ISU Academic Misconduct Policy expectations and abide by the following instructions. The instructor will act consistent with the ISU Academic Misconduct Policy.

**You must follow these instructions:**

1. You can download the question paper anytime between 9 am to 6 pm on the exam day. You must upload your answers within three hours thereafter. You will not be able to download the exam after 6 pm.
2. You can use only the lecture notes, the homework assignments, and the XINU code.
3. We have taken care to ensure clarity to avoid questions during the exam. However, we will try to respond in a timely manner if you have questions. We recommend taking the exam between 11 am to 2 pm if you are concerned about getting quick answers. Email any questions you may have during the exam with copies to both the TAs and the instructor. Include a US phone number or your Skype ID. You will get a response by mail or by phone call. DO NOT POST any questions on piazza.
4. Do your own work. Don’t copy material obtained from any person or any other source. You may be liable for academic misconduct even if it is not an exact copy.
5. Don’t collaborate/discuss with anyone else.
6. Don’t allow anyone else to copy any exam-related material.
7. Don’t take the exam for anyone else.
8. Do not collaborate with others on the exam.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Problem 1 | Problem 2 | Problem 3 | Problem 4 | Problem 5 | Problem 6 | Problem 7 | **Total** |
| 8 | 12 | 5 | 8 | 5 | 6 | 6 | 50 |
|  |  |  |  |  |  |  |  |

**Problem 1 (**8 Points**):**

**Part A** (2 points): We have discussed a *directed acyclic graph* (DAG) to model loop behaviors. Answer the following for transforming CFG to DAG.

1. The *loop back* edges are removed. TRUE or FALSE?
2. Edge is added from the *tail* node of the *loop back* edge to the *successor* of the *loop header which is in the loop body*. TRUE or FALSE?

Diagram

Description automatically generated**Part B** (3 points): Given the code and its CFG, show the DAG to model the loop.

A close up of text on a white background

Description automatically generated

**Part C** (3 Points): Which of the behaviors listed in the following Table can occur?

Notation:

*U(8): UNLOCK at line 8*

*U(16): UNLOCK at line 16*

*Only the behavior pertaining to LOCK and UNLOCK are shown*

*+: the contents of the parenthesis one or more times*

L2: L twice

|  |  |
| --- | --- |
| Behavior | Yes/No |
| (LU(8))+ U(16) | No |
| L U(16) | Yes |
| L2U(16) | No |
| U(16) | Yes |

**Problem 2 (**12 Points**): Use the given graph to answer the following questions.** The*true* edges are labeled by **T.** Two branch nodes have the same condition **C1.** The *true* edges for the first and the second C1 are respectively **e1** and **e11.** The *red* node is labeled R and the two *green* nodes are labeled G1 and G2.

Root

a3d ddd

C3s

RCd2

a4dd

a5dd

C1

T

e1

e2

R

T

e4

e3

X

e7

T

e5

e6

G1

e8

T

e9

C1

e10

T

e12

e11

G2

e14

e13

Leaf

1. How many times the forward pass flag Bf would be set to 1 at the second C1? **(2 points)**
   1. 5
2. Using the edge labels, enumerate all the paths from R to Leaf (e.g., e3e5e8e12e14) **(2 point)**
   1. **E3e5e10e13**
   2. **E3e5e8e11e13**
   3. **E3e5e8e12e14**
   4. **E3e6e9e11e13**
   5. **E3e6e9e12e14**
3. How many feasible paths are there from R to leaf? **(2 points)**
   1. **3**
4. Is there a *feasible* path from R to leaf on which there is no *green* node? **(2 point)**
   1. **No there is not a feasible path because if you take the false edge at X you hit a green node and then if you take the true edge you don’t hit one but since we know that C1 == True either way we go from e10 or e11 will take us to a green node**
5. Which node is the *immediate dominator* (idom) of G2**?** **(2 points)**
   1. **Node C1**
6. How many paths are there from Root to Leaf? **(1 point)**
   1. 12
7. Give the correct algebraic expression for all paths from Root to the second C1. (bonus point if you give the correct and also a minimal expression) Hint: [(e1e3+e2e4)e6 + e2e7] e9 + (e1e3+e2e4)e5e8 **(1 point + 1 bonus point)**
   1. [(e1e3+e2e4)e6+e2e7]e9 + [(e1e3+e2e4)e5]e8+e10(e11e13+e12e14)

**Problem 3 (5 points):** Fill the *Content* in following tables A to D to show the effect of executing each of the given statements A to D, each executed by itself (not following each other).

|  |  |
| --- | --- |
|  | Content |
| 0 | 4 |
| 1 | 4 |
| 2 | 0 |
| 3 | 11 |
| 4 | 22 |
| 5 |  |
| After A executes | |

|  |
| --- |
| P1 |
| P2 |
| Q |
| X |
| Y |
|  |
|  |
|  |

|  |  |
| --- | --- |
|  | Content |
| 0 | 3 |
| 1 | 4 |
| 2 | 0 |
| 3 | 22 |
| 4 | 22 |
| 5 |  |
| After B executes | |

|  |  |
| --- | --- |
|  | Content |
| 0 | 4 |
| 1 | 4 |
| 2 | 0 |
| 3 | 11 |
| 4 | 22 |
| 5 |  |
| After C executes | |

|  |  |
| --- | --- |
|  | Content |
| 0 | 5 |
| 1 | 4 |
| 2 | 0 |
| 3 | 11 |
| 4 | 22 |
| 5 | O1 |
| After D executes | |

|  |  |
| --- | --- |
| Address | Content |
| 0 | 3 |
| 1 | 4 |
| 2 | 0 |
| 3 | 11 |
| 4 | 22 |
| 5 |  |
| Initial Assignment | |

Table for A Table for B Table for C Table for D

Variable Declaration: int X, Y, \*P1, \*P2, \*\*Q;

The variables are located at addresses 0 to 4 as shown above.

The memory allocated by statement D is located at address 5.

**Initial Assignments:** **Statements A to D**:

P1 = &X; **A**. P1 = P2; **B.** \*P1 = Y; **C**. \*Q = P2; **D**. \*Q = malloc();

P2 = &Y;

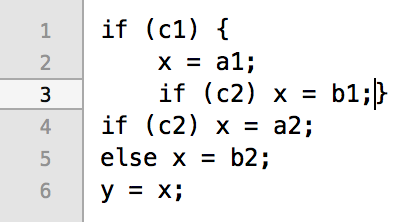
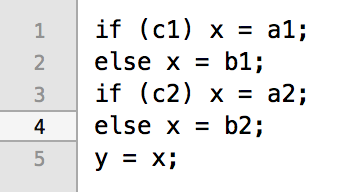
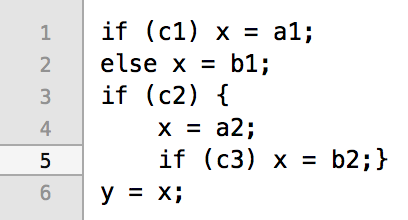
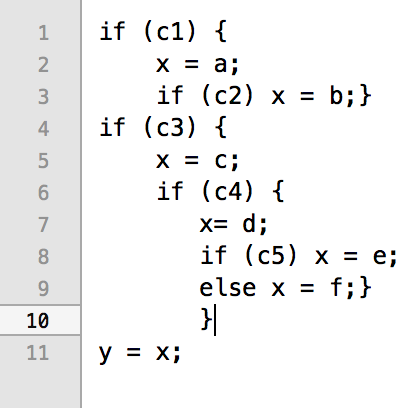
Q = &P1;

X = 11;

Y = 22;

**Problem 4: (8 points)** For each of the following code segments give: (a) the number of *definitions of* **x** that are live at the last *use of* **x**, (b) give the line numbers of these definitions

**Code 1 Code 2 Code 3 Code 4**



**Code 1:** # definitions of **x** are live at the last use of x: Line numbers of the live definitions:

* 2 : (3, x) (4, x)

**Code 2:** # definitions of **x** are live at the last use of x: Line numbers of the live definitions:

* 2 : (4, x) (5, x)

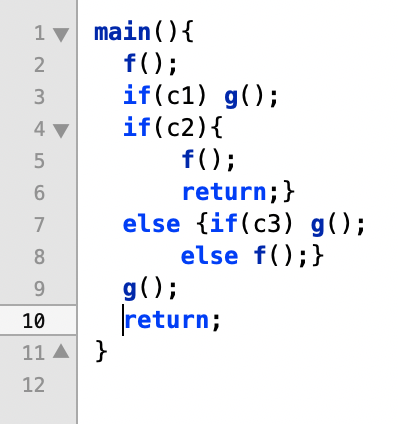
**Code 3:** # definitions of **x** are live at the last use of x: Line numbers of the live definitions:

* 4 : (1,x) (2,x) (4,x) (5,x)

**Code 4:** # definitions of **x** are live at the last use of x Line numbers of the live definitions:

* 5 : (2,x) (3,x) (5,x) (8,x) (9,x)

**Problem 5** (**5 points**): Give the number of **inter-procedural** **paths** in the function **main**. The control flow graphs for main, f and g are given.



Number of interprocedural paths: **29 paths**

CFG for main

A close up of a sign

Description automatically generatedA screenshot of a cell phone

Description automatically generatedDiagram

Description automatically generated

CFG for f

CFG for g

A close up of text on a white background

Description automatically generated

**Problem 6** (**6 Points**): Answer questions with respect to the following code. Assume that the function sum\_digits(x) returns sum of the digits of x. The inputs a1, a2, c1, and c2 are set before main executes. Assume a1and a2 to be non-zero positive integers.

A close up of text on a white background

Description automatically generated

**Part A** (2 points):

1. Give the line numbers for all definitions of the variable **d**.
   1. Def(4,d) def(10, d) def(15, d)
2. Give the line numbers for all uses of the variable **d**.
   1. (15, d) (10, d)

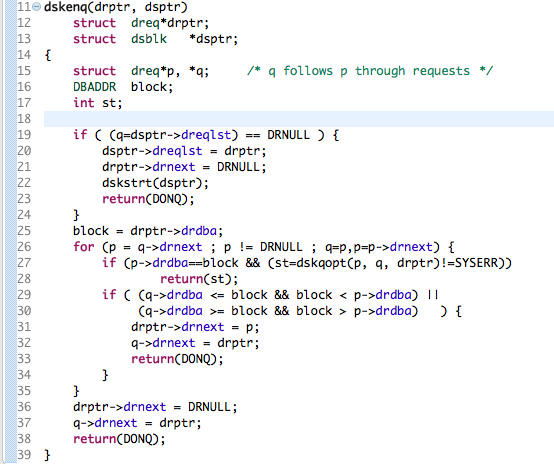
**Part B** (2): Going all the way forward from line 4, list all the **DU chains** for the variable **d**.

* {4 10 17} {4 15 17}

**Part C** (2 points): Going all the way backward from line 17, list only the **UD chains** that can cause the DBZ vulnerability.

* {17 15 4}

**Problem 7** (**6 Points**)**:** Refer to the given code for **dskenq**. The parameter **drptr** points to the memory allocated in a function that calls **dskenq**. A pointer to the allocated memory gets passed to other functions in several places in **dskenq**. This problem is about those passes. Assume that **dsptr** is a pointer to a global data structure.



**Part A** (2 points): Give the number of control paths in **dskenq** on which the pointer to the allocated memory (drptr) is assigned to a global variable?

* 2

**Part B** (2 points): Give the number of control paths in **dskenq** on which the pointer to the allocated memory is passed as a parameter to another function?

* 1

**Part C** (2 points) Does the function **dskstrt** get a pointer to the allocated memory?

* Yes, it does dsptr is a pointer to a global data structure and ->dreqlst is assigned drptr which is the pointer to allocated memory. Then inside of dskstrt() the it is accessing the pointers dreqlst value.