Assignment 3: The Highs and Lows of Information Flow
15-316 Software Foundations of Security and Privacy

Due: **11:59pm**, Wednesday 10/30/19
Total Points: 50

1. **Flow types (15 points).**

   Consider the following program.

   ```
   if(a = b) {
     c := 0
     d := d + 1
   } else {
     d := c × e
   }
   b := c
   ```

   Part 1 (5 points). Identify a *minimal* policy Γ under which this program type checks in the information flow type system described in lecture. The policy that you form must assign $\Gamma(a) = \mathtt{H}$, and be minimal in the sense that it assigns as few variables the label H as possible while still type checking.

   Part 2 (10 points). Use the rules of the information flow type system to show that the program typechecks under your policy.

2. **Exclusive interference (20 points).**

   Consider the following program under the policy $\Gamma = (a : \mathtt{H}, b : \mathtt{H}, c : \mathtt{L})$.

   ```
   if(a > 0) {
     if(b > 0) {
       c := 0;
     } else {
       c := 1;
     }
   } else {
     if(b > 0) {
       c := 1;
     } else {
       c := 0;
     }
   }
   ```

   Part 1 (5 points). Show that this program does not satisfy noninterference by providing a pair of inputs (a, b, c) and (a', b', c') that violate the formal definition given in lecture.

   Part 2 (10 points). Although this program does not satisfy noninterference, does it leak any information about the H variables a and b to an observer who sees the initial and final values of c? Describe the feasible set of initial values of a, b to justify your answer.

Part 3 (5 points). Building on the insights gained in the previous parts of this question, suppose that we propose a declassification rule for exclusive-or terms. (DeclassXor) below says that when $e, \widetilde{e}$ take Boolean (0, 1) values, then their exclusive-or can safely be leaked to the bottom security label.

$$\text{(DeclassXor)} \quad \frac{\Gamma \vdash e : \ell_1 \quad \Gamma \vdash \widetilde{e} : \ell_2 \quad e, \widetilde{e} \text{ evaluate to either } \{0,1\}}{\Gamma \vdash e \oplus \widetilde{e} : \bot}$$

Explain why this may be a justifiable rule to use in terms of the uncertainty remaining about $e, \widetilde{e}$. You may reference points that you have already made in Part 2 of this question.

3. **Leveraging interference (15 points).**

Consider the following program, under the type context $\Gamma = (a : \mathtt{H}, b : \mathtt{L}, c : \mathtt{L})$.

```
if(a < 0) {
  if(b < a) c := 0
  else c := 1
} else {
  if(a < b) c := 0
  else c := 1
}
```

Describe a procedure that leverages the fact that this program does not satisfy non-interference under Γ to learn the value of the H-typed variable. You can make use of the following assumptions.

- Assume that an attacker can control the values of L-typed variables prior to executing the program, and observe their value afterwards. They can neither control nor observe H variables at any point.
- $-N \leqslant a \leqslant N$ for some N whose value is unknown to the attacker.
- Finally, the attacker can run the program with different L inputs any number of times, and the H input will remain the same.

How many times does the attacker need to run the program using your procedure to learn a?