Formal Methods for Security and Privacy (SS2021) Prof. Matteo Maffei

Project 3 – minnieThor

TU Wien

Group 5

Our group consists of the following members:

Tobias Eidelpes
01527193

Exercise 3.1: Abstract Semantics

Nothing to fill in here, unless you feel that there is something about the implementation that we should consider when grading.

Exercise 3.2: Fixing reentrancy

Justification for the soundness of `alice.txt`:

By switching the order of line 5 and line 6 in `alice.txt`, the contract `bob.txt` has been fixed. This works because `sent` is set to 1 *before* the call instruction is executed and therefore reentering the function is possible but will not execute another call instruction because `sent` has already been set to 1. The contract now follows the *Check-Effects-Interactions Pattern*, which requires that checks and state changes have to occur before calls to other contracts.

Exercise 3.3: Soundness of the `CALL` rules

Soundness classification of `CALL` rule sets:

Call 1: Unsound
Call 2: Sound
Call 3: Unsound
Call 4: Unsound
Call 5: Sound

Intuition for the sound rule sets:

Call 2: The first two rules are the same rules as presented in the lecture. The third rule models the case where the contract resumes execution after a call with the same storage as before the call. This rule applies when the contract was not called and produced a successful halting state before the contract resumes the execution after the call.

The fourth rule, however, models the case where the contract resumes execution after it was left by another call from a contract that resulted in a halting state.

Call 5: As for Call 2, the first two rules are the same as presented in the lecture. The third rule is correct because in case of $sa[size - 4] = 1$, the local memory and the persistent storage are correctly over-approximated. The other case is where $sa[size - 4] = 0$. In this case the caller's memory may not be manipulated, as formalized by the small-step semantics.