| galois |

## Exercise 1:

### nfs.cry:

```
property prf uid = if (uid > 0) then ((request_privs uid) == 1) else False
```

### running nfs.cry in Cryptol:

```
Main> :s satNum=10
Main> :sat prf
Satisfiable
prf 4294901760 = True
prf 65536 = True
prf 196608 = True
prf 458752 = True
prf 983040 = True
prf 2031616 = True
prf 4128768 = True
prf 8323072 = True
prf 16711680 = True
prf 33488896 = True
Models found: 10
(Total Elapsed Time: 0.014s, using "Z3")
```

## Exercise 2:

### nfs.bc:

```
clang -g -O0 -emit-llvm -c nfs_1.c -o nfs.bc
```

### nfs.saw:

```
import "nfs_1.cry";  // correct version without %
let safe_setup = do {
   uid <- llvm_fresh_var "uid" (llvm_int 32);
   llvm_execute_func [ llvm_term uid ];
   llvm_return (llvm_term {{ become_user uid }});
};

let main : TopLevel () = do {
   m <- llvm_load_module "nfs.bc";
   saf_proof <- llvm_verify m "become_user" [] false safe_setup yices;
   print "Done!";
};
```

### running nfs.saw:

```
saw nfs.saw
[12:45:43.914] Verifying become_user ...
[12:45:43.914] Simulating become_user ...
[12:45:43.916] Checking proof obligations become_user ...
[12:45:43.923] Subgoal failed: become_user safety assertion:
internal: error: in _SAW_verify_prestate
Literal equality postcondition
Expected term: cryptol:/Main/become_user fresh:uid#788
```

```
Actual term:   let { x@1 = Prelude.Vec 32 Prelude.Bool
      x@2 = Prelude.bvNat 32 0
    }
 in Prelude.ite x@1 (Prelude.bvEq 32 x@2 fresh:uid#788) (Prelude.bvNat 32 3)
      (Prelude.ite x@1
         (Prelude.bvEq 32 x@2
            (Prelude.bvUExt 16 16
               (Prelude.slice Prelude.Bool 16 16 0 fresh:uid#788)))
         (Prelude.bvNat 32 1)
         (Prelude.bvNat 32 2))

[12:45:43.923] SolverStats {solverStatsSolvers = fromList ["SBV→Yices"],...
[12:45:43.924] ----------Counterexample----------
[12:45:43.924]   uid: 2147483648
[12:45:43.924] ----------------------------------
[12:45:43.924] Stack trace:
"llvm_verify" (.../nfs.saw:11:17-11:28):
Proof failed.
```

Observe that 2147483648 is a multiple of 65536 and, after compiling `nfs_1.c`, running `nfs_1 2147483648` yields 1, root is given permission.

**Exercise 3:**

**`type2_1.c`:**

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

uint64_t local_read(uint64_t fd, uint8_t *buf, uint64_t count) {
   return count;
}

uint64_t get_user_length(uint64_t x) {  return x;  }

uint64_t read_user_data(uint64_t fd) {
   char length;
   uint8_t buffer[32];
   uint64_t read;

   length = get_user_length(fd);
   if (length > 32) return 0;
   return local_read(fd, buffer, length);
}

int main (int argc, char** argv) {
   if (argc != 2) {
      printf("Usage: %s <number>\n", argv[0]);
      exit(0);
   }
   int f = atoi(argv[1]);
   printf("%lu\n",read_user_data(f));
}
```

**clang:**

```
[prompt]$ clang -g -O0 -emit-llvm -c type2_1.c -o type2.bc
```

**type2.cry:**

```
local_read : [64] -> [8] -> [64] -> [64]
local_read fd buf count = count

get_user_length : [64] -> [64]
get_user_length x = x

read_user_data : [64] -> [64]
read_user_data fd = if len > 32 then 0 else local_read fd buffer len
  where
    len = get_user_length fd
    buffer = 0
```

**type2.saw:**

```
import "type2.cry";

let safe_setup = do {
   fd <- llvm_fresh_var "fd" (llvm_int 64);
   llvm_execute_func [ llvm_term fd ];
   llvm_return (llvm_term {{ read_user_data fd }});
};

let main : TopLevel () = do {
   m <- llvm_load_module "type2.bc";
   saf_proof <- llvm_verify m "read_user_data" [] false safe_setup yices;
   print "Done!";
};
```

**running `type2.saw`:**

```
[[16:22:14.780] Verifying read_user_data ...
[16:22:14.780] Simulating read_user_data ...
[16:22:14.783] Checking proof obligations read_user_data ...
[16:22:14.790] Subgoal failed: read_user_data safety assertion:
internal: error: in _SAW_verify_prestate
Literal equality postcondition
Expected term: cryptol:/Main/read_user_data fresh:fd#787
Actual term:   let { x@1 = Prelude.Vec 64 Prelude.Bool
      x@2 = Prelude.slice Prelude.Bool 56 8 0 fresh:fd#787
    }
 in Prelude.ite x@1
      (Prelude.bvslt 32 (Prelude.bvNat 32 32) (Prelude.bvSExt 24 7 x@2))
      (Prelude.bvNat 64 0)
      (Prelude.bvSExt 56 7 x@2)

[16:22:14.790] SolverStats {solverStatsSolvers = fromList ["SBV→Yices"],...
[16:22:14.791] ----------Counterexample----------
[16:22:14.791]   fd: 128    ** says this input produces wrong output **
[16:22:14.791] ----------------------------------
[16:22:14.791] Stack trace:
"llvm_verify" (.../type2.saw:11:17-11:28):
Proof failed.
```

**Run `type2_1`:**

```
[prompt]$ type2_1 128
18446744073709551488
```

Hence the type2 code has a serious vulnerability.