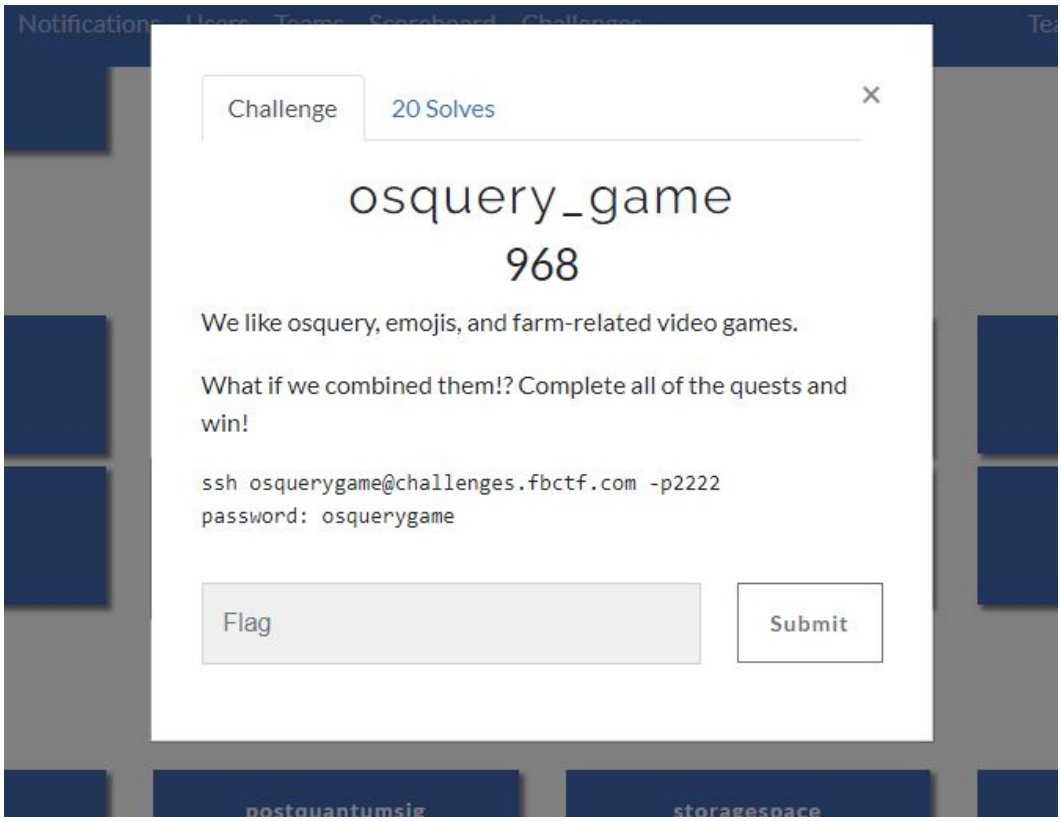
Facebook CTF 2019

OSQuery_Game (MISC)

Credit is shared between Grazfather and Chaocipher.



OK, so we SSH in and see what's going on. The banner message says type ".help" if needed.

```
chaocipher@VBUbuntu:~$ ssh -p 2222 osquerygame@challenges.fbctf.com
osquerygame@challenges.fbctf.com's password:
Using a virtual database. Need help, type '.help'
W0608 16:04:31.237695 24715 challenge.cpp:633] Welcome to the osquery farm simulator extension.
You have 5 days to make your farm successful.
osquery> .help
Welcome to the osquery shell. Please explore your OS!
You are connected to a transient 'in-memory' virtual database.

.all [TABLE]     Select all from a table
.bail ON|OFF     Stop after hitting an error
.echo ON|OFF     Turn command echo on or off
.exit            Exit this program
.features        List osquery's features and their statuses
.headers ON|OFF  Turn display of headers on or off
.help            Show this message
.mode MODE       Set output mode where MODE is one of:
                   csv      Comma-separated values
                   column   Left-aligned columns see .width
                   line     One value per line
                   list     Values delimited by .separator string
                   pretty   Pretty printed SQL results (default)
.nullvalue STR   Use STRING in place of NULL values
.print STR...    Print literal STRING
.quit            Exit this program
.schema [TABLE]  Show the CREATE statements
.separator STR   Change separator used by output mode
.socket          Show the osquery extensions socket path
.show            Show the current values for various settings
.summary         Alias for the show meta command
.tables [TABLE]  List names of tables
.types [SQL]     Show result of getQueryColumns for the given query
.width [NUM1]+   Set column widths for "column" mode
.timer ON|OFF      Turn the CPU timer measurement on or off
```

Let's see what the .table command does.

```
osquery> .tables
  => farm
  => farm_actions
  => farm_emoji
  => farm_quests
  => osquery_extensions
  => osquery_info
  => system_info
  => uptime
osquery>
```

Let's check the .schema on the tables. The system tables were of no interest so I left out those screenshots.

```
osquery> .schema farm
CREATE TABLE farm(`farm` TEXT, `action` TEXT, `src` INTEGER, `dst` INTEGER);
CREATE TABLE farm_actions(`action` TEXT, `description` TEXT);
CREATE TABLE farm_emoji(`emoji` TEXT, `meaning` TEXT);
CREATE TABLE farm_quests(`from` TEXT, `message` TEXT, `done` TEXT);
```

OSQuery only allows for Select queries so let see what we've got.

```
osquery> select * from farm_quests
    ...> ;
from|message|done
Town Mayor|The sheep wants to be next to the pig. Please move him, but be careful, if he sees yo
u he will run away in less than a second, you need to move fast.|no
Town Mayor|Please water something that you have planted. You need to pickup a pail first. The sh
eep was playing with the water pail, if you move him next to his friend he may give it back.|no
Town Mayor|Please pick something that you have grown. Wait a day after planting a seed and water
ing then pickup your plants.|no
Town Mayor|Weeds grow the first day of each season. Be careful, seeds and small plants will be o
vertaken.|yes
osquery>
```

This info tells us what the game is all about, what we need to accomplish, and in what order.

Select from the farm_emoji table.

```
osquery> select * from farm_emoji;
emoji|meaning
|weeds
|tractor
|plowed plot, plant seeds here
|pig
|water pail, pick it up, use it to water planted seeds
|sheep
|seedling that needs water
|a dead plant
|plant
|sunflower
osquery>
```

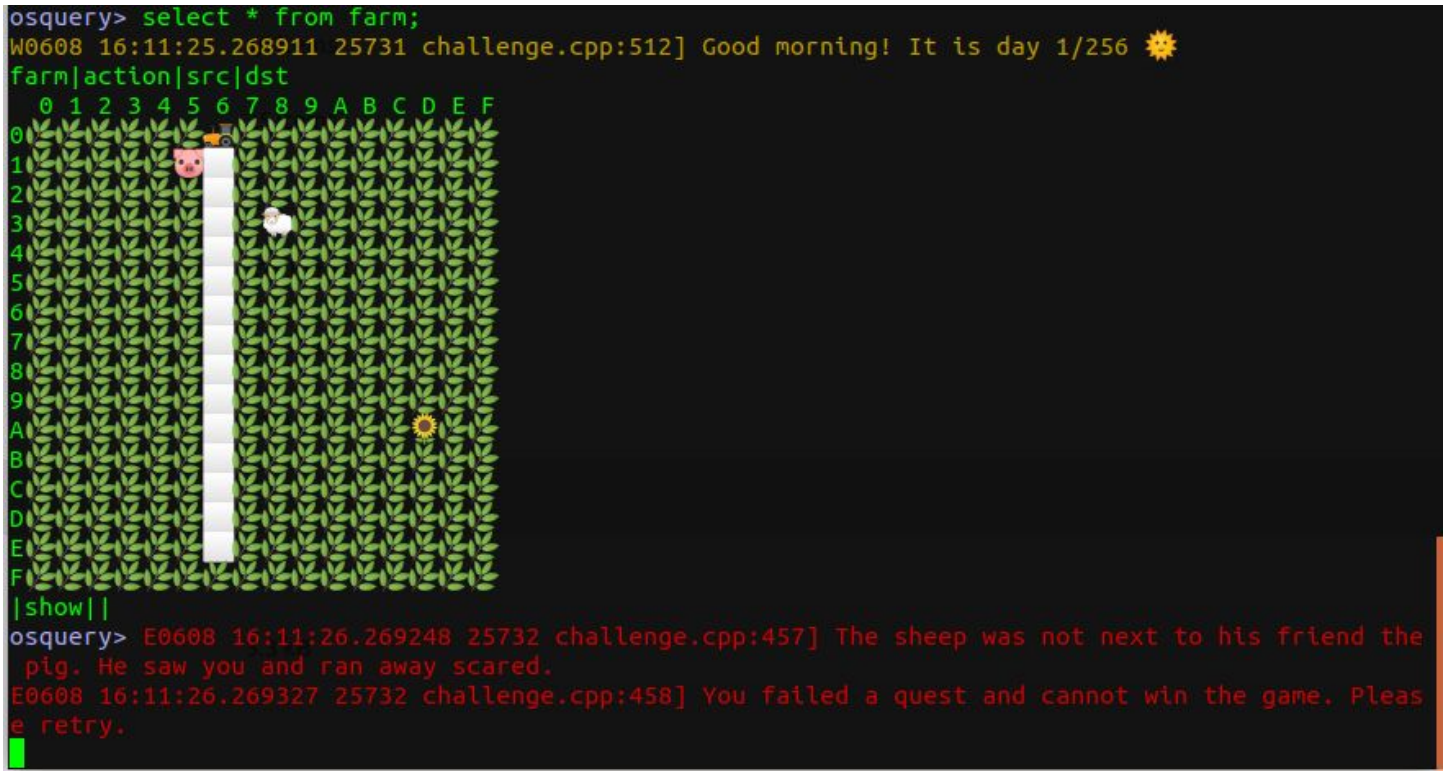
So, we've got unicode characters. That should be interesting to deal with. My teammate pulls up https://apps.timwhitlock.info/emoji/tables/unicode to find the UTF-8 encoding for each emoji.

Select from farm_actions table.

```
osquery> select * from farm_actions;
action|description
show|Default action, shows the farm.
move [src] [dst]|Requests to move animal in SRC field to DST field.
pickup [src]|Pickup item in SRC field.
water [...dst]|Water planted herb located at DST.
plant [...dst]|Plant a herb in the plowed DST.
osquery>
```

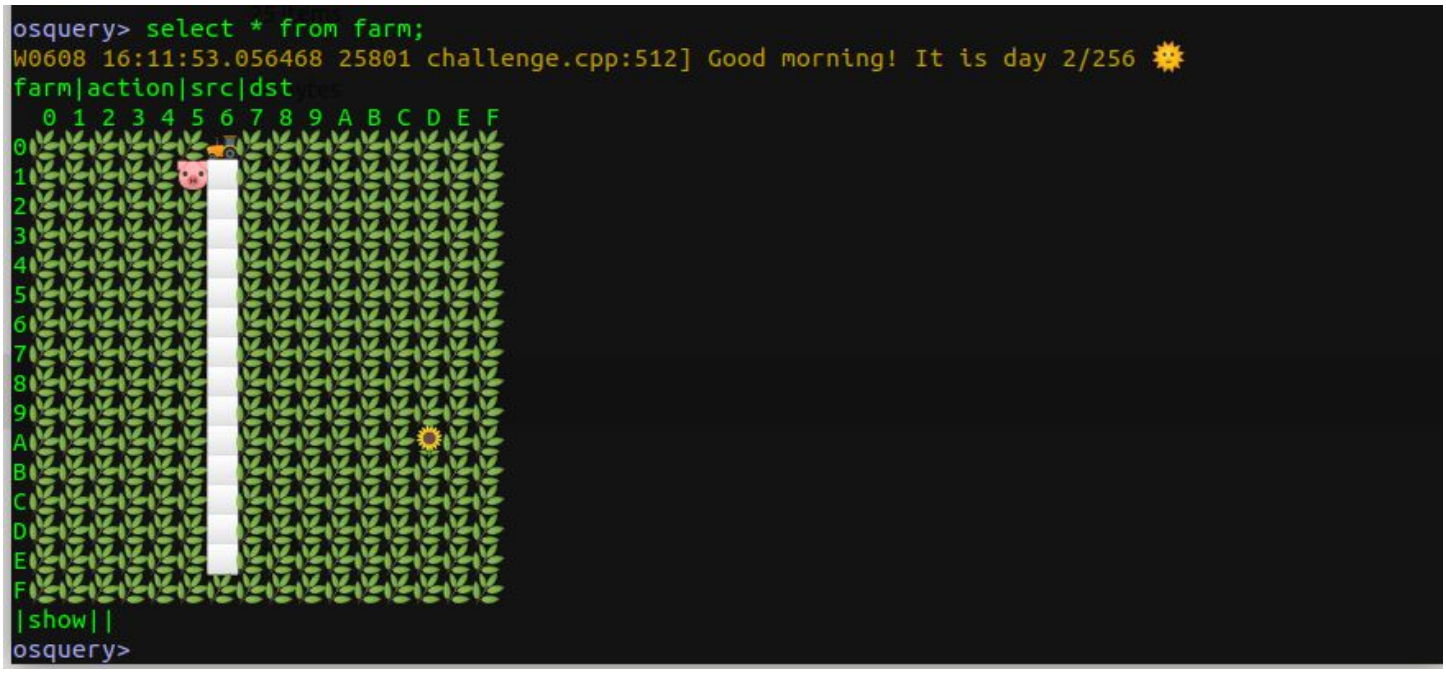
OK, now we understand the actions we can perform to fulfill the quests.

Let’s finally take a look at the farm.



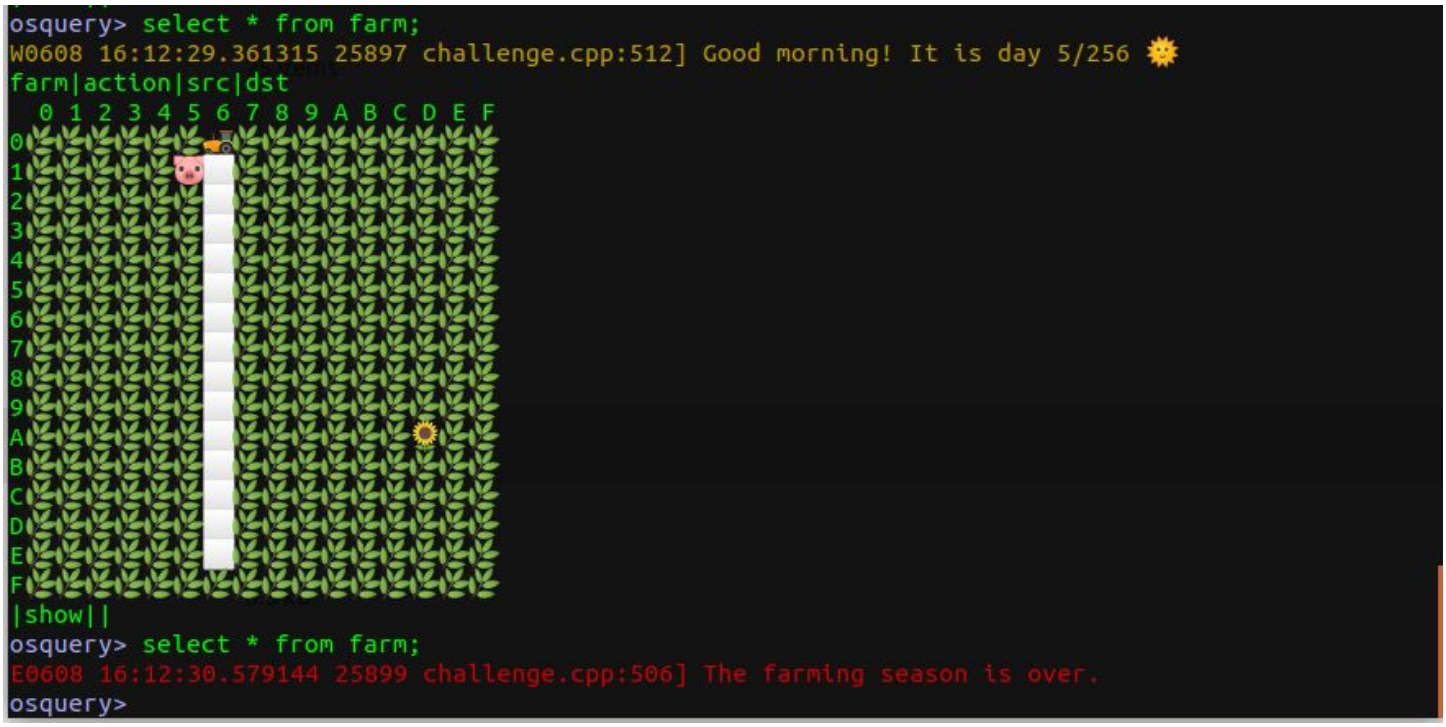
Notice the day 1 message at the top and the red text at the bottom. These provide the constraints of the game and hence the technical difficulty.

Next run of the select of the farm table.



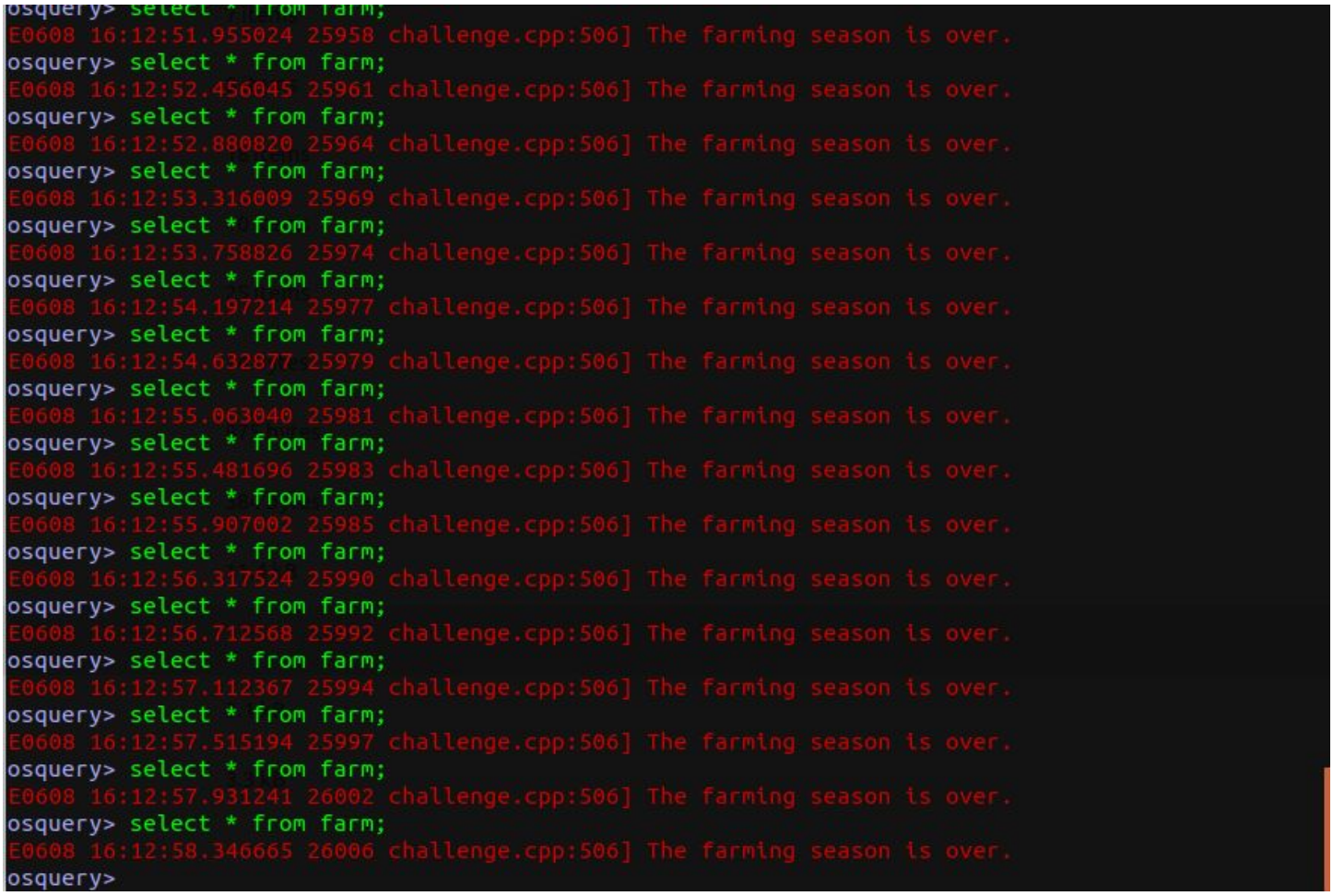
You can see the sheep is no longer there.

Select again a couple more times.



Now you can see we hit day 5 and the farming season is over message at the bottom.

OK. Let’s run the select a few more time and see what happens.



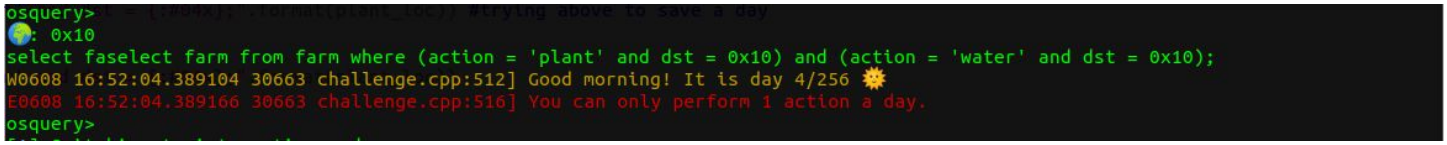
So, the problem statement is that we must:

Connect to the ssh server.

1. Figure out where the sheep is.
2. Move the sheep next to the pig within 1 second.
3. Pick up the bucket.
4. Plant a seed, then...
5. Water is with the bucket, then...
6. Pick the fruit.

So, for those keeping score at home. That’s 6 steps that need to be completed with only 5 moves and the movement of the sheep needs to happen within 1 second of viewing the farm. Steps 5, 6, and 7 need to be performed sequentially because the weeds will kill your seed before it becomes a fruit otherwise.

I know what you're thinking. Do an AND statement to perform two actions at once.



Yup, the organizers thought of that as you can see. OR statments and Joins were no luck either.

My teammate suggested earlier that we figure out how to get another season. There didn't seem to be any action that would give us that, but on a whim I decided to perform select statements manually 250ish more times and boom, new season. That was the solution, to move the sheep and grab the bucket in season 1 and roll the calendar over to the next year and perform the remaining actions.

Below is the code written by Grazfather, since mine was a little messy ;).

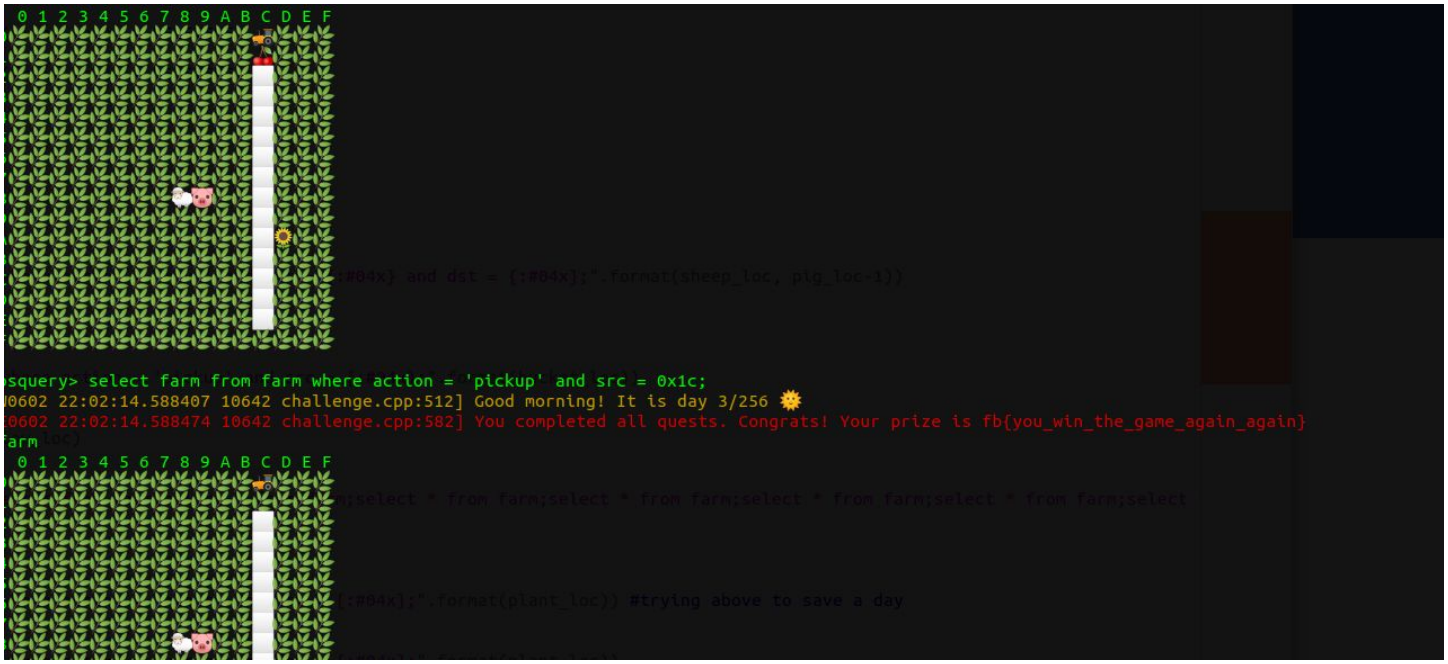
Code:

```
from pwn import *

sheep = "\xf0\x9f\x90\x91"
pig = "\xf0\x9f\x90\xb7"
weed = "\xf0\x9f\x8c\xbf"
plant = "\xf0\x9f\x8c\xb1"
fruit = "\xf0\x9f\x8d\x92"
tractor = "\xf0\x9f\x9a\x9c"
flower = "\xf0\x9f\x8c\xbb"
old_plow = "\xe2\xac\x9c"
plow = "\xf0\x9f\x8c\x8d"
bucket = "\xf0\x9f\x9a\xb0"

def prompt():
    return r.recvuntil("osquery> ")

def get_location(farm, square):
    farm = farm.replace(old_plow, plow) # We don't want 3 byte chars in there
    new_farm = []
    # print farm
    # print farm.splitlines()
    try:
        i = farm.splitlines().index('  0 1 2 3 4 5 6 7 8 9 A B C D E F ')
    except:
```

```
        print farm
        print farm.splitlines()
        return -1
    for line in farm.splitlines()[i+1:]:
        line = line[1:]
        new_farm.append(line)
    farm = "".join(new_farm)
    offset = farm.find(square)
    offset /= 4
    row = offset / 16
    column = offset % 16
    return offset

def print_all(farm):
    print farm
    loc = get_location(farm, weed)
    print "{}: {:#04x}".format(weed, (loc)) # DELETEME
    loc = get_location(farm, pig)
    print "{}: {:#04x}".format(pig, (loc)) # DELETEME
    loc = get_location(farm, sheep)
    print "{}: {:#04x}".format(sheep, (loc)) # DELETEME
    loc = get_location(farm, tractor)
    print "{}: {:#04x}".format(tractor, (loc)) # DELETEME
    loc = get_location(farm, flower)
    print "{}: {:#04x}".format(flower, (loc)) # DELETEME
    loc = get_location(farm, plow)
    print "{}: {:#04x}".format(plow, (loc)) # DELETEME

def move(src, dst):
    r.sendline("select farm from farm where action = 'move' and src = {:#04x} and dst = {:#04x};".format(src, dst))

def pickup(src):
    r.sendline("select farm from farm where action = 'pickup' and src = {:#04x};".format(src))


r = ssh("osquerygame", "challenges.fbctf.com", port=2222, password="osquerygame")
r = r.shell()
# This wastes a day
prompt()
r.sendline("select farm from farm where action = 'show';");
farm = prompt()
print farm
```

```
# Move sheep to pig
sheep_loc = get_location(farm, sheep)
pig_loc = get_location(farm, pig)
move(sheep_loc, pig_loc-1)
farm = prompt()
print farm

# Grab bucket
bucket_loc = get_location(farm, bucket)
pickup(bucket_loc)
farm = prompt()
print farm

for i in range(251):
    r.sendline("select farm from farm where action = 'show';");
    farm = prompt()
r.sendline("select farm from farm;");
farm = prompt()
print farm
r.sendline("select farm from farm;");
farm = prompt()
print farm

# Plant seed
plow_loc = get_location(farm, plow)
flower_loc = get_location(farm, flower)
r.sendline("select farm from farm where action = 'plant' and dst = {:#04x};".format(plow_loc))
farm = prompt()
print farm

# Water herb
r.sendline("select farm from farm where action = 'water' and dst = {:#04x};".format(plow_loc))
farm = prompt()
print farm

# Pick fruit
r.sendline("select * from farm where action = 'pickup' and src = {:#04x}".format(plow_loc))
r.interactive()
```

Flag:


squery> select farm from farm where action = 'pickup' and src = 0x1c;
0602 22:02:14.588407 10642 challenge.cpp:512] Good morning! It is day 3/256
0602 22:02:14.588474 10642 challenge.cpp:582] You completed all quests. Congrats! Your prize is fb{you_win_the_game_again_again}