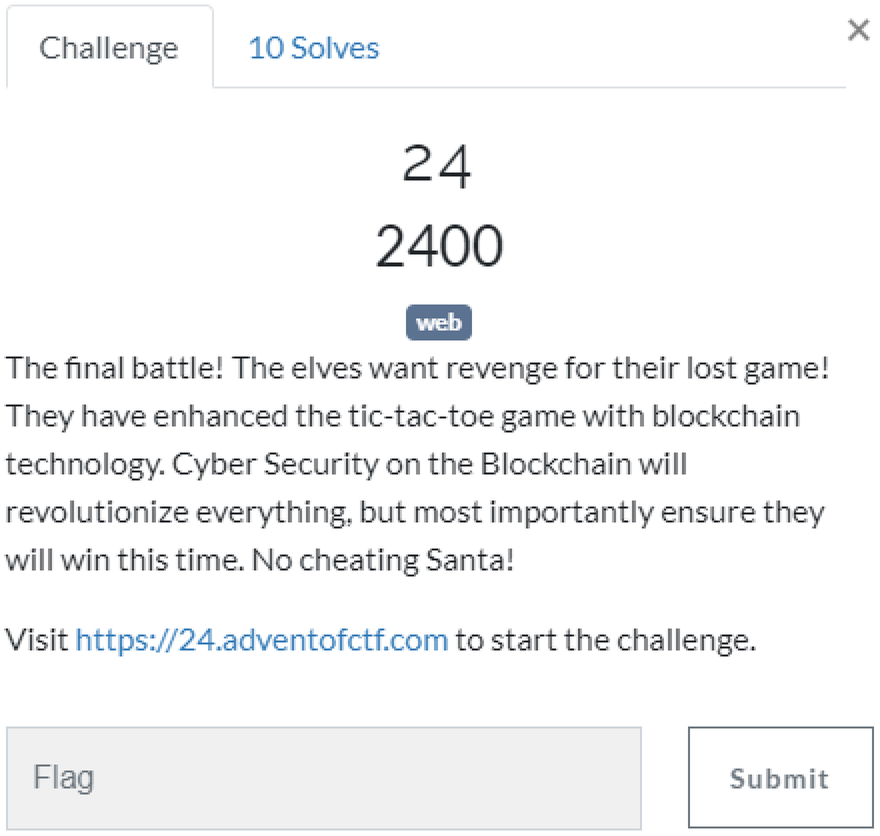
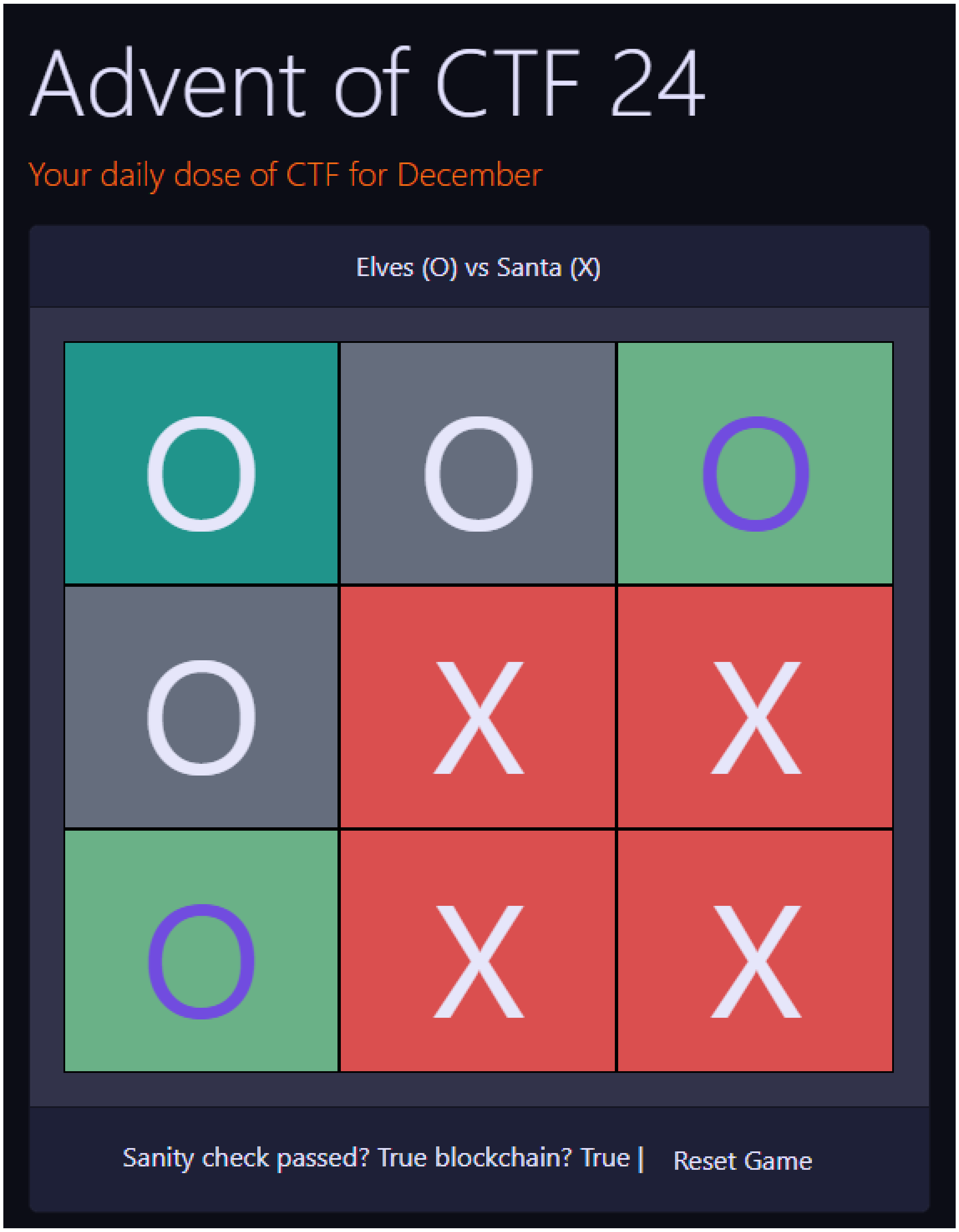
AdventOfCTF-24



The final battle! The elves want revenge for their lost game! They have enhanced the tic-tac-toe game with blockchain technology. Cyber Security on the Blockchain will revolutionize everything, but most importantly ensure they will win this time. No cheating Santa!

Visit https://24.adventofctf.com to start the challenge.

This was a similar challenge to day 20.


Advent of CTF 24

Your daily dose of CTF for December

Elves (O) vs Santa (X)

O	O	O
O	X	X
O	X	X

Sanity check passed? True blockchain? True | Reset Game

In the cookies tab there is a cookie that holds the game information.

```
gAN9cQAoWAUAAABib2FyZHEBXXECKF1xAyhYAQAAAE9xBGgETmVdcQUoaARYAQAAAFhxBmgGZV1xByhOaAZoB
mVlWAQAAAB0dXJucQhoBFgIAAAAZmluaXNoZWRxCYlYBgAAAHdpbm5lcnEKWAAAAABxC1gEAAAAc2FuZXEMiF
gKAAAAYmxvY2tjaGFpbnENiFgFAAAAY2hhaW5xDl1xDyh9cRAoaAFdcREoXXESKE5OTmVdcRMoTk5OZV1xFCh
```

```
OTmgGZWVYBAAAAHByZXZxFVggAAAAY2VmMjE1YzViZThjZjYzZmNmM2Q0M2VjZjI1MTBiMzNxFlgEAAAAaGFz
aHEXWCAAAABlN2RjOGUxZjdhNjc4OGJjMGNiNjg0MTUzOGIyMTZlOHEYdX1xGShoAV1xGihdcRsoaAROTmVdc
RwoTk5OZV1xHShOTmgGZWVoFWgYaBdYIAAAAGZjOTMyMzZiNWVlYTVmMWQ1NWUyYjViMzA4ZDY3MzkwcR51fX
EfKGgBXXEgKF1xIShoBE5OZV1xIihOaAZOZV1xIyhOTmgGZWVoFWgeaBdYIAAAAGE4ZGMwZDNkYTI5MGQxZTg
5NGVhYWZmY2I5ODM5OGM5cSR1fXElKGgBXXEmKF1xJyhoBGgETmVdcSgoTmgGTmVdcSkoTk5oBmVlaBVoJGgX
WCAAAABlNzRmNWIyMmY1MjEzYmE0YzI0NDk3NTljZTkxYzJhYXEqdX1xKyhoAV1xLChdcS0oaARoBE5lXXEuK
E5oBmgGZV1xLyhOTmgGZWVoFWgqaBdYIAAAAGVmMjU1MTRkZmZiZjgyNDdjZmY2MDYzYmU5MGYyZDU0cTB1fX
ExKGgBXXEyKF1xMyhoBGgETmVdcTQoaARoBmgGZV1xNShOTmgGZWVoFWgwaBdYIAAAAGUzYTRjMDM3YmRmMTU
0YjM0NGVkOWJkMTY0M2E2MjlkcTZ1fXE3KGgBXXE4KF1xOShoBGgETmVdcTooaARoBmgGZV1xOyhOaAZoBmVl
aBVoNmgXWCAAAABjMjQwZmExNjE3MzdjOTY3ZWNlNWZkOTQ2NzJhYjBmOHE8dWV1Lg==
```

There is now in place a blockchain technology that prevents hacker from manipulating the game state. To decode the game information you can use the following code:

```
import base64
import pickle
data="gAN9c....dWV1Lg=="
game=base64.b64decode(data)
print(pickle.loads(game))
```

```
{'board': [['O', 'O', None], ['O', 'X', 'X'], [None, 'X', 'X']], 'turn': 'O', 'finish
ed': False, 'winner': '', 'sane': True, 'blockchain': True, 'chain': [{'board': [[Non
e, None, None], [None, None, None], [None, None, 'X']], 'prev': 'cef215c5be8cf63fcf3d
43ecf2510b33', 'hash': 'e7dc8e1f7a6788bc0cb6841538b216e8'}, {'board': [['O', None, No
ne], [None, None, None], [None, None, 'X']], 'prev': 'e7dc8e1f7a6788bc0cb6841538b216e
8', 'hash': 'fc93236b5eea5f1d55e2b5b308d67390'}, {'board': [['O', None, None], [None,
'X', None], [None, None, 'X']], 'prev': 'fc93236b5eea5f1d55e2b5b308d67390', 'hash':
 'a8dc0d3da290d1e894eaaffcb98398c9'}, {'board': [['O', 'O', None], [None, 'X', None],
[None, None,
'X']], 'prev': 'a8dc0d3da290d1e894eaaffcb98398c9', 'hash': 'e74f5b22f5213ba4c2449759c
e91c2aa'}, {'board': [['O', 'O', None], [None, 'X', 'X'], [None, None, 'X']], 'prev':
'e74f5b22f5213ba4c2449759ce91c2aa', 'hash': 'ef25514dffbf8247cff6063be90f2d54'}, {'bo
ard': [['O', 'O', None], ['O', 'X', 'X'], [None, None, 'X']], 'prev': 'ef25514dffbf82
47cff6063be90f2d54', 'hash': 'e3a4c037bdf154b344ed9bd1643a629d'}, {'board': [['O',
 'O', None], ['O', 'X', 'X'], [None, 'X', 'X']], 'prev': 'e3a4c037bdf154b344ed9bd1643
a629d', 'hash': 'c240fa161737c967ece5fd94672ab0f8'}]}
```

In the page source code we see also this hint:

```
<!-- Development notes: Do not let santa see!

def hash_string(string):
    return hashlib.md5(string.encode('utf-8')).hexdigest()

def hash_row(row):
    conv = lambda i : i or ' '
    res = [conv(i) for i in row]
    return hash_string(' '.join(res))

def hash_board(board):
    acc = ""
    for row in board:
        acc += hash_row(row)
    return acc

def verify_chain(game):
    board=game["board"]
    chain = game["chain"]

    if len(chain) > 0:
        if board != chain[-1]["board"]:
            return False

    for i in range(len(chain)):
        block=chain[i]
        h = hash_board(block["board"])
        h = hash_string(h + block["prev"])
        if h != block["hash"]:
            return False
    return True

-->
```

This is the code that checks if we have a valid board state. By changing the original game state to one where we can make the X player win we will complete the challenge.

```
{'board': [['O', 'O', None], [None, 'X', 'X'], [None, None, 'X']], 'turn': 'O', 'fini
shed': False, 'winner': '', 'sane': True, 'blockchain': True, 'chain': [{'board': [[N
one, None, None], [None, None, None], [None, None, 'X']], 'prev': 'cef215c5be8cf63fcf
3d43ecf2510b33', 'hash': 'e7dc8e1f7a6788bc0cb6841538b216e8'}, {'board': [['O', None,
 None], [None, None, None], [None, None, 'X']], 'prev': 'e7dc8e1f7a6788bc0cb6841538b2
16e8', 'hash': 'fc93236b5eea5f1d55e2b5b308d67390'}, {'board': [['O', None, None], [No
ne, 'X', None], [None, None, 'X']], 'prev': 'fc93236b5eea5f1d55e2b5b308d67390', 'has
h': 'a8dc0d3da290d1e894eaaffcb98398c9'}, {'board': [['O', 'O', None], [None, 'X', Non
e], [None, None, 'X']], 'prev': 'a8dc0d3da290d1e894eaaffcb98398c9', 'hash': 'e74f5b22
f5213ba4c2449759ce91c2aa'}, {'board': [['O', 'O', None], [None, 'X', 'X'], [None, Non
e, 'X']], 'prev': 'e74f5b22f5213ba4c2449759ce91c2aa', 'hash': 'ef25514dffbf8247cff606
3be90f2d54'}]}
```

So i chose this game state. Using the code provided in the challenge we can make sure it is valid for the blockchain check.

```
import hashlib
def hash_string(string):
    return hashlib.md5(string.encode('utf-8')).hexdigest()

def hash_row(row):
    conv = lambda i : i or ' '
    res = [conv(i) for i in row]
    return hash_string(' '.join(res))

def hash_board(board):
    acc = ""
    for row in board:
        acc += hash_row(row)
    return acc

def verify_chain(game):
    board=game["board"]
    chain = game["chain"]

    if len(chain) > 0:
        if board != chain[-1]["board"]:
            return False

    for i in range(len(chain)):
        block=chain[i]
        h = hash_board(block["board"])
        h = hash_string(h + block["prev"])
        if h != block["hash"]:
            return False
    return True
if __name__ == '__main__':
    game = {'board': .... 'hash': 'ef25514dffbf8247cff6063be90f2d54'}]}
    print(verify_chain(game))
```

```
True
```

Nice, now we need to convert back this game state to a pickle and base64 encode it. We will use the code from the day 20 challenge.

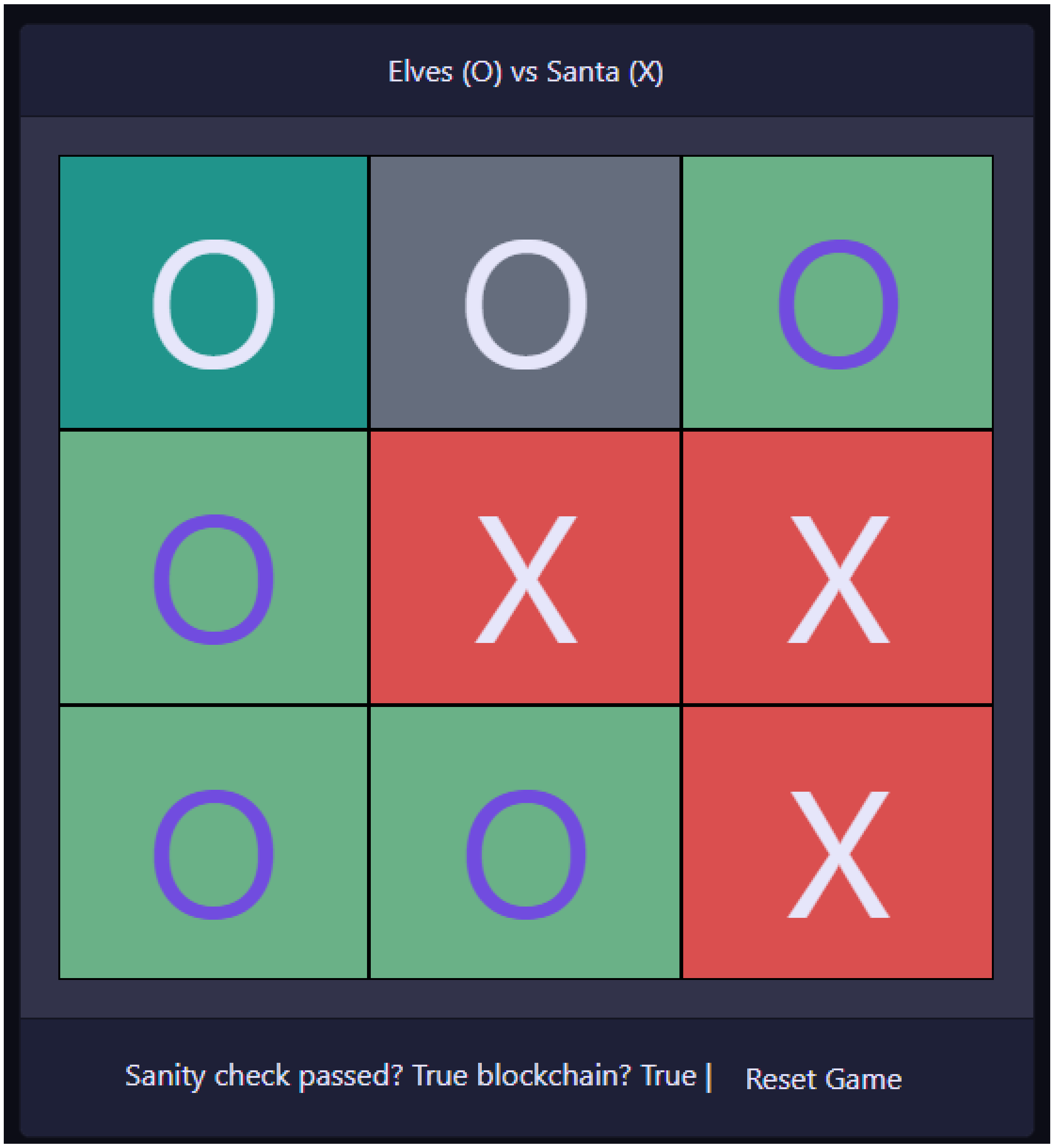
```python
import base64
import pickle

game = {'board': .... 'hash': 'ef25514dffbf8247cff6063be90f2d54'}]}

game = base64.b64encode(pickle.dumps(game))
print(game)
```

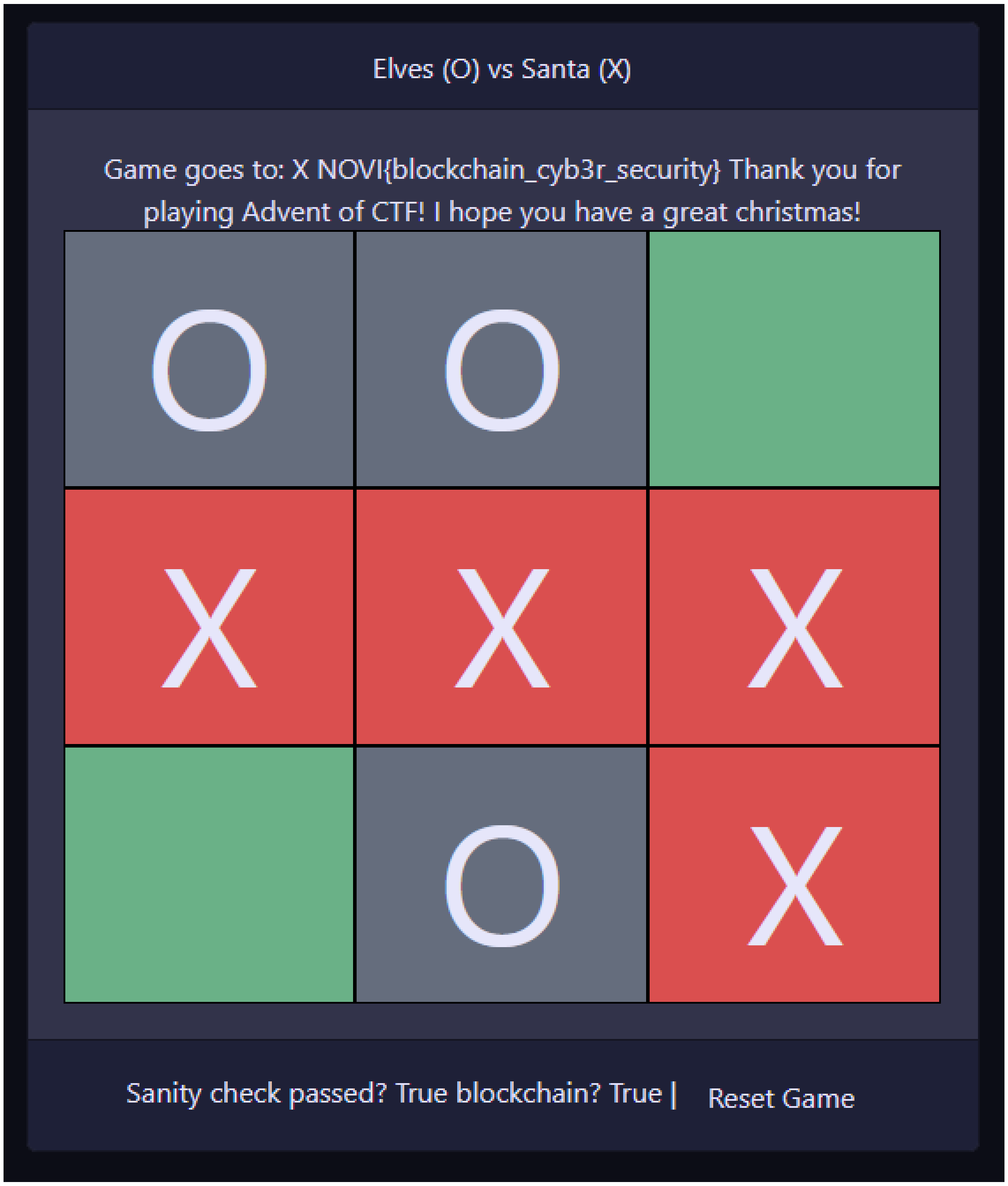
```
b'gASVFAIAAAAAAAB9lCiMBWJvYXJklF2UKF2UKIwBT5RoBE5lXZQoTowBWJRoBmVdlChOTmgGZWWMBHR1cm6
UaASMCGZpbmlzaGVklImMBndpbm5lcpSMAJSMBHNhbmWUiIwKYmxvY2tjaGFpbpSIjAVjaGFpbpRdlCh9lCho
AV2UKF2UKE5OTmVdlChOTk5lXZQoTk5oBmVljARwcmV2lIwgY2VmMjE1YzViZThjZjYzZmNmM2Q0M2VjZjI1M
TBiMzOUjARoYXNolIwgZTdkYzhlMWY3YTY3ODhiYzBjYjY4NDE1MzhiMjE2ZTiUdX2UKGgBXZQoXZQoaAROTm
VdlChOTk5lXZQoTk5oBmVlaBVoGGgXjCBmYzkzMjM2YjVlZWE1ZjFkNTVlMmI1YjMwOGQ2NzM5MJR1fZQoaAF
dlChdlChoBE5OZV2UKE5oBk5lXZQoTk5oBmVlaBVoHmgXjCBhOGRjMGQzZGEyOTBkMWU4OTRlYWFmZmNiOTgz
OThjOZR1fZQoaAFdlChdlChoBGgETmVdlChOaAZOZV2UKE5OaAZlZWgVaCRoF4wgZTc0ZjViMjJmNTIxM2JhN
GMyNDQ5NzU5Y2U5MWMyYWGUdX2UKGgBXZQoXZQoaARoBE5lXZQoTmgGaAZlXZQoTk5oBmVlaBVoKmgXjCBlZj
I1NTE0ZGZmYmY4MjQ3Y2ZmNjA2M2JlOTBmMmQ1NJR1ZXUu'
```

Now edit the cookie in the page.



Sanity check passed? True blockchain? True |

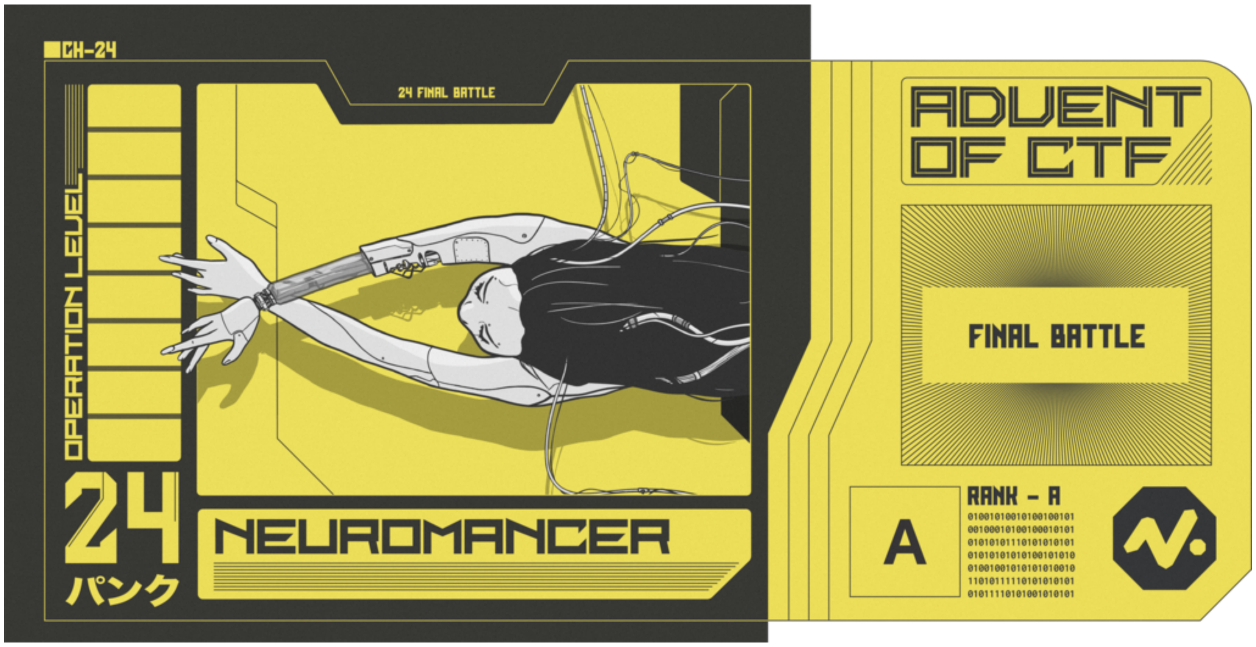
This is the new board state and both checks are True. Since we control both players we can win easily by clicking on (2,1) and then on (1,0) and we will reach this board state where player X wins.


Elves (O) vs Santa (X)

Game goes to: X NOVI{blockchain_cyb3r_security} Thank you for playing Advent of CTF! I hope you have a great christmas!

Sanity check passed? True blockchain? True |

Flag: NOVI{blockchain_cyb3r_security}


CH-24
24 FINAL BATTLE
OPERATION LEVEL
24
パンク
NEUROMANCER
ADVENT OF CTF
FINAL BATTLE
RANK - A
A