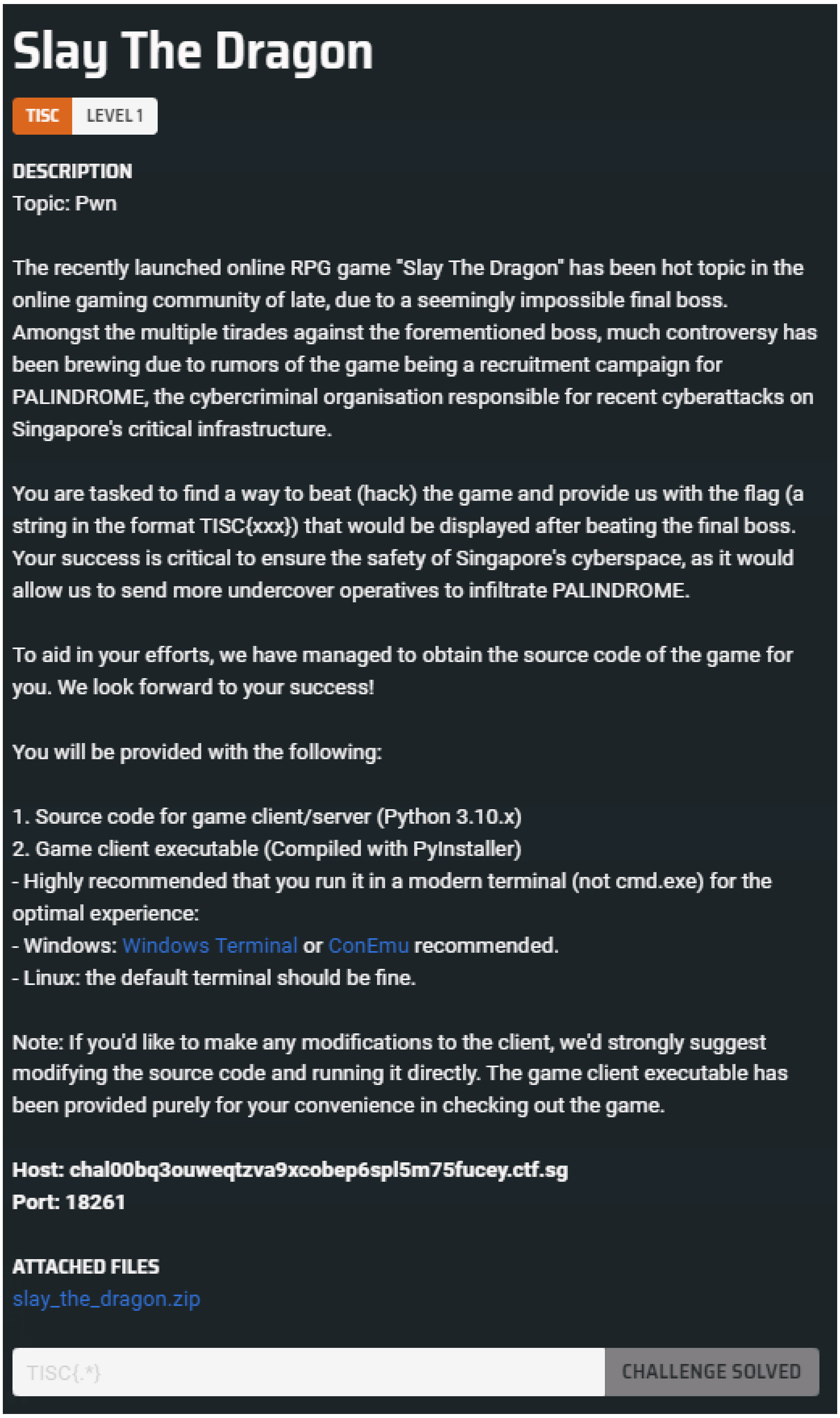


Slay The Dragon

DESCRIPTION

Topic: Pwn

The recently launched online RPG game "Slay The Dragon" has been hot topic in the online gaming community of late, due to a seemingly impossible final boss. Amongst the multiple tirades against the forementioned boss, much controversy has been brewing due to rumors of the game being a recruitment campaign for PALINDROME, the cybercriminal organisation responsible for recent cyberattacks on Singapore's critical infrastructure.

You are tasked to find a way to beat (hack) the game and provide us with the flag (a string in the format TISC{xxx}) that would be displayed after beating the final boss. Your success is critical to ensure the safety of Singapore's cyberspace, as it would allow us to send more undercover operatives to infiltrate PALINDROME.

To aid in your efforts, we have managed to obtain the source code of the game for you. We look forward to your success!

You will be provided with the following:

1. Source code for game client/server (Python 3.10.x)
2. Game client executable (Compiled with PyInstaller)
- Highly recommended that you run it in a modern terminal (not cmd.exe) for the optimal experience:
- Windows: Windows Terminal or ConEmu recommended.
- Linux: the default terminal should be fine.

Note: If you'd like to make any modifications to the client, we'd strongly suggest modifying the source code and running it directly. The game client executable has been provided purely for your convenience in checking out the game.

Host: chal00bq3ouweqtzva9xcobep6spl5m75fucey.ctf.sg
Port: 18261

ATTACHED FILES

slay_the_dragon.zip

Install python 3.10.x in anaconda environment

Install poetry as per PLEASE_READ.txt

Install a modern terminal

Activate environment in the terminal using “activate dragon”, where dragon is name of environment

- python main.py --host chal00bq3ouweqtzva9xcobep6spl5m75fucey.ctf.sg --port 18261

Can change client side things like client>event>workevent and set the CREEPER_ENCOUNTER_CHANCE = 0

So we can mine for gold without chance of dying, that way we can buy more potions and this allows us to defeat the 2nd creature

But dragon boss has 100hp and 50dmg, when we only have 10hp so we will die 1 shot, even if we have a lot of potions

Try to change core>models>player

```
def attack(self) -> int:
    return 100#BASE_ATTACK + self.__compute_bonus_attack()
```

```
Traceback (most recent call last):
  File "C:\Users\custo\Desktop\Andre\Others\CTFs\CTFs\2022\TISC22\temp\slay_the_dragon\src\main.py", line 42, in <module>
    run()
  File "C:\Users\custo\anaconda3\envs\dragon\lib\site-packages\click\core.py", line 1130, in __call__
    return self.main(*args, **kwargs)
  File "C:\Users\custo\anaconda3\envs\dragon\lib\site-packages\click\core.py", line 1055, in main
    rv = self.invoke(ctx)
  File "C:\Users\custo\anaconda3\envs\dragon\lib\site-packages\click\core.py", line 1404, in invoke
    return ctx.invoke(self.callback, **ctx.params)
  File "C:\Users\custo\anaconda3\envs\dragon\lib\site-packages\click\core.py", line 760, in invoke
    return __callback(*args, **kwargs)
  File "C:\Users\custo\Desktop\Andre\Others\CTFs\CTFs\2022\TISC22\temp\slay_the_dragon\src\main.py", line 30, in run
    client.run_event(BattleEvent(client=client))
  File
"C:\Users\custo\Desktop\Andre\Others\CTFs\CTFs\2022\TISC22\temp\slay_the_dragon\src\client\gameclient.py", line 33, in run_event

    event.run()

  File
"C:\Users\custo\Desktop\Andre\Others\CTFs\CTFs\2022\TISC22\temp\slay_the_dragon\src\client\event\battleevent.py", line 42, in run

    match self.client.fetch_result():

  File
"C:\Users\custo\Desktop\Andre\Others\CTFs\CTFs\2022\TISC22\temp\slay_the_dragon\src\client\gameclient.py", line 40, in fetch_result

    return Result(self.__recv())

  File "C:\Users\custo\anaconda3\envs\dragon\lib\enum.py", line 385, in __call__

    return cls.__new__(cls, value)

  File "C:\Users\custo\anaconda3\envs\dragon\lib\enum.py", line 710, in __new__

    raise ve_exc

ValueError: '' is not a valid Result
```

Cannot change attack of player to 100

Visit server>service>battleservice

Here we can see that the server creates their own player, own boss class, so we change client side but it doesn't match with server side thus give error in validation.

```
def __handle_battle_win(self):
        self.server.game.remove_next_boss()
        if self.__boss_available_for_next_battle():
            self.server.send_result(Result.VALIDATED_OK)
            return
        self.server.send_result(Result.OBTAINED_FLAG)
        self.server.send_flag()
        self.server.exit()
```

Check client>event>battlevent

We can change things here and it will affect the game. Note that client is sending command with send_command which we can see from client>gameclient that it is

```
    def send_command(self, command: Command):
        self.__send(command.value)
```

Since there was an error in client fetching the result, ValueError: '' is not a valid Result

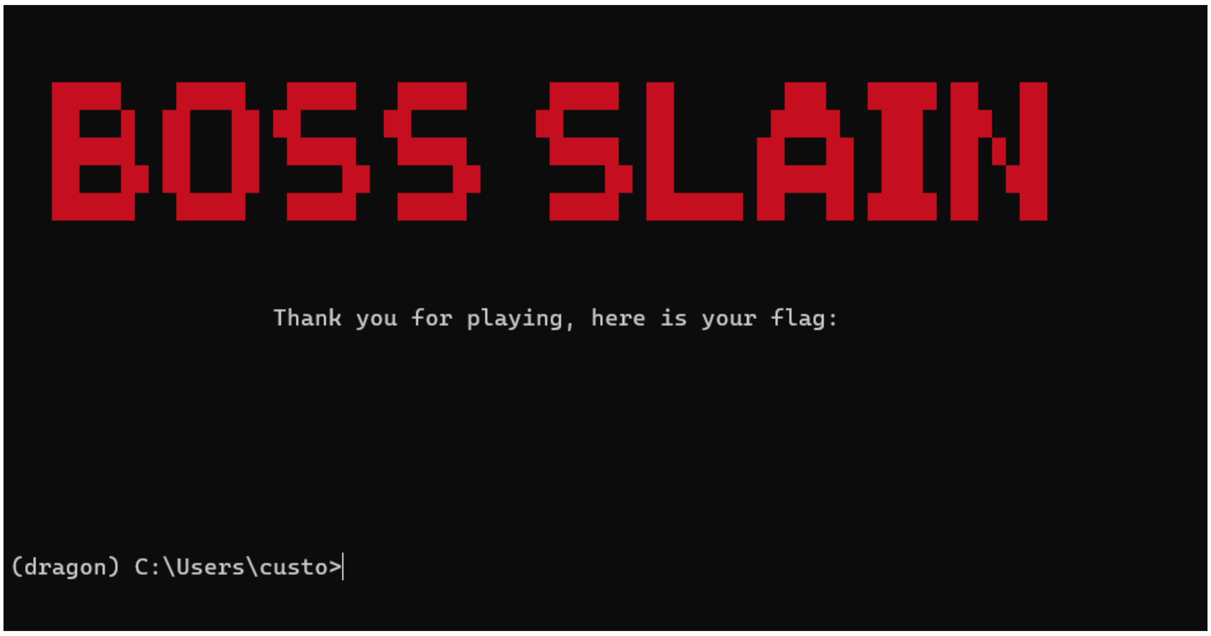
I thought maybe I could set “” as a Result in core>models>result

```
class Result(Enum):
    PLAYER_WIN_BATTLE = "PLAYER_WIN_BATTLE"
    BOSS_WIN_BATTLE = "BOSS_WIN_BATTLE"
    VALIDATED_OK = "VALIDATED_OK"
    CANNOT_AFFORD = "CANNOT_AFFORD"
    PURCHASE_OK = "PURCHASE_OK"
    OBTAINED_FLAG = "OBTAINED_FLAG"
    WORK_OK = "WORK_OK"
    TEST = ""
```

Followed by:

```
        match self.client.fetch_result():
            case Result.VALIDATED_OK:
                screens.display_boss_slain_screen()
                return
            case Result.OBTAINED_FLAG:
                screens.display_flag_screen(self.client.fetch_flag())
                self.client.exit()
            case Result.TEST:
                screens.display_flag_screen(self.client.fetch_flag())
                self.client.exit()
            case _:
                screens.display_error(Error.RECEIVED_MALFORMED_RESULT)
        self.client.exit(1)
```

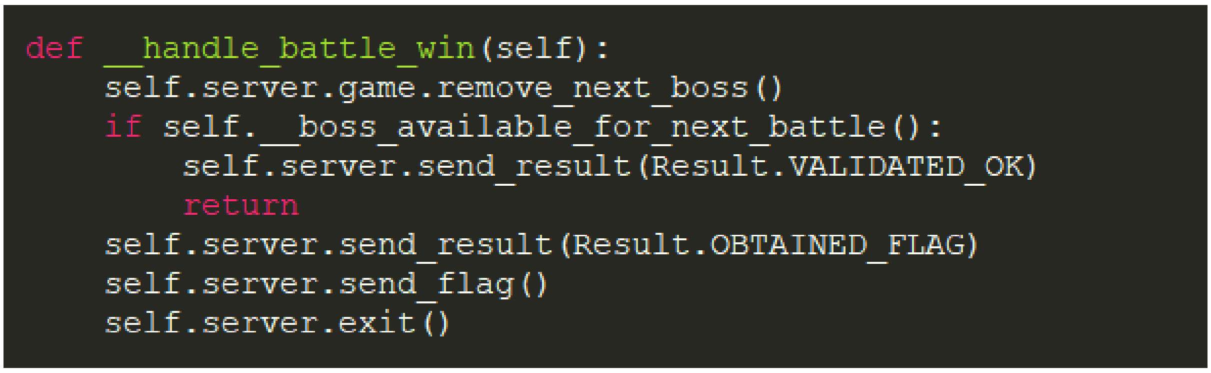
such that when I get invalid result “”, it will be a result of type TEST and thus display the flag screen with the flag.



```
Thank you for playing, here is your flag:

(dragon) C:\Users\custo>
```

But I get empty, because the server has not yet sent the flag



```python
def __handle_battle_win(self):
    self.server.game.remove_next_boss()
    if self.__boss_available_for_next_battle():
        self.server.send_result(Result.VALIDATED_OK)
        return
    self.server.send_result(Result.OBTAINED_FLAG)
    self.server.send_flag()
    self.server.exit()
```

Thus this part of the code needs to be entered for sure.

Vulnerability is in the compute battle outcome where boss is receiving attack first

```
def __compute_battle_outcome(self) -> Optional[Result]:
    for command in self.history.commands:
        match command:
            case Command.ATTACK:
                self.boss.receive_attack_from(self.player)
                if self.boss.is_dead:
```

```
            return Result.PLAYER_WIN_BATTLE
        case Command.HEAL:
          self.player.use_potion()
        case Command.BOSS_ATTACK:
          self.player.receive_attack_from(self.boss)
          if self.player.is_dead:
            return Result.BOSS_WIN_BATTLE
    return None
```

So if case Command.ATTACK comes before Command.BOSS_ATTACK then it will allow me to attack first before the boss. So if I am able to chain 100 attacks such that the boss dies, then it will just handle my victory

However, if we analyze, everytime we attack with the client, or heal, we will append a boss attack

```
 while True:
     self.history.log_commands_from_str(self.server.recv_command_str())

     match self.history.latest:
         case Command.ATTACK | Command.HEAL:
             self.history.log_command(Command.BOSS_ATTACK)
         case Command.VALIDATE:
             break
         case Command.RUN:
             return
         case _:
             self.server.exit(1)
```

Core>models>commands

```
@dataclass
class CommandHistorian:
    commands: List[Command] = field(default_factory=list)

    def log_command(self, command: Command):
        self.commands.append(command)

    def log_commands(self, commands: List[Command]):
        self.commands.extend(commands)

    def log_command_from_str(self, command_str: str):
        self.log_command(Command(command_str))

    def log_commands_from_str(self, commands_str: str):
        self.log_commands(
            [Command(command_str) for command_str in commands_str.split()]
        )
```

It basically takes in a string and splits by space to form a list of strings, then apply the Command over it. So if my string is “ATTACK” then it will be Command.ATTACK. If my string is “ATTACK HEAL” then the list will be [“ATTACK”, “HEAL”] which will be Command.ATTACK , Command.HEAL

In our client, we can only return commands with our match case statement

```python
def __get_battle_command(self) -> Optional[Command]:
    match input():
        case "1":
            return Command.ATTACK
        case "2":
            return Command.HEAL
        case "3":
            return Command.RUN
        case _:
            return None
```

But, we can notice that client can send_command(Command.RUN)

```python
while True:
    self.__display()

    match self.__get_battle_command():
        case Command.ATTACK:
            self.__attack_boss()
            if self.boss.is_dead:
                break
        case Command.HEAL:
            self.__use_potion()
        case Command.RUN:
            self.client.send_command(Command.RUN)
            return
        case _:
            continue
```

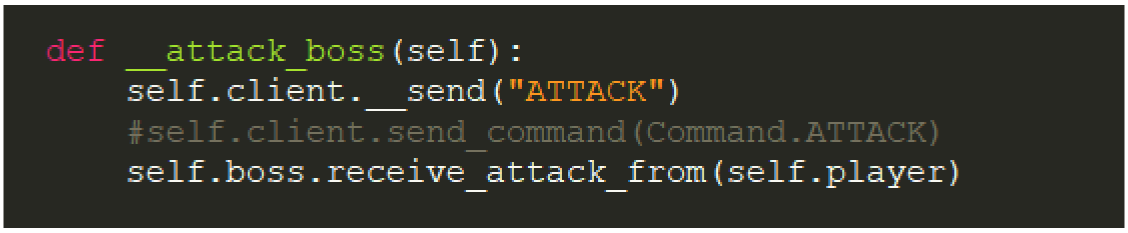
But take note we don’t want to send commands, because remember everytime we send a Command type, we will append BOSS_ATTACK

So why not we instead make use of the log commands and send “ATTACK ATTACK ATTACK…” such that they will convert to Command later

We look at server>gameclient

```
  def send_command(self, command: Command):
    self.__send(command.value)
```

it uses __send, so why not we just call __send instead



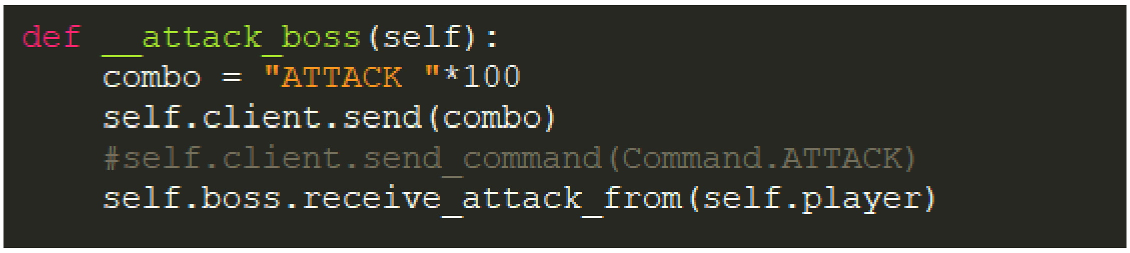
```
def __attack_boss(self):
    self.client.__send("ATTACK")
    #self.client.send_command(Command.ATTACK)
    self.boss.receive_attack_from(self.player)
```

AttributeError: 'GameClient' object has no attribute '_BattleEvent__send'

This is because __ marks a private method (it is a python 3.10 thing but I learn in java), so just change to public access

```
def send_command(self, command: Command):
    self.send(command.value)
  def send(self, data: str) -> int:
    return self.connection.send(data)
```

We can see that it works as per normal, so now instead of just sending attack, we send a combo



```
def __attack_boss(self):
    combo = "ATTACK "*100
    self.client.send(combo)
    #self.client.send_command(Command.ATTACK)
    self.boss.receive_attack_from(self.player)
```

We can see that even though our hp reduce during the battle, but it resets back to full once the battle ends because technically we attacked 100 times first.

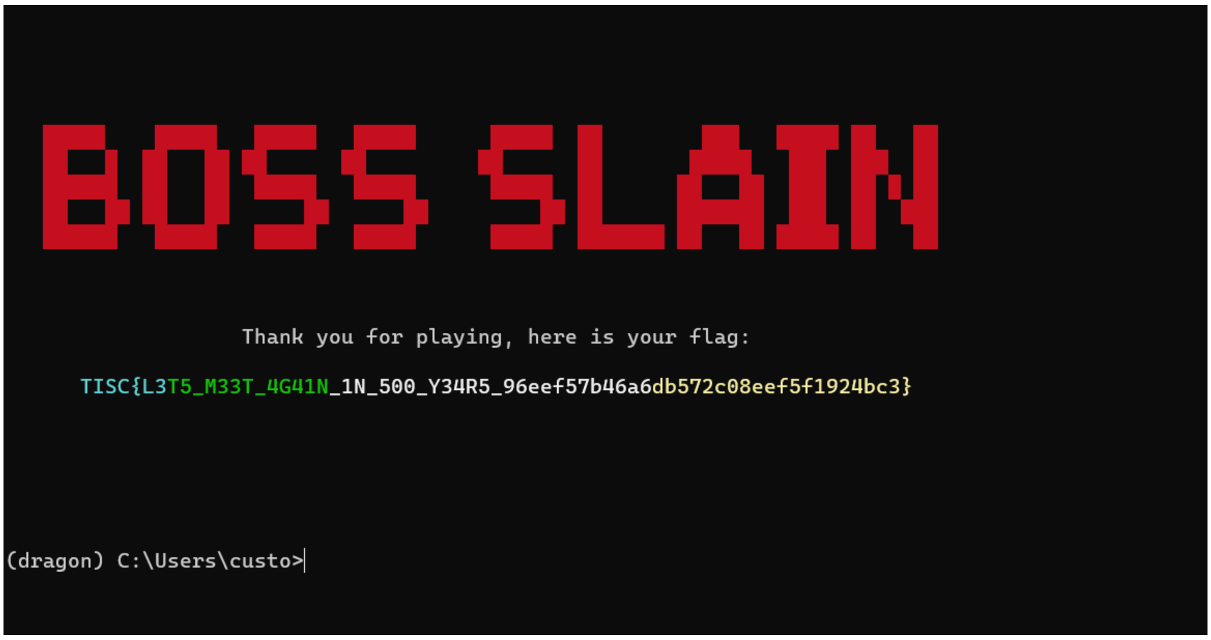
However, since the final boss will one shot us, we need to comment the line out so we take 0 damage because if we die on the client side, there will be a game over screen which make us unable to view the flag even though on server side we won

#self.player.receive_attack_from(self.boss)

We can now also change our attack to 100 because that aligns with what we sent to the server

```
def attack(self) -> int:
    return 100
```

now we can run and kill the final boss



```
Thank you for playing, here is your flag:

TISC{L3T5_M33T_4G41N_1N_500_Y34R5_96eef57b46a6db572c08eef5f1924bc3}

(dragon) C:\Users\custo>
```

Find the working exploit video here: https://www.youtube.com/watch?v=FYw4YDBsXV0

TISC{L3T5_M33T_4G41N_1N_500_Y34R5_96eef57b46a6db572c08eef5f1924bc3}