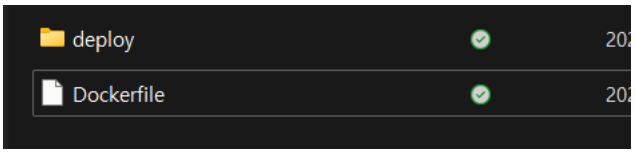
[9월 / Dreamhack Season6] 31기 노희민

태그 ctf 드림핵

Spooky Little Ghost

- 분야: 크립토
- 설명: `Slide` **away from this spooky little GHOST!**



문제를 다운로드 받자 다음과 같이 도커가 나왔다. 우분투 환경에서 도커 빌딩을 시도해보겠다.

도커 컨테이너는 '이미지'를 기반으로 구축된다고 하니, 우분투 가상 머신 안에 도커 컨테이너용 우분투 이미지를 한번 더 다운로드 받았다.

https://ljhyunstory.tistory.com/305

```
[09/12/24]seed@VM:~$ docker pull ubuntu
Using default tag: latest

latest: Pulling from library/ubuntu
31e907dcc94a: Pull complete
Digest: sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e
34a9ab63ee
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

```
[09/12/24]seed@VM:~$ docker images
REPOSITORY                   TAG       IMAGE ID       CREATED       SIZE
ubuntu                       latest    edbfe74c41f8   5 weeks ago   78.1MB
handsonsecurity/seed-ubuntu  large     cecb04fbf1dd   3 years ago   264MB
```

```
FROM ubuntu:22.04@sha25
```

이제 코드의 이 부분을 충족할 수 있을 듯.

이후 도커파일을 옮기려는데, 공유 폴더 설정이 너무 까다로워서 애를 먹었다.

우분투 공유폴더/드래그 앤 드롭 설정에 실패해서 로컬 우분투에서 실행하겠다.

.

우분투 환경설정

우선 autorun.sh 우클릭 메뉴에 run as program이 없어 그냥 터미널에서 실행

```
Verifying archive integrity...  100%   MD5 checksums are OK. All good.
Uncompressing VirtualBox 7.0.20 Guest Additions for Linux  100%
VirtualBox Guest Additions installer
Removing installed version 7.0.20 of VirtualBox Guest Additions...
update-initramfs: Generating /boot/initrd.img-5.15.0-119-generic
update-initramfs: Generating /boot/initrd.img-5.4.0-54-generic
Copying additional installer modules ...
Installing additional modules ...
VirtualBox Guest Additions: Starting.
VirtualBox Guest Additions: Setting up modules
VirtualBox Guest Additions: Building the VirtualBox Guest Additions ker
nel
modules.  This may take a while.
VirtualBox Guest Additions: To build modules for other installed kernel
```

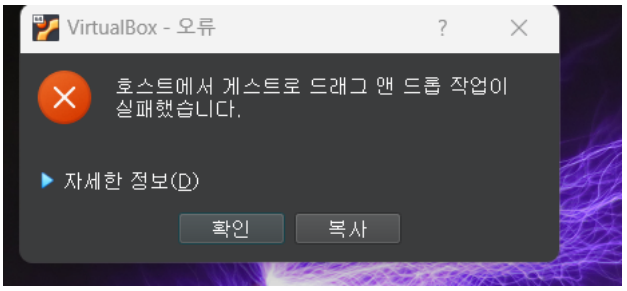
guest additions 설치 중. (이전 버전은 제거한 듯)

제발.... 이번에는 드라이버 충돌 일으키질 않길.... (저번에 이것 때문에 블루 스크린 떠서 수리 센터까지 갔다왔다)

```
VirtualBox Guest Additions:   /sbin/rcvboxadd quicksetup <version>
VirtualBox Guest Additions: or
VirtualBox Guest Additions:   /sbin/rcvboxadd quicksetup all
VirtualBox Guest Additions: Building the modules for kernel 5.15.0-11
generic.
update-initramfs: Generating /boot/initrd.img-5.15.0-119-generic
Press Return to close this window...
```

된 건가?

우선 재시작.



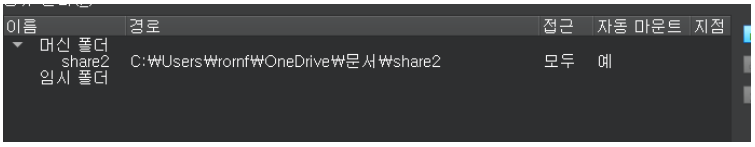
https://weftnwarp.kr/site-it/archives/게스트-os인-우분투-22-04에-게스트-확장-설치-및-필수-설정/

이걸로 다시 시도해보자.

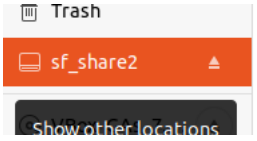
```
09/13/24]seed@VM:.../VBox_GAs_7.0.20$ sudo ./VBoxLinuxAdditions.run
erifying archive integrity...  100%   MD5 checksums are OK. All good.
ncompressing VirtualBox 7.0.20 Guest Additions for Linux  100%
irtualBox Guest Additions installer
emoving installed version 7.0.20 of VirtualBox Guest Additions...
pdate-initramfs: Generating /boot/initrd.img-5.15.0-119-generic
```

autorun 대신 수동 설치 이용. 이번에도 실패하면 그냥 가상 환경에서 문제 파일을 다운로드 받아 풀이하겠다.

+) 공유 폴더 오류 해결

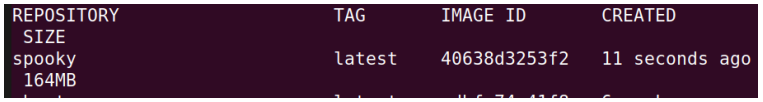


home/share로 지정했던 “지점” value를 삭제하자 자동 마운트 되었다.



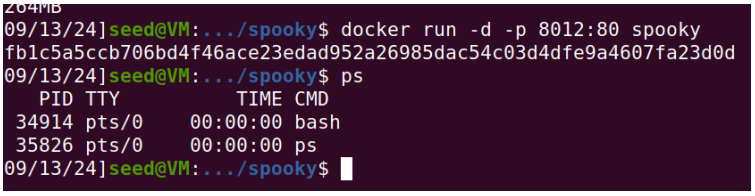
다소 허무하다만, 다행은 다행이다.... (드래그앤드롭은 여전히 안 됨)

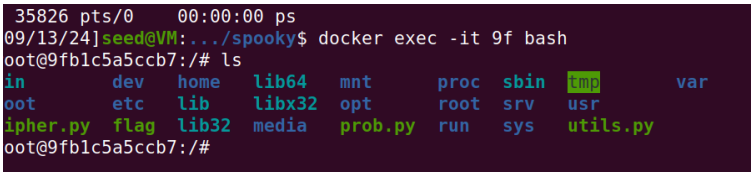
도커



Dockerfile 파일 폴더에서 docker image build

⇒ spooky 이미지 빌딩 완료

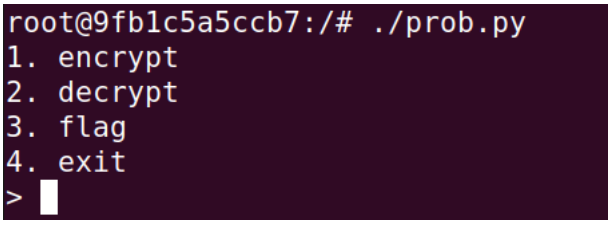




일단 컨테이너를 포트 8012에 올리고 접속?을 했다.

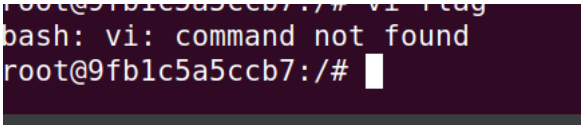
docker exec -it (도커 ID 앞글자 2자리) bash

→ 컨테이너 접속



./prob.py 출력

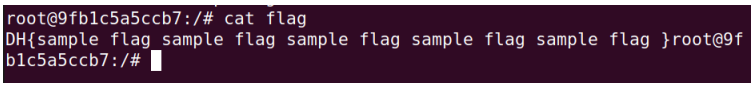
암호화 → 복호화 → 플래그 → 퇴장



vi flag로 문서의 내용을 보고자 했는데 vi가 설치되어 있지 않다.

https://javacpro.tistory.com/75

해당 문서를 참고해 도커 컨테이너에 vi를 설치해 보자.



cat으로 보자 보임. 플래그의 형식인듯.

먼저 cipher.py (암호문으로 추정)

.

```
from utils import *
```

```
class GHOST:
    sbox = (
        (0xC, 0x4, 0x6, 0x2, 0xA, 0x5, 0xB, 0x9, 0xE, 0x8, 0xD, 0x7, 0x0, 0x3, 0xF, 0x1),
        (0x6, 0x8, 0x2, 0x3, 0x9, 0xA, 0x5, 0xC, 0x1, 0xE, 0x4, 0x7, 0xB, 0xD, 0x0, 0xF),
        (0xB, 0x3, 0x5, 0x8, 0x2, 0xF, 0xA, 0xD, 0xE, 0x1, 0x7, 0x4, 0xC, 0x9, 0x6, 0x0),
        (0xC, 0x8, 0x2, 0x1, 0xD, 0x4, 0xF, 0x6, 0x7, 0x0, 0xA, 0x5, 0x3, 0xE, 0x9, 0xB),
        (0x7, 0xF, 0x5, 0xA, 0x8, 0x1, 0x6, 0xD, 0x0, 0x9, 0x3, 0xE, 0xB, 0x4, 0x2, 0xC),
        (0x5, 0xD, 0xF, 0x6, 0x9, 0x2, 0xC, 0xA, 0xB, 0x7, 0x8, 0x1, 0x4, 0x3, 0xE, 0x0),
        (0x8, 0xE, 0x2, 0x5, 0x6, 0x9, 0x1, 0xC, 0xF, 0x4, 0xB, 0x0, 0xD, 0xA, 0x3, 0x7),
        (0x1, 0x7, 0xE, 0xD, 0x0, 0x5, 0x8, 0x3, 0x4, 0xF, 0xA, 0x6, 0x9, 0xC, 0xB, 0x2)
    )

    def __init__(self, key: bytes) -> None:
        self._block_size = 8
        self._round_keys = self._expand_key(key)
        self._state = []

    def _expand_key(self, key: bytes) -> list[list[int]]:
        assert len(key) == 8
        key_bits = bytes_to_bits(key)
        round_keys: list[list[int]] = []
        for i in range(0, 64, 32):
            round_keys.append(key_bits[i:i+32])
        return round_keys

    def _substitution(self, bit32: list[int]) -> list[int]:
        res: list[int] = []
        for i,j in enumerate(range(0,32,4)):
            res += int_to_bits(self.sbox[7-i][bits_to_int(bit32[j:j+4])], 4)
        return res

    def _round_function(self, round_n: int, is_enc: bool) -> None:
        round_key_idx = round_n%2

        state_hi = self._state[:32]
        state_lo = self._state[32:]

        state_lo = add_mod_2_32(state_lo, self._round_keys[round_key_idx])
        state_lo = self._substitution(state_lo)
        state_lo = rol11(state_lo)
        state_lo = xor_lst(state_lo, state_hi)

        if (is_enc and round_n == 31) or (not is_enc and round_n == 0):
            self._state[:32] = state_lo
        else:
            self._state[:32] = self._state[32:]
            self._state[32:] = state_lo

    def _encrypt(self, plaintext: bytes) -> bytes:
        self._state = bytes_to_bits(plaintext)
        for i in range(32):
            self._round_function(i, is_enc=True)
        return bits_to_bytes(self._state)

    def _decrypt(self, ciphertext: bytes) -> bytes:
        self._state = bytes_to_bits(ciphertext)
        for i in range(31, -1, -1):
            self._round_function(i, is_enc=False)
```

```
        return bits_to_bytes(self._state)

    def encrypt(self, plaintext: bytes) -> bytes:
        assert len(plaintext)%self._block_size == 0
        ciphertext  = b''
        for i in range(0, len(plaintext), self._block_size):
            ciphertext += self._encrypt(plaintext[i:i + self._block_size])
        return ciphertext

    def decrypt(self, ciphertext: bytes) -> bytes:
        assert len(ciphertext)%self._block_size == 0
        plaintext  = b''
        for i in range(0, len(ciphertext), self._block_size):
            plaintext += self._decrypt(ciphertext[i:i + self._block_size])
        return plaintext

def test():
    import os
    trial = 0x1000
    for _ in range(trial):
        key = os.urandom(8)
        g = GHOST(key)
        plain = os.urandom(8)
        cipher = g.encrypt(plain)
        assert plain == g.decrypt(cipher)

if __name__ == '__main__':
    test()
```

암호문에 import된 암호화 알고리즘 utils.py

.

```
def xor_lst(a: list[int], b: list[int]) -> list[int]:
    return [x^y for x,y in zip(a,b)]

def xor_bytes(a: bytes, b: bytes) -> bytes:
    return bytes([x^y for x,y in zip(a,b)])

def int_to_bits(d: int, bits_len:int = 8) -> list[int]:
    return [int(x) for x in bin(d)[2:].zfill(bits_len)]

def bits_to_int(bits: list[int]) -> int:
    return int(''.join([str(x) for x in bits]),2)

def bytes_to_bits(m: bytes) -> list[int]:
    bits = []
    for b in m:
        bits += [int(x) for x in bin(b)[2:].zfill(8)]
    return bits

def bits_to_bytes(bits: list[int]) -> bytes:
    return bits_to_int(bits).to_bytes(len(bits)//8, 'big')

def add_mod_2_32(bit32: list[int], key32: list[int]) -> list[int]:
    return int_to_bits((bits_to_int(bit32) + bits_to_int(key32)) % (2**32), 32)
```

```
def rol11(bit32: list[int]) -> list[int]:
    return bit32[11:]+bit32[:11]
```

prob.py

```
#! /usr/bin/env python3
import os
from cipher import GHOST
from utils import *
#cipher에서 GHOST import

def menu() -> int:
    print('1. encrypt')
    print('2. decrypt')
    print('3. flag')
    print('4. exit')
    return int(input('> '))
    #input값 int 변환해 return

if __name__ == '__main__':
    with open('flag', 'rb') as f:
        flag = f.read()
  #'flag'파일 데이터를 바이너리로 변환해 flag 변수에 저장

    key = os.urandom(8)
    #무작위 8바이트 비밀 키
    g = GHOST(key)
    #

    while True:
        i = menu()
        #입력값 i에 저장
        if i == 1:
            msg = input('plaintext(hex)> ').
            enc = g.encrypt(bytes.fromhex(msg))
            print('ciphertext(hex)>', enc.hex())
            #1 입력시 -> 평문 입력받아 암호화, 결과 hex로 출력.
        elif i == 2:
            enc = input('ciphertext(hex)> ')
            msg = g.decrypt(bytes.fromhex(enc))
            print('plaintext(hex)>', msg.hex())
            #2 입력시 -> 암호문 입력받아 복호화, 결과 hex로 출력.

        elif i == 3:
            new_key = xor_bytes(key, bytes.fromhex('deadbeefcafebabe'))
            new_g = GHOST(new_key)
            enc_flag = new_g.encrypt(flag)
            print('encrypted_flag(hex)> ', enc_flag.hex())
        else:
            break
```

우선 docker stop 하고 코드를 해석해 보자.

https://velog.io/@bbangi/Docker-생성-실행-종료

ptog.py에서 3번을 실행해 새로운 암호화 키 획득.

encrypted_flag(hex)>
bd93c6e19af586cf840b42d75d51a1d5b9381ae84c47d6457d8582167e31b436840b42d75d51a1d5b9381ae84c47d6457

복호화할 암호문을 먼저 찾아야 하는 거 아닐까??

암호화 옵션이 제대로 작동하지 않길래 일단 입력받은 암호화 키를 복호화 해봤다.

복호화(HEX):

73578a67b754b41934698afb2c1df1e4c16d99a3a014939f37c3def4cea23dfd34698afb2c1df1e4c16d99a3a014939f37c3

문제의 힌트는 SLIDE라고 돼 있긴 했는데... 이거 실제로 사용하는 명령언가?

크립토에 대해서는 아는 게 하나도 없어서 더 이상은 진행을 못 하겠다. 개인적으로는 드디어 도커를 사용법을 익힌 것에 의의를 두고 싶음.