디지털

도구 이해

Otter CTF 5번 문제

Name : 조민혁

Class Number : 5

Phone Number : 010-4923-2198

목 차

1. Otter CTF 5번 문제

1-1. 코드 설명

```python
import re

with open("pid.708.dmp", "rb") as f:
    data = f.read()

pattern = re.compile(b'\x64.{6,8}\x40\x06.{18}\x5a\x0c\x00\x00')
matches = pattern.finditer(data)

for match in matches:
    offset = match.start()
    after = data[match.end() : match.end() + 64]
    matched = match.group()

    total_bytes = matched + after

    decoded_text = ''.join(
        chr(b) if 0x20 <= b <= 0x7E else '.' for b in total_bytes
    )


    hex_pairs = [total_bytes[j:j+2].hex() for j in range(0, len(total_bytes)
, 2)]


    spaced_output = ' '.join(hex_pairs)


    print(f"[+] Offset: {match.start():#x}")
    print(spaced_output)
    print(f"\n{decoded_text}")
```

그림 1. 필터링을 위한 Python 코드

그림 1은 Otter CTF 5번 문제에서 708번 PID를 덤프한 결과를 필터링 하기 위한 코드이다. 정규 표현식을 이용하기 위해 import re를 하였다. 먼저 제시된 정규 표현식을 컴파일하여 객체로 반환하기 위해 re.compile 함수를 호출하였고 매치된 결과를 matches 변수에 저장하였다. 그리고 해당 시그니처 뒤의 바이트를 파악하기 위한 64바이트를 추가로 가져와서 after 변수에 저장했다. 이후 바이트 값을 Ascii 값으로 복호화하여 decoded_text 변수에 저장 후 출력했다.

1-2. 필터링 결과

그림 2. Otter CTF 5번 필터링 결과

그림 2는 그림 1에서 작성한 파이썬 코드의 실행 결과를 보여준다. 정상적으로 바이트 코드를 출력하는 것을 확인할 수 있었고, 해당 바이트 값을 ASCII로 변환하였을 때 M0rtyL0L 이라는 값을 출력하는 것을 알 수 있었다.