# Narnia

## Informações gerais

"narnia.labs.overthewire.org through SSH on port 2226"
Todos os níveis têm seus programas e códigos em /narnia.

### narnia0

```
$ ( echo -e "0123456789012345678\xef\xbe\xad\xde" ; cat ; ) | /narnia/narnia0
```

### narnia1

colocar um shellcode na variavel de ambiente EGG para ler o arquivo /etc/narnia_pass/narnia2 (vide shellcode.asm em anexo)

e.g. para injetar 0x01 0x02:

```
        $ export EGG="$(echo -ne '\x01\x02')"
```

Conferimos o shellcode com

```
        $ echo -n "$EGG" | hedxump -C
```

### narnia2

stack smashing: escrever um número N de bytes, seguido do endereço B do buffer, para sobrescrever o endereço de retorno da função na stack frame, igual ao jumpy2 do limbo. Colocamos um shellcode nesses N bytes para ler /etc/narnia_pass/narnia3.

O valor de N é a diferença entre o endereço B do buffer e o endereço R da pilha no qual está guardado o endereço de retorno de main().

Para achar R: Rodando no gdb podemos ver centenas de bytes a partir do endereço que estiver em esp imediatamente antes da chamada de strcpy(). Com isso acharíamos R, mas não sabemos de qual endereço main() foi chamada. Sabemos que foi da função __libc_start_main(), então basta ver os endereços do começo e do final dessa função e procurar na pilha um valor nesse intervalo, e cuja instrução imediatamente anterior seja call.

Para checar que o valor de N está certo, rodamos de novo no gdb, com N bytes seguidos de 4

bytes fixos, e vemos se o programa vai parar lá. Fazendo isso, vemos que N = 140.

Finalmente, precisamos achar B, que varia de acordo com o tamanho da string que passamos como argumento. Agora, já sabemos que devemos escrever uma string com N+4 bytes (porque queremos N bytes seguidos do endereço B, e como o programa está em 32 bits, B tem 4 bytes). Então para achar B, basta rodar narnia2 no ltrace, com uma string de N+4 bytes como argumento; B será o endereço mostrado como destino de strcpy(). Depois é só escrever o shellcode para ler o arquivo que da senha narnia3, preencher os bytes que faltam com NOP para ter N bytes, e colocar o endereço B em seguida. Passamos o shellcode da mesma maneira que o anterior, com

```
$(echo -ne '\x<byte1>\x<byte2>...')
```

Vide shellcode2.asm em anexo.

# narnia3

outro stack smashing, mas dessa vez devemos sobrescrever a string ofile (vide /narnia/narnia3.c) para que a senha seja escrita em um arquivo que podemos ler. Ao mesmo tempo, temos que passar como ifile o caminho do arquivo que tem a próxima senha: /etc/narnia_pass/narnia4.

Para sobrescrever o ofile, basta colocar várias '/' depois do caminho do ifile, até dar 32 caracteres; tudo depois disso será visto como o ofile. Por exemplo:

```
$ /narnia/narnia3 "/etc/narnia_pass/narnia4////////tmp/foo"
                   ^1                            32^
```

O problema: a string do ofile também vai fazer parte do ifile. Nesse caso o ofile será /tmp/foo e o ifile será a string inteira: /etc/narnia_pass/narnia3////////tmp/foo, porque a string só acaba no terminador nulo de ofile. Como estamos passando essa string como um parâmetro no bash, não dá para colocar um '\0' antes de começar o ofile, porque esses parâmetros também são armazenados como strings, e um '\0' no meio da string faria com que ela acabasse antes do que devia.

Primeiramente, o comando acima é equivalente a

```
$ /narnia/narnia3 ////////etc/narnia_pass/narnia4/tmp/foo
```

Contanto que o número de barras seja igual. O que fizemos, então, foi usar caminhos relativos a nosso favor: em vez de tentar escrever em /tmp/foo, podemos criar um arquivo /tmp/narnia4 e rodar, em /tmp, esse comando:

```
$ /narnia/narnia3 "..//////////////etc/narnia_pass/narnia4"
                   ^1                            32^
```

De modo que o ofile seria narnia4 (ou seja, /tmp/narnia4, porque estamos em /tmp) e o ifile seria /tmp/..///////////////etc/narnia_pass/narnia4 (novamente porque estamos em /tmp). Daí vem um segundo problema: o arquivo /tmp/narnia4 já existe, e é um link simbólico para /etc/narnia_pass/narnia4; não podemos escrever nele. A maneira mais fácil de resolver isso é colocando mais 2 barras...

```
$ /narnia/narnia3 "..////////////////etc/narnia_pass/narnia4"
                   ^1                            32^
```

...para que o ofile passe a ser s/narnia4. Então basta criar a pasta s/ e dentro dela o arquivo narnia4, rodar o comando, e ler s/narnia4.

---

Revision #4

Created Fri, Oct 26, 2018 7:36 PM by Luana

Updated Wed, Dec 12, 2018 7:41 PM by Luana