03/07/2023 – PicoCTF – Stonks

Link: <https://play.picoctf.org/practice/challenge/105?page=1&search=>

1. Ao pegar o arquivo em “C” dado pelo desafio, verifica-se uma “format string vulnerability”:

Texto

Descrição gerada automaticamente

Em “printf(user\_buf)”, o programa pega a string passada pelo usuário (em “printf(“%300s”, user\_buf)”, e a exibe. Porém, como não existe um “string specifier” neste printf, dependendo do input do user, pode ser explorada uma vulnerabilidade de se verificar os dados armazenados no buffer.

1. Utilizar-se dessa vulnerabilidade para checar o buffer. Como passado no enunciado do desafio, conectar-se via “Netcat” com “mercury.picoctf.net 53437”:

Texto

Descrição gerada automaticamente

1. Ao analisar o código, verifica-se que a vulnerabilidade pode ser encontrada na opção 1, pois ela que chamará a função “buy\_stonks”:

Texto

Descrição gerada automaticamente

Ao acessar a primeira opção:

Texto

Descrição gerada automaticamente

Verificando os valores em hexadecimal armazenados no buffer:

Texto

Descrição gerada automaticamente

1. Pegando o valor em hexa e transformando em letras (ASCII):

Link: <https://www.rapidtables.com/convert/number/hex-to-ascii.html>

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Verifica-se a seguinte string: ocip{FTC0l\_I4\_t5m\_ll0m\_y\_y3n4cdbae52ÿ }que armazena a flag. Porém, percebe-se, a partir de “ocip”, que há uma inversão da string a cada 4 letras. Portanto, deve-se revertê-las

1. Revertendo as letras usando Python:

Texto

Descrição gerada automaticamente

1. Verifica-se que há 2 caracteres especiais “ü÷” que não pertencem à verdadeira flag e, portanto, ao removê-los, chega-se na flag: $flag