EasyBox

发表于 2021-01-04 分类于 Challenge， 2019， UNCTF， Misc
Challenge | 2019 | UNCTF | Misc | EasyBox

点击此处获得更好的阅读体验

题目考点

- *深度优先搜索算法（英语：Depth-First-Search，简称DFS）是一种用于遍历或搜索树或图的算法。 沿着树的深度遍历树的节点，尽可能深的搜索树的分支。当节点v的所在边都已被探寻过或者在搜寻时结点不满足条件，搜索将回溯到发现节点v的那条边的起始节点。整个进程反复进行直到所有节点都被访问为止。属于盲目搜索,最糟糕的情况算法时间复杂度为O(!n)。*
- *Pwntools提供了方便的网络交互编程的接口*

解题过程

使用nc连接到靶机开放端口。返回结果如下所示，可以看出是一个数独之类的游戏，但是交互时间很短，只能通过编写脚本来完成：

利用dfs（深度优先搜索算法）来编写计算数独空缺数字脚本，根据题目提示，这个数独只需要横向和纵向的数字和为45，并且1-9只能出现一次，部分脚本如下所示：

exp.py使用pwntools库负责接收数据和发送数据，solve.py负责将数据整理并利用 exp.py

```python
#coding:utf-8
from pwn import *
import datetime
import sovle
#context.log_level='debug'
p=remote('123.206.21.178',10000)
shudu=[]
p.recvuntil("+-+-+-+-+-+-+-+-+-+")
p.recvuntil("|")
shudu.append(p.recv(17).strip("\n").split('|'))
print shudu
p.recvuntil("+-+-+-+-+-+-+-+-+-+")
p.recvuntil("|")
shudu.append(p.recv(17).strip("\n").split('|'))
p.recvuntil("+-+-+-+-+-+-+-+-+-+")
p.recvuntil("|")
shudu.append(p.recv(17).strip("\n").split('|'))
p.recvuntil("+-+-+-+-+-+-+-+-+-+")
p.recvuntil("|")
shudu.append(p.recv(17).strip("\n").split('|'))
p.recvuntil("+-+-+-+-+-+-+-+-+-+")
p.recvuntil("|")
shudu.append(p.recv(17).strip("\n").split('|'))
p.recvuntil("+-+-+-+-+-+-+-+-+-+")
p.recvuntil("|")
shudu.append(p.recv(17).strip("\n").split('|'))
p.recvuntil("+-+-+-+-+-+-+-+-+-+")
p.recvuntil("|")
shudu.append(p.recv(17).strip("\n").split('|'))
p.recvuntil("+-+-+-+-+-+-+-+-+-+")
p.recvuntil("|")
shudu.append(p.recv(17).strip("\n").split('|'))
p.recvuntil("+-+-+-+-+-+-+-+-+-+")
p.recvuntil("|")
shudu.append(p.recv(17).strip("\n").split('|'))
res=[]
for i in range(9):
    res.append(['0' if x==' ' else x for x in shudu[i]])
m=''
for i in range(9):
    for j in range(9):
        res[i][j]=int(res[i][j])
        if res[i][j]==0:
            m+=str(i)+'.'+str(j)+' '
print res
m = m[:-1].split(' ')
print m
res = sovle.start(res)
answer=[]
for i in range(9):
    s=''
    for x in m:
        if i == int(x[0]):
            s += str(res[int(x[0])][int(x[2])])+' '
    answer.append(s[:-1])
print answer
p.recv()
for i in answer:
    payload = i.replace(' ',',')
    print payload
    p.sendline(payload)
p.recvuntil('')
p.interactive()
```

solve.py

```
#coding:utf-8
def my_dfs(A):
    result=[]
    ALL_SET=set({1,2,3,4,5,6,7,8,9,0})#全集
    for i in range(9):
        for j in range(9):
            if A[i][j]==0:
                d={}
                set_x=set(A[i][k] for k in range(9))#横向的集合
                set_y=set(A[k][j] for k in range(9))#纵向的集合
                set_x_y=set_x|set_y
                data=list(ALL_SET-set_x_y)#总的集合-横向纵向的集合=可能解的集合
                d['%s,%s'%(str(i),str(j))]=data
                result.append(d)
    return result
#判断数独中是否含有未确定的数字
def check(A):
    flag=False
    for i in range(9):
        for j in range(9):
            if A[i][j] is 0:
                flag=True
    #A中有0为True
    return flag
def start(A):
    count=1
    while check(A):
        data=my_dfs(A)
        for each in data:
            for index,item in each.items():
                if len(item)==1:#如果可能解只有一个，那么该点就是这个解
                    i,j=index.split(',')
                    A[int(i)][int(j)]=item[0]#将这个解赋值给该点
        count+=1
    return A
if __name__=='__main__':
    A = [
        [0, 8, 9, 0, 1, 6, 0, 2, 0],
        [8, 0, 3, 0, 4, 0, 7, 0, 6],
        [7, 1, 2, 9, 0, 8, 0, 4, 5],
        [4, 7, 0, 6, 9, 5, 3, 1, 2],
        [0, 6, 7, 0, 8, 4, 2, 9, 0],
        [0, 0, 0, 4, 7, 0, 0, 0, 9],
        [0, 0, 5, 3, 6, 0, 0, 7, 0],
        [0, 0, 0, 0, 5, 1, 8, 6, 7],
        [6, 9, 1, 8, 2, 7, 0, 0, 4]
    ]
    print start(A)
```

深度优先搜索算法来得出空缺的数字，最后由exp.py发送，结果如下：

- ***本文作者：*** *CTFHub*
- ***本文链接：*** *https://writeup.ctfhub.com/Challenge/2019/UNCTF/Misc/iMHxRBrym7WhsPvwMm6q6s.html*
- ***版权声明：*** *本博客所有文章除特别声明外，均采用 BY-NC-SA 许可协议。转载请注明出处！*

 # Challenge # Misc # 2019 # UNCTF
BACON
Happy_puzzle