虎符CTF2022 handle - MISC

原创

ShallowF 于 2022-03-23 09:56:47 发布 3837 收藏

文章标签： 网络安全

虎符CTF2022 | MISC | handle

> (2022数字中国创新大赛虎符网络安全赛道)
> Part of【COMPASS CTF】 WriteUp
> by Frankss@COMPASS

handle (二血 909pt)

思路

~~（隔壁某show平台上周刚做过某套神类似的题，于是很快就把字典搞出来了，但是交互写了很久丢掉了一血）~~
思路就是找一个有优势的固定词开头，然后根据返回结果分枝(枝)，根据不同枝选择不同的尝试（剪枝），这样接下来要处理的重复量就少很多，重复这个过程两次，几乎每一个可能的词都有对应唯一序列（key）了。
简单统计一下生母韵母音调的频率，但是依据这个找出比较常见的词跑三轮之后会有400+个重复的路线，也就是这400+个词如果抽到大概率失败，算一下成功率`pow(1-400/26000,512)`是恐怖的万分之三，于是random choice字典里的词开始跑。

跑了十几分钟找到 **露己扬才** 只有一百多重复，成功率`pow(1-100/26000,512)`已经提升到了十分之一，决定多跑几次出flag。
P.S.因为成功之后就没有 **>** 输出了，所以边界条件炸了，在第512轮强制进交互赌第二轮出结果。

脚本

首先是生成字典的函数，kk是第一个固定的开头词，根据每次返回的文本更新键值对，然后对有多个结果的key迭代延长，每次选择第一个可能的结果，最后的重复个数即为有可能失败的词的数量：

```
# 其余内容和源代码完全一样，节省空间就不粘贴了
with open('idioms.txt', 'r', encoding='utf8') as f: # utf8
    idioms = [x.strip() for x in f.readlines()]

def check(guess, answer): # 魔改的check，改返回值为输出内容 方便到时候直接用服务器返回内容更新
    guesspy = get_pinyin(guess)
    answerpy = get_pinyin(answer)
    r = ""
    py_results = [check_part(guesspy[i], answerpy[i]) for i in range(3)]
    for i in range(4):
        for j in range(3):
            r += (wrap_color(guesspy[j][i], py_results[j][i]))
        r += ' '
    r += '\n'
    results = check_part(guess, answer)
    for i in range(4):
        r += wrap_color(guess[i], results[i])
    r += '\n'
    return r.encode(), r
```

```python
def gen(kk):
    d = {}
    dup = []
    for i in idioms:
        s = check(kk, i)[0]
        if s in d.keys():
            d[s].append(i) # 其实这里用s的哈希也是可以的，但是不方便debug，而且提速不明显
            dup.append(s)
        else:
            d[s] = [i]
    print(f'finish init round1 with {len(dup)} dup.')
    dup = set(dup)
    dup2 = []
    for i in dup:
        for j in d[i]:
            s = check(d[i][0], j)[0]
            if s in d.keys():
                d[s].append(j)
                dup2.append(s)
            else:
                d[s] = [j]
    print(f'finish init round2 with {len(dup2)} dup.')
    dup2 = set(dup2)
    dup3 = []
    for i in dup2:
        for j in d[i]:
            s = check(d[i][0], j)[0]
            if s in d.keys():
                d[s].append(j)
                dup3.append(s)
            else:
                d[s] = [j]
    print(f'finish init round3 with {len(dup3)} dup.')
    return d

while True:
    s=random.choice(idioms)  # 因为遍历不完，连续的词特征重复性高，所以随机抽了
    print(s)
    d = gen(s) # 如果第三个结果能小于200就能用了，多跑几轮肯定能拿到flag
    l = open('l.pickle', 'wb1')
    pickle.dump(d, l)
    l.close()
```

然后是多次尝试的利用脚本

```python
l = open('l.pickle', 'rb') # 上边生成的
d = pickle.load(l)
# context.log_level = 'debug'

while True:
    try:
        p = remote("120.77.30.1", 48771)
        p.recvuntil(b"> ")
        for r in range(512):
            print(r)
            p.sendline('露己扬才'.encode())
            res = p.recvuntil(b"> ")
            while b'Round' not in res:
                s = res[:-2]
                p.sendline(d[s][0])
                if r == 511:
                    p.interactive() # 边界条件炸了，在第512轮强制进交互赌第二轮出结果
                res = p.recvuntil(b"> ")
    except:
            time.sleep(1)

p.interactive()
```

食用

因为疫情不在学校，交互提速的小trick就是去IP所在地租个服务器，比如这里租个广东的阿里云或者腾讯云的服务器，交互飞快，几秒跑一轮，体验如同本地一般。

首先用上边的脚本生成一个小点（<150）的数据集，然后第二个交互脚本跑五六分钟就出来了（~~没有优化边界，逃子逃子~~ 。