ACTF Dice Game Writeup

原创

zh_explorer 于 2015-05-04 23:39:12 发布 695 收藏

分类专栏： 没事撸题 文章标签： python ctf writeup

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。
本文链接：https://blog.csdn.net/zh_explorer/article/details/45488917
版权

没事撸题 专栏收录该内容
3 篇文章 0 订阅
订阅专栏

ACTF也算是结束了，浙大的那帮人脑洞也是够大的 π__π 这题目也是坑得可以。。。像我这样的渣渣只能是挑所有题目中最简单的来了。

这次的ctf好坑啊，竟然所有逆向破解题都是apk，我完全不会啊(>__<)，只能弄到ppc了。

Dice Game 题目直接把python的源码贴了出来。源码如下：

```
#!/usr/bin/env python
# coding: utf-8
"""
Yet another problem for ACTF2015.
By H4sIAOzUQ1UCAwtIzUuvLFVwzkjNU9AoAHMccjKTiksSi4oy0yr1ivM1uQCpNCsJJAAAAA==
COPYLEFT, ALL WRONGS RESERVED.
"""

import random
import time
import sys
assert sys.version_info >= (3, 2)


NUM_ROUNDS = 1000
SCORE_INITIAL = 100
SCORE_ROUND = 10
SCORE_BONUS = 2


class WeBreakup(Exception):
    """Hmmm.. :("""
    pass


def play_round(scores):
    """Plays a single round. Returns round status and new scores."""
    print('..and your guess is?')
    guess = input()
    point = random.randint(1, 6)
    mapping = {x: 'small' if x <= 3 else 'big' for x in range(1, 7)}
    if guess == mapping[point]:
        print('Correct. :(')
        guessing_correct = True
```

```
            guessing_correct = True
            # You know I'm the boss, right?
            score_delta = SCORE_ROUND - SCORE_BONUS
            scores = (scores[0] - score_delta, scores[1] + score_delta)
        else:
            print('Incorrect! :)')
            guessing_correct = False
            # You know I'm the boss, right?
            score_delta = SCORE_ROUND + SCORE_BONUS
            scores = (scores[0] + score_delta, scores[1] - score_delta)
        return (guessing_correct, scores)


    def play_game():
        """Plays a game."""
        print('Hey hey, you you, I wanna play a game! :)')
        reply = input()
        if reply not in ('Sure!', 'Okay!', 'With pleasure. :)'):
            raise WeBreakup('You bad guy :(')
        scores = tuple(SCORE_INITIAL for i in range(2))
        random.seed(int(time.time() * 1e5))
        recently_is_beaten = []
        print('Well, we play a simple dice game! :)')
        for i in range(NUM_ROUNDS):
            is_beaten, scores = play_round(scores)
            print('After round %d the scores are %s' % (i, scores))
            if scores[0] <= 0:
                raise WeBreakup('I feel humiliated. :(')
            recently_is_beaten.append(is_beaten)
            if recently_is_beaten[-5:].count(True) >= 5:
                raise WeBreakup('How dare you beat me 5 times in a streak! :(')
        if scores[0] < scores[1]:
            raise WeBreakup('Do you think it is more important to win this game than making me happy? :(')
        elif scores[1] > scores[0]:
            raise WeBreakup('You are still too young too simple! :(')
        else:
            print(u'I \u2661 U!')
            response = input()
            if response == 'Me too! <3':
                key_f = open('./flag', 'r')
                print('Flag: %s' % key_f.read().strip())
                key_f.close()


    if __name__ == '__main__':
        play_game()
```

确实算是个掷筛子的游戏，向服务器发送“big”或者“small”猜大小。双方各100分，猜对我方加分8分，猜错扣12分，对方反之。猜1000次。如果最后两人的分数一样，则打印flag。一个两人零和的游戏。但是猜对概率是二分之一。而且还有2条限制条件

1.不能连赢5次

2.对方分数不能小于0

有这么多限制，在概率上达到平局的概率基本上是不可能了。虽然可以发送些“aaa”之类的可以保正出错。但是无论如何，我方是必输的。。。╮（￣▽￣）╭

所以只能是另外想办法了。

然后注意到了这个random.seed(int(time.time() * 1e5))这个随机函数是以系统时间来做seed的。相信了解过随机函数的都知道，只要知道了一个随机函数的seed和函数的算法，那么随机函数的值是完全可以预测的。问题就是得到int(time.time() * 1e5)的值了。不过因为网络延时和系统时间的不同，再加上坑爹的1e5次。所以要同步时间是个大问题。

所以先写了这么两段代码

```
import socket
import time
import random
list = []
scores = 0
a=0
str=""
s = socket.socket()
s.connect(('119.254.101.232',9999))

print (s.recv(100))
s.send(b'Sure!\n')
print (s.recv(100))

Seed=int(time.time() * 1e5)
i=0
right = 0
while i<20:
    s.send(b"big\n")
    h=s.recv(100)
    print (h)
    if h[0] == 67:
        list = list + [1]
        scores += 1
    else:
        list = list + [0]
    i+=1
print (v,list)
s.close()
```

第一段代码，用本地的一个time函数获取seed可能的值，然后从服务器上拖了20次猜数字的结果下来。这里我故意在recv之后再调用time。目的是为了让本地的时间数字上大于服务器端的。这样算法比较简单。

把刚才的Seed，和list粘贴到这里。

```
flag=0
x=1
while(flag==0):
    x += 1
    random.seed(v-x)
    print (x)
    i=0
    while(i<20):
        point = random.randint(1, 6)
        if point<=3:
            if list[i] == 1:
                break
        else:
            if list[i] == 0:
                break
        i += 1
    else:
        print ("boom",x)
        flag = 1
```

这段代码模拟了服务器上的环境，用暴力枚举的方法把seed给爆破了出来。大概是5W左右。。。
坑得的1e5。每次同步修正的数值都不一样，波动都在100以上。在这里卡住了。

后来行为来回的粘贴挺麻烦的，所以我把两段代码给放在一起，然后再得到样本的同时直接跑出修正值来看看。

然后，我为我的智商感到担忧。。。既然可以直接跑出seed的值了。我干啥还要断开连接呢？直接继续跑就可以了
(╯‵□′)╯ ┻━┻

然后，就有了下面这段代码。

```
#!/bin/env python
#coding:utf-8

import socket
import time
import random
list = []
right_times = 0
a=0
str=""
s = socket.socket()
s.connect(('119.254.101.232',9999))

print (s.recv(100))
s.send(b'Sure!\n')
print (s.recv(100))

x=50000
v=int(time.time() * 1e5-x)
random.seed(v)
i=0
right = 0
while i<20:
    point = random.randint(1, 6)
    s.send(b"big\n")
    h=s.recv(100)
    print (h)
    if h[0] == 67:
```

```
            list = list + [1]
            right_times += 1
        else:
            list = list + [0]
        i+=1


    print(v)
    flag=0
    x=1
    while(flag==0):
        x += 1
        random.seed(v-x)
        print (x)
        i=0
        while(i<20):
            point = random.randint(1, 6)
            if point<=3:
                if list[i] == 1:
                    break
            else:
                if list[i] == 0:
                    break
            i += 1
        else:
            print ("boom",x)
            flag = 1

    v -=x
    random.seed(v)
    i=0
    while i<20:
        point = random.randint(1, 6)
        i+=1

    i=0
    k= right_times//4+1
    while i<230+k+right_times:
        point = random.randint(1, 6)
        s.send(b"aaa\n")
        h=s.recv(100)
        print (h)
        i+=1
    q=0
    while right_times<600:
        if q ==4:
            point = random.randint(1, 6)
            s.send(b"aaa\n")
            h=s.recv(100)
            print (h)
            q=0

        point = random.randint(1, 6)
        if point > 3:
            s.send(b"big\n")
        else:
            s.send(b"small\n")
        h=s.recv(100)
        print (h)
        q += 1
```

```
            right_times +=1
        s.send(b"Me too! <3\n")
        h=s.recv(100)
        print (h)

        s.close()
```

总之，思想就是先在线取样本，在本地跑seed，然后总共构造出赢600次，输400次。注意不要半路被踢和random次数始终和服务器保持一致就可以了。最后就是服务器有时间限制，如果超时直接断开连接。所以注意跑seed的时候要稍微估算一下防止超时。

PS：靠着我那刚学1星期的python写出的半吊子代码，就不要吐槽了ヽ(￣▽￣)ノ
祝贺协会在这次ctf中取得好成绩。