【Writeup】i春秋网络安全领域专项技能赛_Reverse_srcleak

原创

BAKUMANSEC 于 2019-08-17 17:19:55 发布 1153 收藏 1
分类专栏： ichunqiu - Writeups 文章标签： Writeup Reverse

【Writeup】i春秋网络安全领域专项技能赛_Reverse_srcleak

本题给了一个cpp文件，其内容如下：

```
#include<iostream>
using namespace std;

typedef unsigned int uint;



template <bool Flag, class MaybeA, class MaybeB> class IfElse;

template <class MaybeA, class MaybeB>
class IfElse<true, MaybeA, MaybeB> {
public:
 using ResultType = MaybeA;
};

template <class MaybeA, class MaybeB>
class IfElse<false, MaybeA, MaybeB> {
public:
 using ResultType = MaybeB;
};

template <uint N, uint L, uint R> struct func1 {
 enum { mid = (L + R + 1) / 2 };

 using ResultType = typename IfElse<(N < mid * mid),
  func1<N, L, mid - 1>, func1<N, mid, R> >::ResultType;

 enum { result = ResultType::result };
};

template <uint N, uint L> struct func1<N, L, L> { enum { result = L }; };

template <uint N> struct _func1 { enum { result = func1<N, 1, N>::result }; };
```

```cpp
template<size_t Input>
constexpr size_t func2 = (Input % 2) + func2< (Input / 2) >;

template<>
constexpr size_t func2<0> = 0;
```

```cpp
template<size_t num>
constexpr size_t func3 = num % 2;
```

```cpp
template<uint n, uint m>struct NEXTN {
 const static uint value = ((n % m != 0) * n);
};
template<uint n, uint m>struct NEXTM {
 const static uint value = (m * m <= n ? (m + 1) : 0);
};
template<uint n, uint m>struct TEST {
 const static uint value = TEST<NEXTN<n, m>::value, NEXTM<n, m>::value>::value;
};
template<uint m>struct TEST<0, m> {
 const static uint value = 0;
};
template<uint n>struct TEST<n, 0> {
 const static uint value = 1;
};
template<uint n>struct func4 {
 const static uint value = TEST<n, 2>::value;
};
template<>struct func4<1> {
 const static uint value = 0;
};
template<>struct func4<2> {
 const static uint value = 1;
};
```

```
int main(int argc, char**argv) {
 //input 5 uint numbers ,x1,x2,x3,x4,x5
 //the sum of them should be MIN


 cout << func3< func2<x1> > << endl;
 cout << func3< func2<x2> > << endl;
 cout << func3< func2<x3> > << endl;
 cout << func3< func2<x4> > << endl;
 cout << func3< func2<x5> > << endl;

 // output: 1 1 1 1 1


 cout << _func1<x1>::result << endl;
 cout << _func1<x2>::result << endl;
 cout << _func1<x3>::result << endl;
 cout << _func1<x4>::result << endl;
 cout << _func1<x5>::result << endl;

 //output: 963 4396 6666 1999 3141




 //how many "1" will func4<1>,func4<2>,fun4<3>......fun4<10000> ::value  return?
 x6 = count;


 // your flag is flag{x1-x2-x3-x4-x5-x6}
 // if x1=1,x2=2,x3=3,x4=4,x5=5,x6=6
 // flag is     flag{1-2-3-4-5-6}




 return 0;
}
```

要求比较明确：读懂源码按条件计算得到x1-x6并以-连接即为flag。

笨一点的方法就是改写一下源码写出python脚本爆破出x1-x5，再计算出x6：

x1-x5：

```python
def func2(x):
 if x == 0:
  return 0
 return (x % 2) + func2(x // 2)

def func3(x):
 return x % 2

def func1(N, L, R):
 if L == R:
  return L
 mid = (L + R + 1) // 2
 if N < mid * mid:
  return func1(N, L, mid - 1)
 else:
  return func1(N, mid, R)

def _func1(x):
 return func1(x, 1, x)



if __name__ == '__main__':
 x1_flag = False
 x2_flag = False
 x3_flag = False
 x4_flag = False
 x5_flag = False
 for i in range(10000000, 100000000):
  if func3(func2(i)) != 1:
   continue
  if _func1(i) == 963 and not x1_flag:
   print("x1:",i)
   x1_flag = True
  if _func1(i) == 4396 and not x2_flag:
   print("x2:",i)
   x2_flag = True
  if _func1(i) == 6666 and not x3_flag:
   print("x3:",i)
   x3_flag = True
  if _func1(i) == 1999 and not x4_flag:
   print("x4:",i)
   x4_flag = True
  if _func1(i) == 3141 and not x5_flag:
   print("x5:",i)
   x5_flag = True
```

x6：

```
def nextm(n, m):
 if m*m <= n:
  return m+1
 else:
  return 0

def nextn(n, m):
 return (n % m != 0) * n

def test(n, m):
 if n == 0:
  return 0
 if m == 0:
  return 1
 return test(nextn(n, m), nextm(n, m))

def func4(x):
 if x == 1:
  return 0
 if x == 2:
  return 1
 return test(x, 2)


if __name__ == '__main__':
 x6 = 0
 for i in range(1, 5):
  if func4(i*2-1) == 1:
   x6 += 1
 print(x6)
```

快一点的话就是由语义和简单的测试可以得出，_func1的输入值是输出值的平方，即x1-x5可以由output的5个数分别平方获取；而x6是1-10000整数中质数的个数。

最终flag：
`flag{927369-19324816-44435556-3996001-9865881-1229}`