Untitled

BY: A GUEST ON SEP 4TH, 2015 | SYNTAX: NONE | SIZE: 4.99 KB | VIEWS: 71 | EXPIRES: NEVER

DOWNLOAD | RAW | EMBED | REPORT ABUSE | PRINT | QR CODE | CLONE


```
#!/usr/bin/env python2.7
#By aaaddress1@gmail.com

from z3 import *
b = [ BitVec('b%d' % i, 32) for i in range(16) ]
s = Solver()

for i in range(16):
        s.add(b[i] & 0xff == b[i])

#  _   _   _
# | | | | __ _ ___| |__
# | |_| |/ _` / __| '_ \
# |  _  | (_| \__ \ | | |
# |_| |_|\__,_|___/_| |_|
#

v6 = BitVec('s1', 32)
v7 = BitVec('s2', 32)
v8 = BitVec('s3', 32)
v9 = BitVec('s4', 32)
s.add(v6 == 0 and v7 == 0 and v8 == 0 and v8 == 0)

'''
v4 = 0LL;
    v6 = 0;
    v7 = 0;
    v8 = 0;
    v9 = 0;
    do
    {
      v10 = v1[v4];
      if ( (unsigned __int8)(v10 - 33) > 0x59u )
      {
        LODWORD(v4) = 0;
        return (unsigned int)v4;
      }
      v11 = __ROR4__(v8, 23);
      ++v4;
      v9 += v10;
      v7 = v10 ^ 8 * v7;
      v6 = v10 + 32 * v6;
      v8 = 37 * v10 + v11;
    }
    while ( v4 != 16 );
'''
for v4 in range(16):
        v10 = b[v4]
        s.add((0xff &(v10 - 33)) <= 0x59)
        v11 = RotateRight(v8, 23)
        v9 += v10
        v7 = v10 ^8 * v7
        v6 = v10 + 32 * v6
        v8 = 37 * v10 + v11
# if ( v9 == 1720 && v8 == 14470772 && v7 == -340847546 && v6 == -1721686258 )
s.add(v6 == 0x9961270E)
```

```
s.add(v7 == 0xEBAF1446)
s.add(v8 == 14470772)
s.add(v9 == 1720)


#   _____ _ _   _
#  / ___|___  _ __   __| (_) |_(_) ___  _ __
# | |   / _ \| '_ \ / _` | | __| |/ _ \| '_ \
# | |__| (_) | | | | (_| | | |_| | (_) | | | |
#  \____\___/|_| |_|\__,_|_|\__|_|\___/|_| |_|
#
v = [0] * 16
for i in range(16):
        v[i] = b[i]
#*(_DWORD *)v1 ^= 0x11111111u;
for i in range(0,4):
        v[i] = v[i] ^ 0x11;
#*((_DWORD *)v1 + 1) ^= 0xAAAAAAAA;
for i in range(4,8):
        v[i] = v[i] ^ 0xAA
#*((_DWORD *)v1 + 2) ^= 0x55555555u;
for i in range(8,12):
        v[i] = v[i] ^ 0x55
#*((_DWORD *)v1 + 3) ^= 0x33333333u;
for i in range(12,16):
        v[i] = v[i] ^ 0x33
#v12 = v1[1] ^ *v1;
v12 = (0xff & v[1]) ^ v[0]
#v13 = (unsigned __int8)(v1[2] ^ v12);
v13 = 0xff & (v[2] ^ v12)
#v14 = (unsigned __int8)(v1[3] ^ v1[2] ^ v12);
v14 = 0xff & (v[3] ^ v[2] ^ v12)
#v15 = (unsigned __int8)(v1[4] ^ v1[3] ^ v1[2] ^ v12);
v15 = 0xff & (v[4] ^ v[3] ^ v[2] ^ v12)
#v16 = (unsigned __int8)(v1[5] ^ v1[4] ^ v1[3] ^ v1[2] ^ v12);
v16 = 0xff & (v[5] ^ v[4] ^ v[3] ^ v[2] ^ v12)
#v17 = (unsigned __int8)(v1[6] ^ v1[5] ^ v1[4] ^ v1[3] ^ v1[2] ^ v12);
v17 = 0xff & (v[6] ^ v[5] ^ v[4] ^ v[3] ^ v[2] ^ v12)
#v18 = v1[8] ^ v1[7] ^ v1[6] ^ v1[5] ^ v1[4] ^ v1[3] ^ v1[2] ^ v12;
v18  = (v[8] ^ v[7] ^ v[6] ^ v[5] ^ v[4] ^ v[3] ^ v[2] ^ v12)
#v19 = (unsigned __int8)(v1[7] ^ v1[6] ^ v1[5] ^ v1[4] ^ v1[3] ^ v1[2] ^ v12);
v19 = 0xff & (v[7] ^ v[6] ^ v[5] ^ v[4] ^ v[3] ^ v[2] ^ v12)
#*((_BYTE *)v1 + 8) = v18;
v[8] = v18
#v20 = v1[9] ^ v18;
v20 = v[9] ^ v18
#v21 = (unsigned __int8)(v1[10] ^ v20);
v21 = 0xff & (v[10] ^ v20)
#v22 = (unsigned __int8)(v1[11] ^ v1[10] ^ v20);
v22 = 0xff & (v[11] ^ v[10] ^ v20)
#v23 = (unsigned __int8)(v1[12] ^ v1[11] ^ v1[10] ^ v20);
v23 = 0xff & (v[12] ^ v[11] ^ v[10] ^ v20)
#v24 = v1[14] ^ v1[13] ^ v1[12] ^ v1[11] ^ v1[10] ^ v20;
v24 = v[14] ^ v[13] ^ v[12] ^ v[11] ^ v[10] ^ v20
#v25 = v1[13] ^ v1[12] ^ v1[11] ^ v1[10] ^ v20;
v25 = v[13] ^ v[12] ^ v[11] ^ v[10] ^ v20
#*((_BYTE *)v1 + 15) ^= v24;
v[15] ^= v24
#*v1 ^= 0x63u;
v[0] ^= 0x63
#*((_BYTE *)v1 + 8) ^= 0x30u;
v[8] = 0x30
#*((_BYTE *)v1 + 1) = (2 * v12 | ((signed int)v12 >> 1)) ^ 0x2F;
v[1] = (2 *v12 | v12 >> 1) ^ 0x2f
#*((_BYTE *)v1 + 2) = (4 * v13 | (v13 >> 2)) ^ 0xDC;
v[2] = (4 *v13 | v13 >> 2) ^ 0xdc
#*((_BYTE *)v1 + 3) = (8 * v14 | (v14 >> 3)) ^ 0x20;
v[3] = (8 *v14 | v14 >> 3) ^ 0x20
#*((_BYTE *)v1 + 4) = (16 * v15 | (v15 >> 4)) ^ 0xCD;
v[4] = (16*v15 | v15 >> 4) ^ 0xcd
#*((_BYTE *)v1 + 5) = (32 * v16 | (v16 >> 5)) ^ 0xA0;
v[5] = (32*v16 | v16 >> 5) ^ 0xa0
#*((_BYTE *)v1 + 6) = (((_BYTE)v17 << 6) | (v17 >> 6)) ^ 0x83;
v[6] = ( (0xff & v17) << 6 | v17 >> 6) ^ 0x83
```

```
#*((_BYTE *)v1 + 7) = ((_BYTE)v19 << 7) | (v19 >> 7);
v[7] = ( (0xff & v19) << 7 | v19 >> 7)
#*((_BYTE *)v1 + 9) = (2 * v20 | ((signed int)v20 >> 1)) ^ 0x7D;
v[9] = (2 * v20 | v20 >> 1) ^ 0x7d
#*((_BYTE *)v1 + 10) = (4 * v21 | (v21 >> 2)) ^ 0x19;
v[10] = (4 * v21 | v21 >> 2) ^ 0x19
#*((_BYTE *)v1 + 11) = (8 * v22 | (v22 >> 3)) ^ 4;
v[11] = (8 * v22 | v22 >> 3) ^ 4
#*((_BYTE *)v1 + 12) = (16 * v23 | (v23 >> 4)) ^ 0xC4;
v[12] = (16 * v23 | v23 >> 4) ^ 0xc4
#*((_BYTE *)v1 + 13) = (32 * v25 | ((signed int)v25 >> 5)) ^ 0x20;
v[13] = (32 * v25 | v25 >> 5) ^ 0x20
#LODWORD(v4) = v1[15];
v4 = v[15]
#*((_BYTE *)v1 + 14) = ((v24 << 6) | ((signed int)v24 >> 6)) ^ 0xC1;
v[14] = (v24 << 6 | v24 >> 6) ^ 0xc1
#*((_BYTE *)v1 + 15) = ((_BYTE)v4 << 7) | ((signed int)v4 >> 7);
v[15] = (v4 << 7 | v4 >> 7)
#LODWORD(v4) = (*((_DWORD *)v1 + 3) | *((_DWORD *)v1 + 2) | *(_DWORD *)v1 | *((_DWORD *)v1 + 1)) == 0;
s.add(v[0] & 0xff == 0)
s.add(v[1] & 0xff == 0)
s.add(v[2] & 0xff == 0)
s.add(v[3] & 0xff == 0)
s.add(v[4] & 0xff == 0)
s.add(v[5] & 0xff == 0)
s.add(v[6] & 0xff == 0)
print(s.check())

m = s.model()
res = ""
for i in range(16):
        v = int(str(m[b[i]]))
        c = chr(v)
        print('%d: %d %s' % (i, v, c))
        res += c

print("Key is %s" % res)
```

RAW Paste Data clone this paste

```
#!/usr/bin/env python2.7
#By aaaddress1@gmail.com

from z3 import *
b = [ BitVec('b%d' % i, 32) for i in range(16) ]
s = Solver()

for i in range(16):
        s.add(b[i] & 0xff == b[i])

#  _   _   _
# | | | | __ _ ___| |__
```