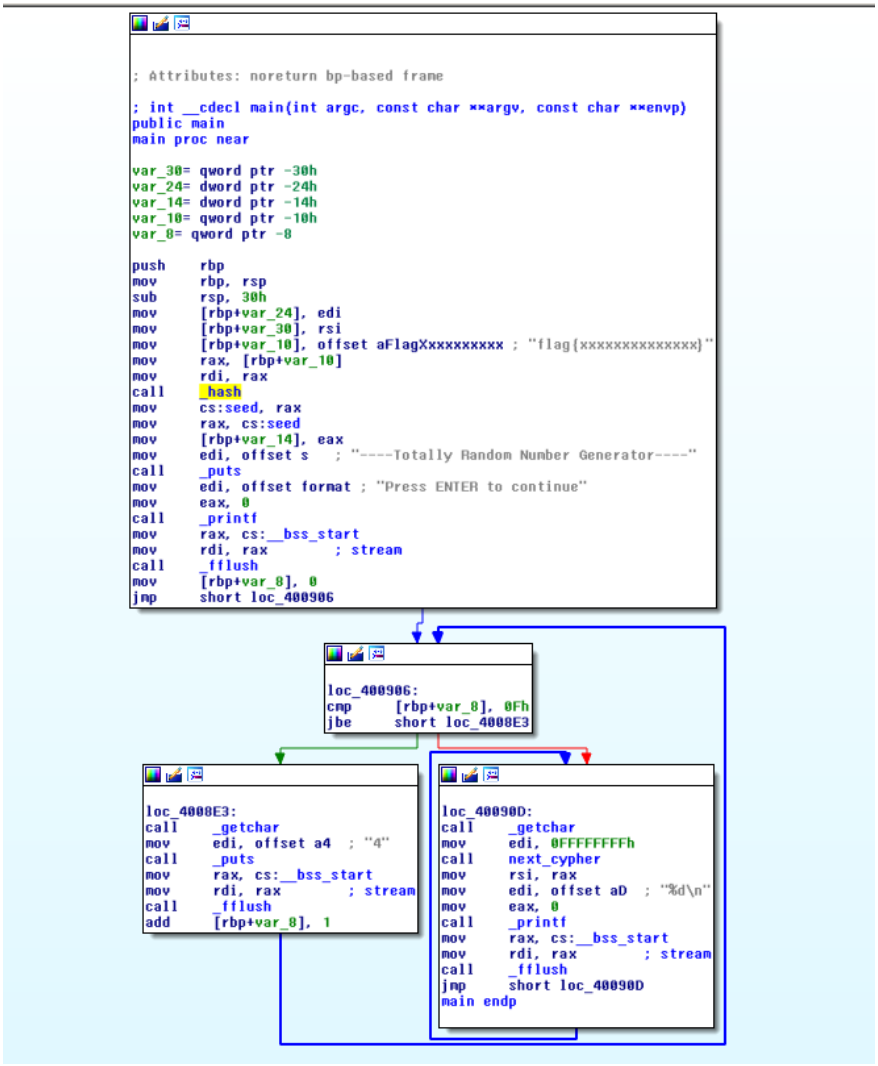
CSAW CTF 2016 Finals, Katy writeup

Katy was a crypto 50 question in CSAW. We're given a 64-bit standard ELF binary. Download from here.

```
aneesh@aneesh-ubuntu:~/csaw_finals$ file release
release: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked
, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=6
75399f73a52ff88383a475ad8ffba9aed65bd71, not stripped
aneesh@aneesh-ubuntu:~/csaw_finals$ strings ./release  | grep flag
flag{xxxxxxxxxxxxx}
```

katy binary

Lets see what IDA has to say:



Katy in IDA pro 64 bit

It passes the redacted flag through a custom subroutine _hash. It uses the output to initialize the seed. Then it prints “4”, oxF times and then prints the output from next_cypher.

_hash:

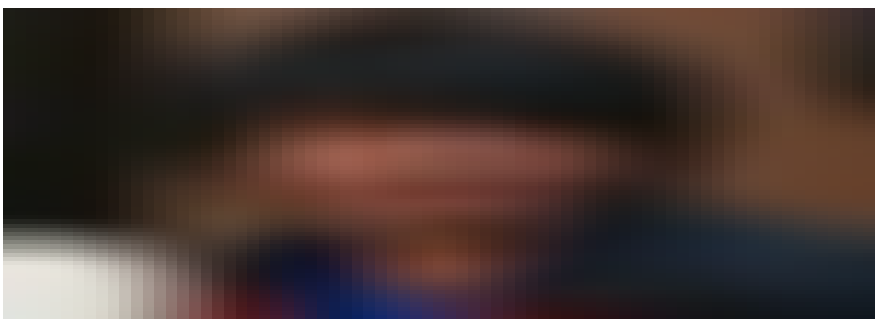
Looking at the assembly of float multiplication is a bit too scary for me, but IDA decompiles it so easily:

It seems like the first condition in the for loop will never meet and then it resorts to a simple hash function (2*input[0] + 4*input[1] + 8*input[2] …).

next_cypher:

katy's next_cypher subroutine

The next_cypher uses the existing value of the seed and changes it by applying some simple operations. We can easily recover the initial seed value from the first random number the program outputs. I wrote a tiny z3 script for this.



katy solution in z3

But recovering the flag from the seed value seems impossible. Later they updated the question and asked us to submit the initial seed

value. So that’s all it took :D