LOGIC & BINARIES

Thais Moreira Hamasaki aka barbieAuglend

2018-10-03 | Virus Bulletin 2018 | Montreal, CA

DISCLAIMER

This research was accomplished by me in my personal capacity during my spare time.

ABOUT ME

```
echo 'Stare at binaries during the day @ F-Secure';

echo 'Blackhoodie - Organizer and Board Member';
echo 'HackLu program`s committee';
echo 'Disobey`s Lead of Technical Content';
echo 'x86 Assembly & RE101 - Lead of both groups @chaosdorf';
echo 'Logical Programming, RE, static analysis, Mountaineering FTW';

echo 'Stare at binaries by night';
```

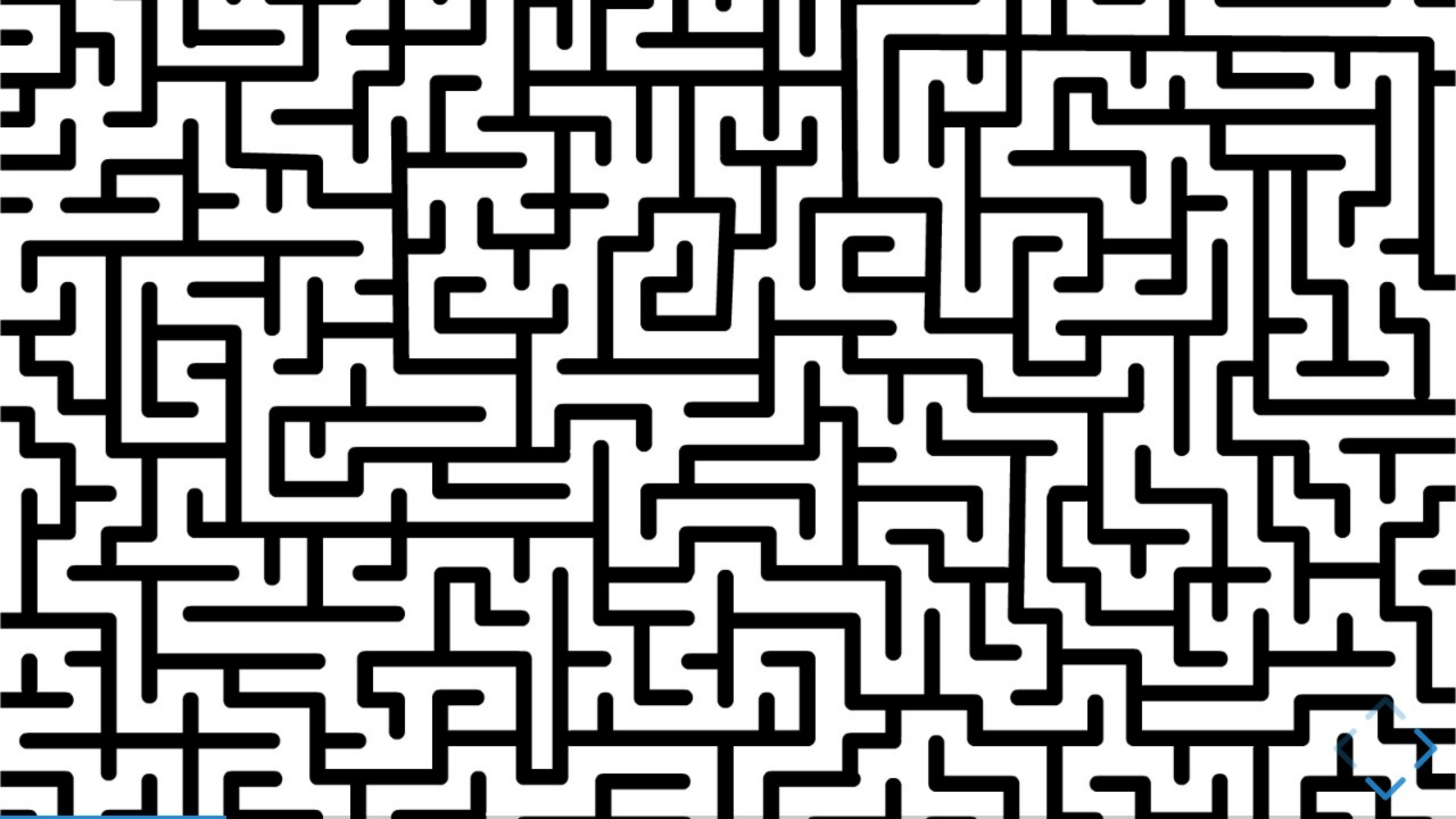
WHAT AM I GOING TO TALK ABOUT?

- constraint logic programming (CLP)
- solvers
- malware RE challenges
- Logic vs. Malware

S..A..M.. WHAT?

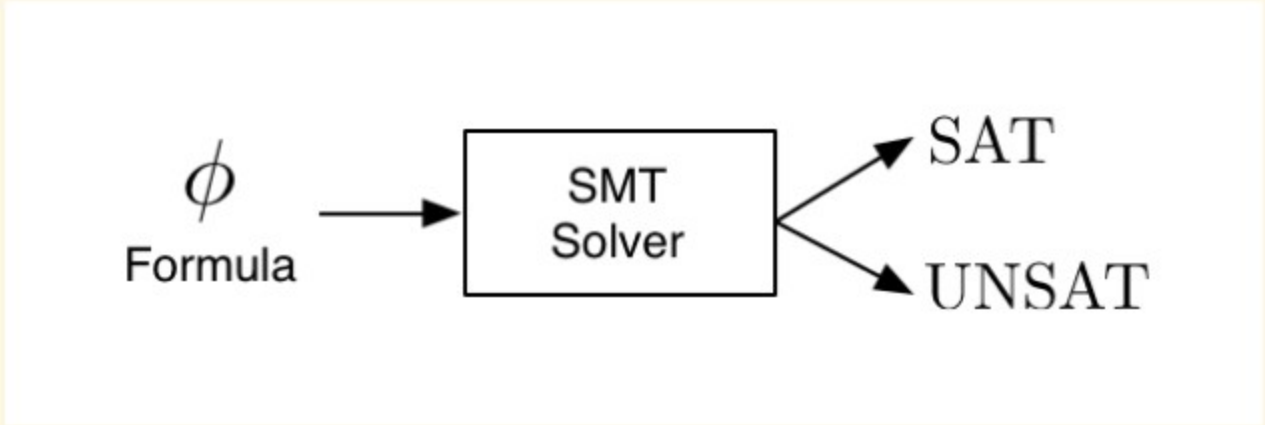
- Solver!
  - Satisfiability Modulo Theories (SMT)

CONSTRAINTS



"Constraint programming represents one of the closest approaches computer science has yet made to the Holy Grail of programming: the user states the problem, the computer solves it." Eugene C. Freuder, Constraints, April 1997

AUTOMATED THEOREM PROVING



- Hardware and Software → Large-scale verification
- Languages specification and Computing proof obligations

SYMBOLIC EXECUTION

IT LOOKS LIKE THAT ...

```
1   x = input();
2   x = x + 7;
3
4   if (x > 0)
5   y = input();
6   else
7   y = 11;
8
9   if (x > 2)
10  if (y == 42)
11  throw
      Exception()
```

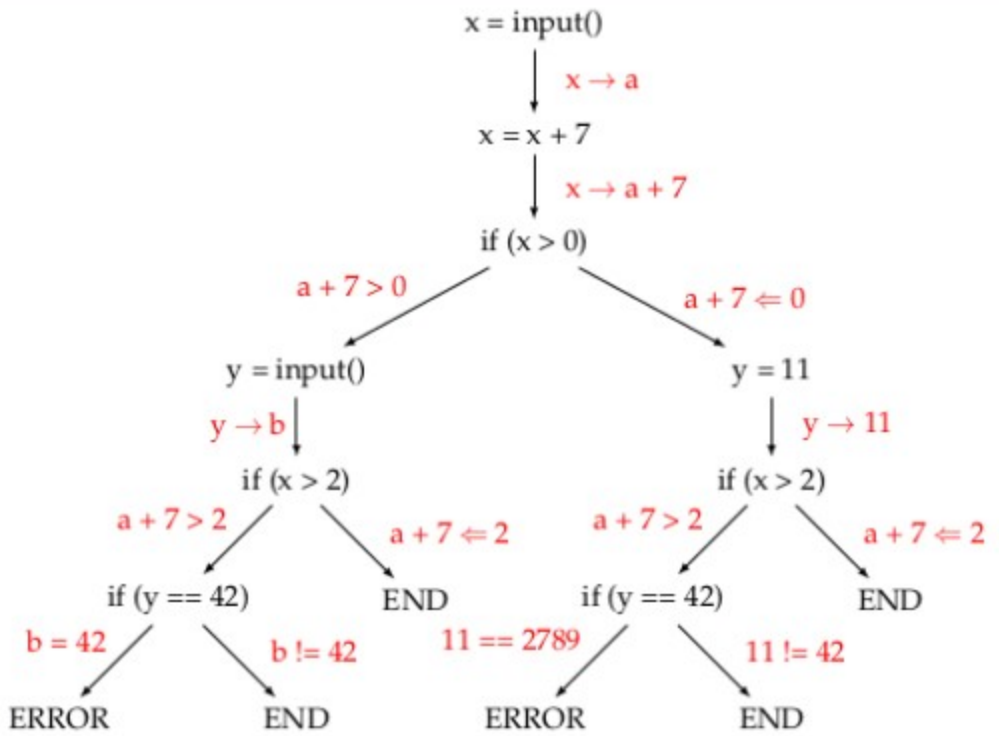


Figure 8: *Example of symbolic execution for simple program*

HOW IT WORKS

- Create a process (pc = 0, state = [])
- Add the process (pc, state) to the domain system D
- while D not empty:
  - Remove process (pc, state) from system
  - Execute it until the next branching point
    - If both paths are feasible, add both to D
    - if just one is feasible, add the feasible path and the negation of the not feasible path to D

KEEP IN MIND!

- Symbols as arguments
  - any feasible path
  - all Program states
- Symbolic values also for memory allocations
- Path conditions

APPLICATIONS

BUG HUNTING

- Fuzzing
- Code verification
- Binary Analysis

EXPLOITATION

- PoC
- AEG
- APG

MALWARE ANALYSIS

- Obfuscation
- Compiler optimizations
- Crypto-analysis

BINARY OBFUSCATION

TWO SIDES OF THE SAME COIN

- Malware obfuscation
- Software property protection

MALWARE DEOBFUSCATION

WHAT CAN POSSIBLY GO WRONG?

Compiler optimization

Packing

Various obfuscation techniques

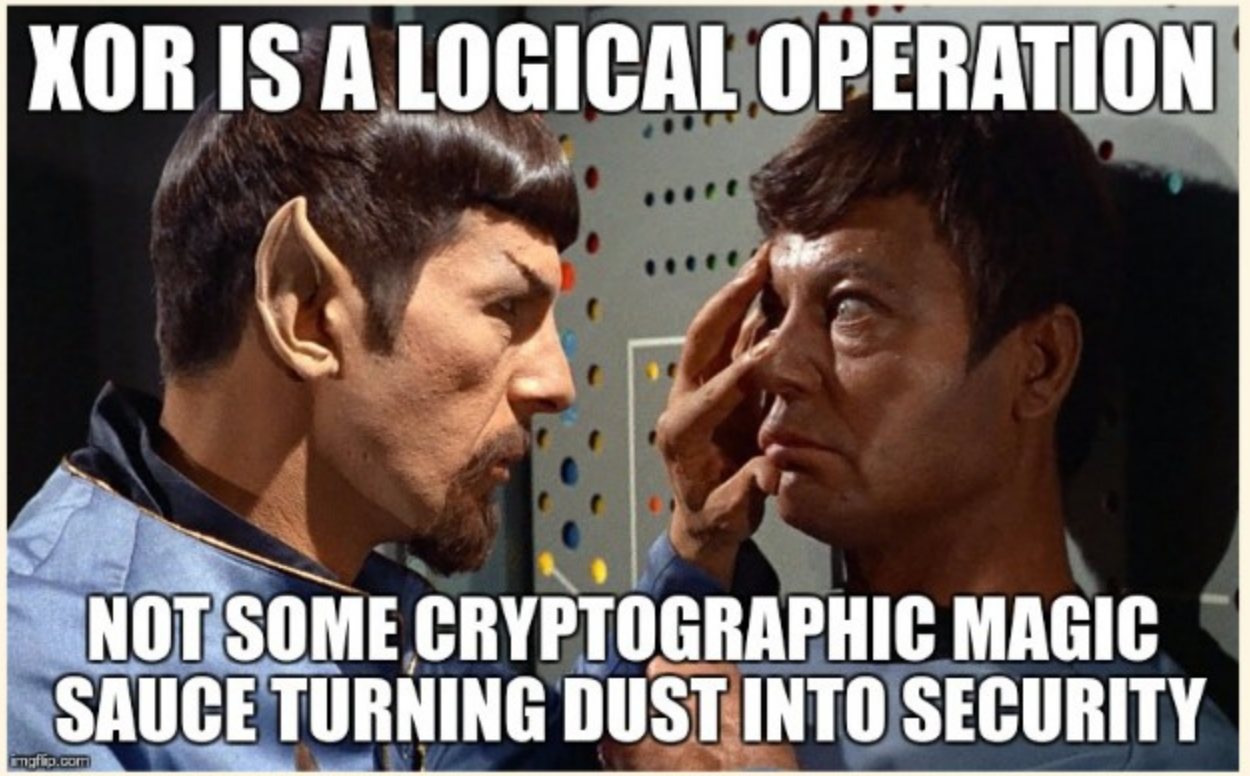
OBFUSCATION TECHNIQUES

Garbage code

Unnecessary instructions

Opaque predicates

OBFUSCATION TECHNIQUES



FOR PACKERS' SAKE

UPX

NSIS

Self implemented

IMPLEMENTATION

CONTROL FLOW

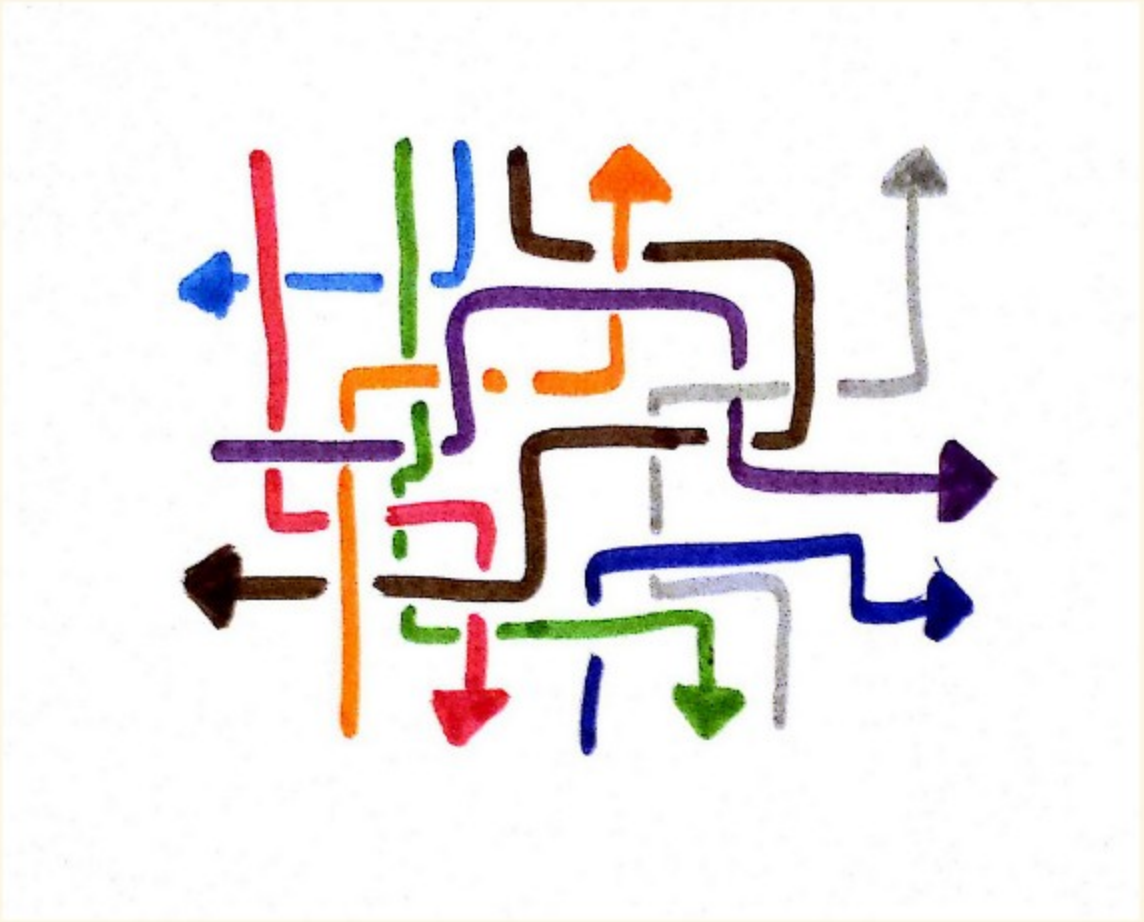
All the J* instructions

DATA FLOW

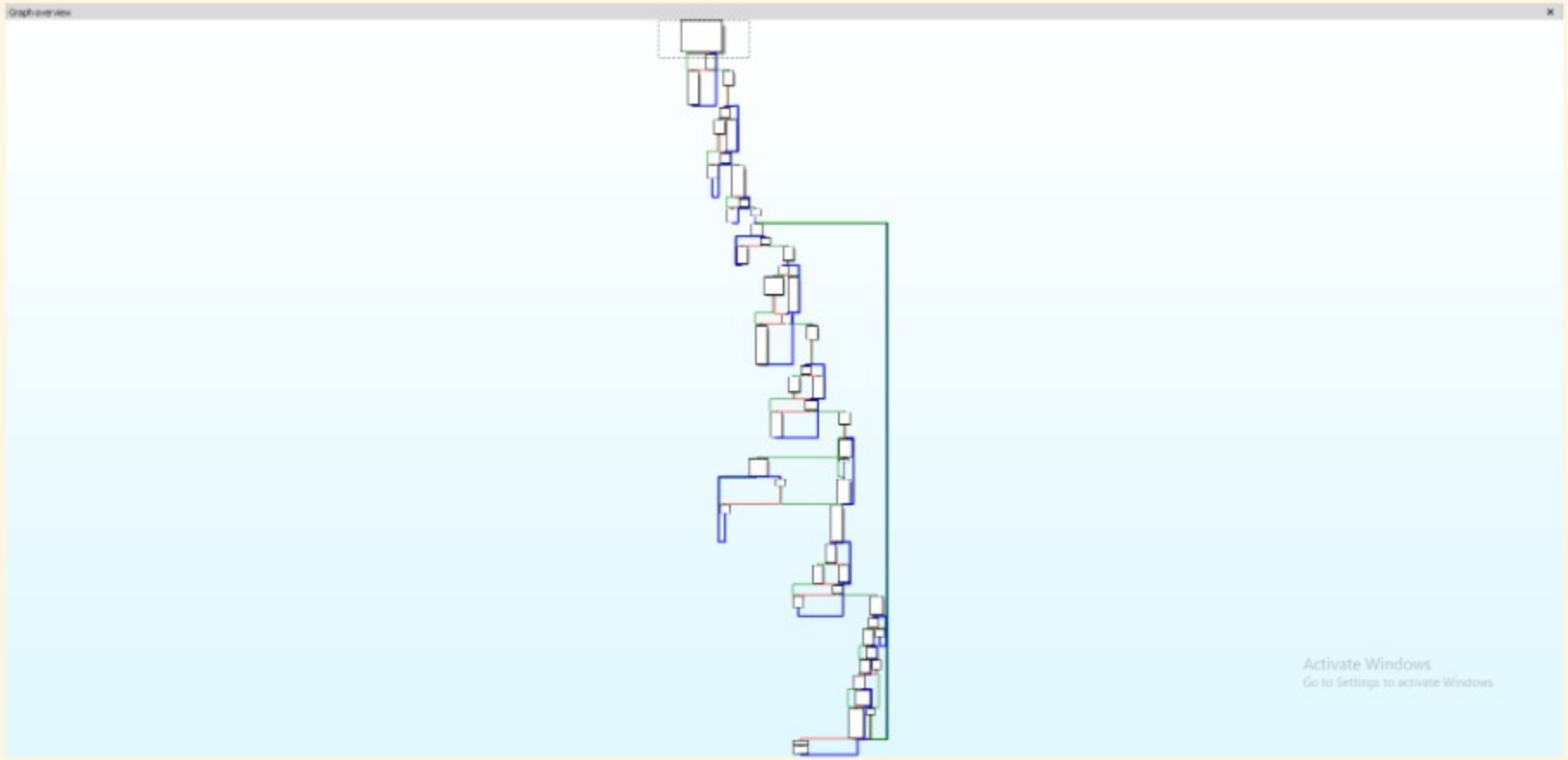
MOV, ADD, SUB, MUL, DIV

OR, AND, XOR, CMP, TEST

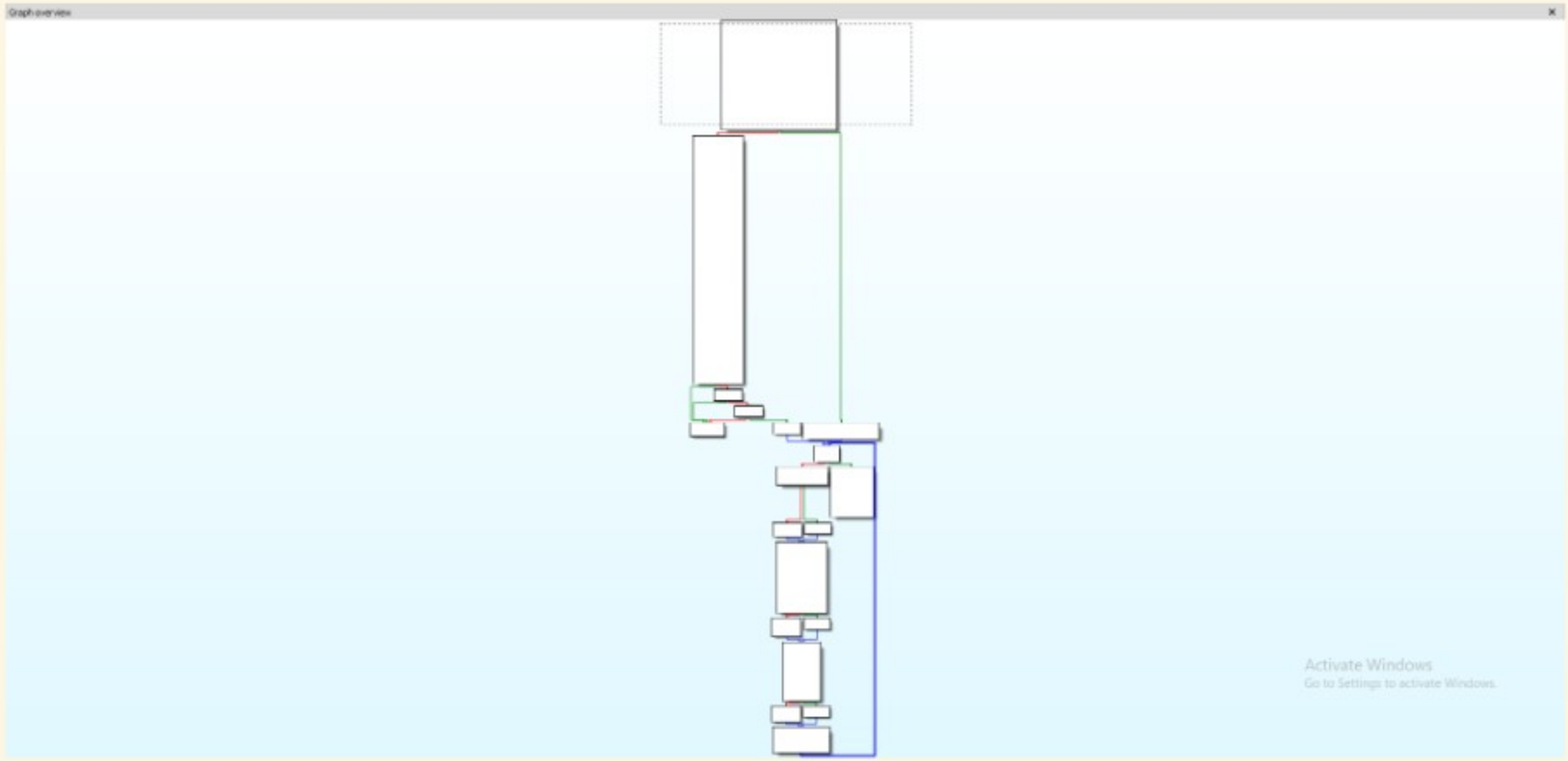
API CALLS



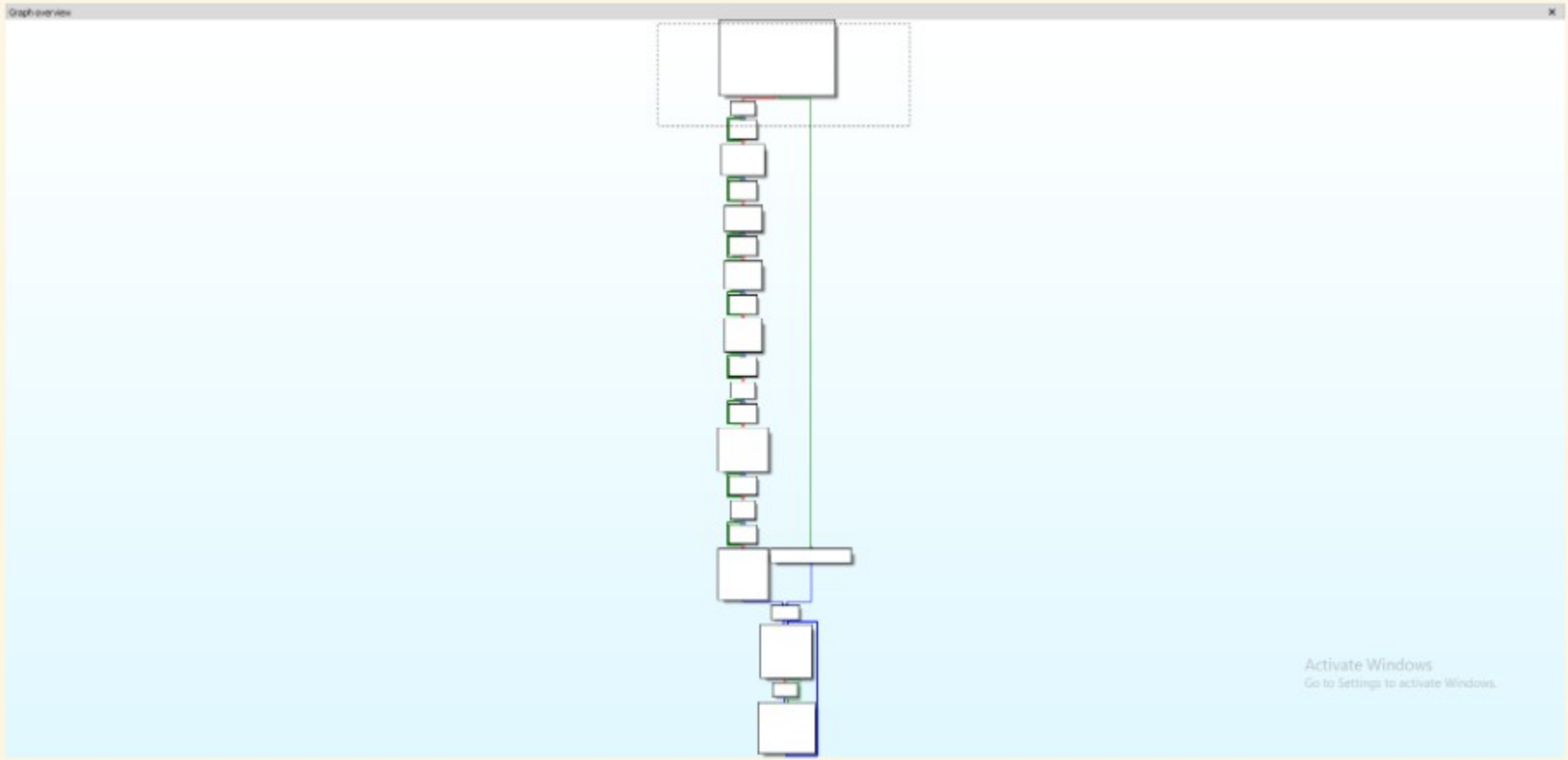
RESULTS - DEMO



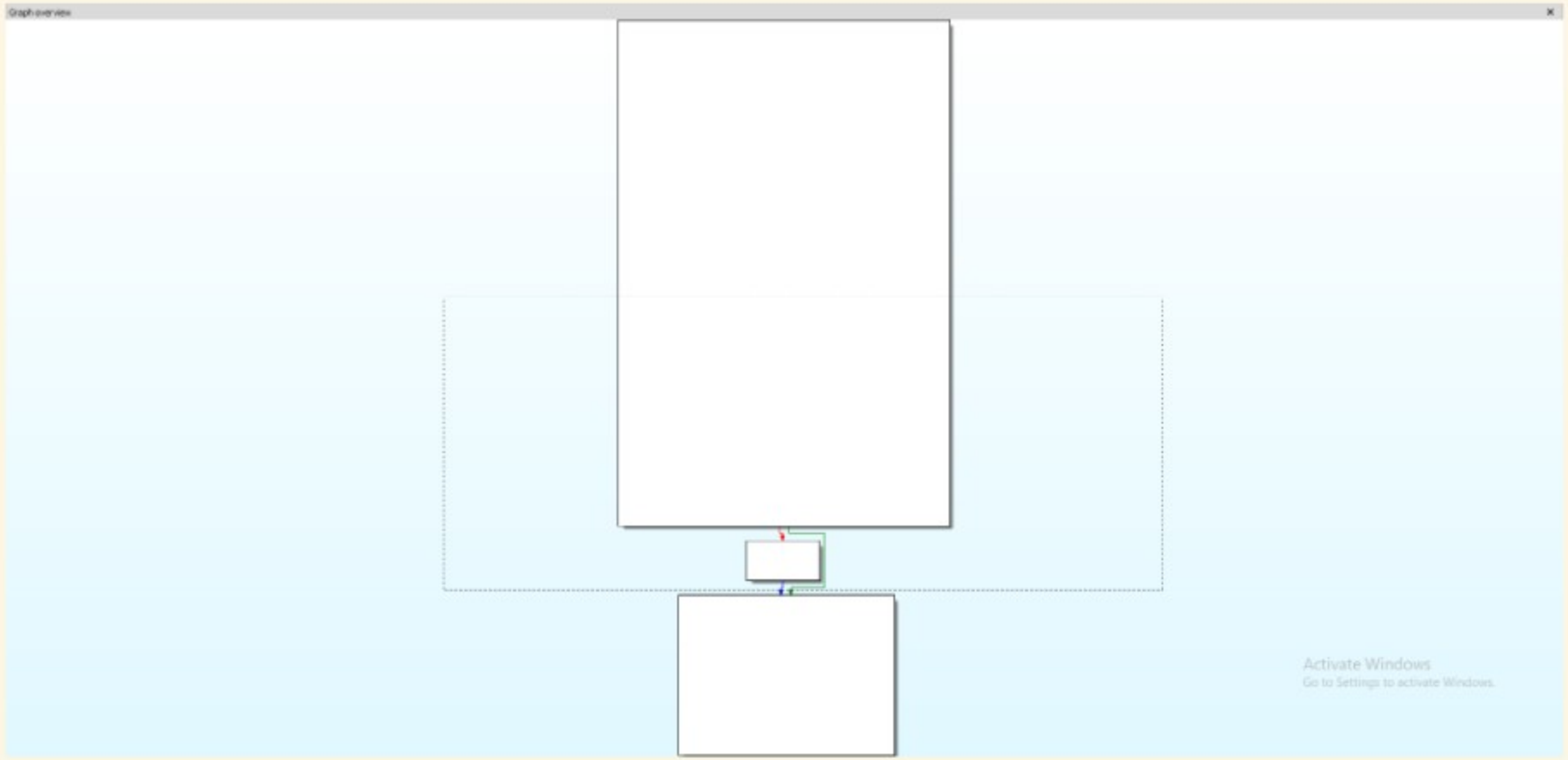
RESULTS - DEMO



RESULTS - DEMO



RESULTS - DEMO



LIMITATIONS

THEORETICAL

Rice's Theorem

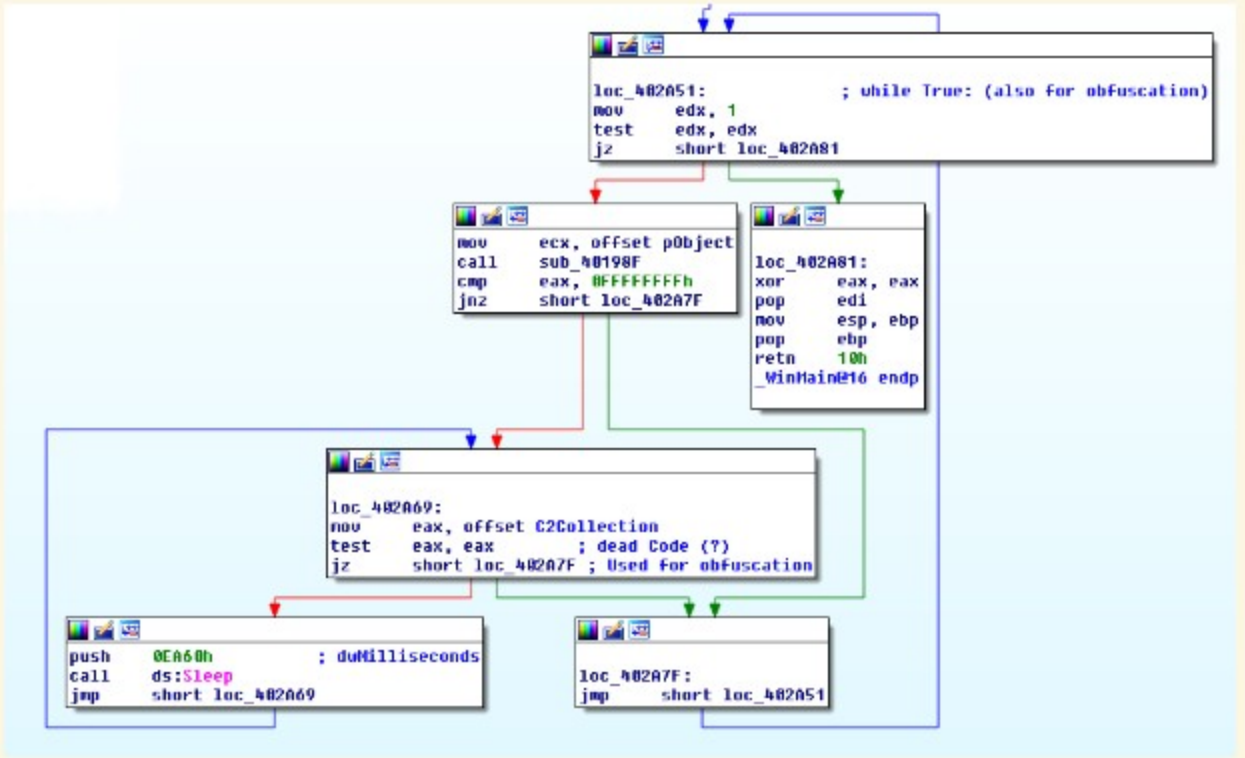
Theorem

Let L be a subset of Strings representing Turing machines, where

1. If M_1 and M_2 recognize the same language, then either $\langle M_1 \rangle, \langle M_2 \rangle \in L$ or $\langle M_1 \rangle, \langle M_2 \rangle \notin L$.
2. $\exists\ M_1, M_2$ s.t $\langle M_1 \rangle \in L$ and $\langle M_2 \rangle \notin L$.

Then L is undecidable.

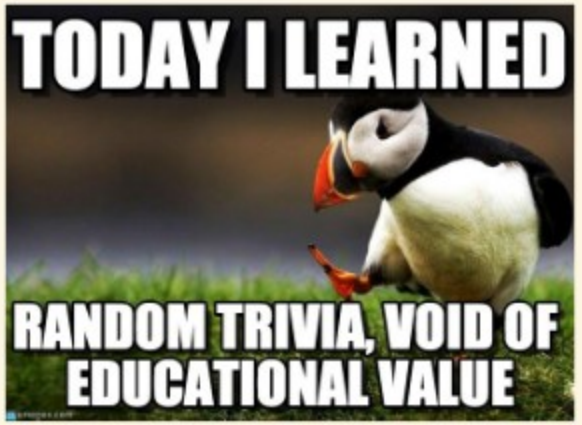
PRACTICAL



Remember...

CONCLUSION

LEARNINGS / TAKE AWAY



- Symbolic execution is a powerful tool while analysing malware
- SMT solvers can be used to simplify CFG and support analysts while reversing

WORK DONE:

- a binary garbage-code eliminator, a XOR search, some "cryptographic" algorithm breaker, a generic unpacker, a binary structure recognizer, a C++ class hierarchy reconstructor.

WORKING ON ...

- r2 integration,
- maybe IDA-Plugin.

QUESTIONS?

barbieAuglend

barbieAuglend@chaosdorf.de

More information and sources can be found in the VB paper!