Understanding how Mergen deobfuscates VM-based obfuscations

YUSUF "NACI" İŞLEK

The goal

- Applying compiler optimizations to obfuscated code.
- Creating a fast and generic solution for lifting binaries.

What is Mergen

- Mergen is a symbolic execution engine and a lifter to LLVM IR.
- Designed for deobfuscating virtual-machine based obfuscations.
- It’s built on LLVM and Zydis (a disassembler library).
- We analyze the values, pre-optimizations, pattern matching, KnownBits analysis.
- If it still fails, optionally we can try Z3 solver.

Output Quality

Let’s see how Mergen performs on target binaries.

Our targets:

1- simple target binary

2- flattened code

3- example vm-obfuscated code

4- flattened code protected by VMP

It’s fast

flattened_code.vmp.exe (linear version)

Triton : 32 seconds (87s SE+opt)

Mergen : 891.569ms (1612.96ms SE + opt)

Why Mergen is faster?

Why Mergen is faster?

- Proving is slow.

Instead:

- Find a way to re-use simplification rules like KnownBits, pattern matching etc.

- Once you prove an expression, you dont have to prove again, you can re-use it.

Improvements

- It’s not perfect, hand made instruction semantics are not 100% accurate.
- It will not solve all the MBA’s (Mixed Boolean Arithmetics). It tries to solve cfg and run usual passes.
- Some MBA’s can confuse Mergen.

Future ideas

- Lazy eval; we dont have to calculate everything.
- Better pattern matching; dump expressions and rewrite them so its easy to integrate.

Alternative usage ideas?

- Sanitizer
- Fuzzer
- Obfuscation

Conclusion

- You can use LLVM as a symbolic execution engine without too much relying on a solver.
- LLVM is able to optimize commercial VM-based obfuscations.
- Proving is slow, but once you prove an expression, you can generalize it and pattern match it.