Fracture: Inverting LLVM’s Target Independent Code Generator

Technical Approach

- We mirrored the LLVM compilation process
  - This was sometimes easier, sometimes harder
    - Easy: Objects, graphs, etc already done
    - Hard: fighting LLVM API, decoding symbols/memory accesses, classic decompilation problems (which we haven’t solved yet)

- Two major software components:
  - A “reverse” DAG manipulation table in TableGen
  - Libraries to handle LLVM API interaction (“CodeInv”)

Ideas to Use Fracture

- Multiple code “views” (e.g., IDA Pro plugin) to see IR->MC when user doesn’t know Target
- Use KLEE to analyze code
  - Lots of neat recent research could be applied here
  - Can use to solve indirect control transfer (ICT)
- Interpreter (II) can run code
  - Extend to simulate entire embedded system if you can work out HW interaction
  - Binary editor - retarget (for tractable programs), fix bugs, change functionality
- Decompiler (i.e., C-Backend)

Summary

- Modular (and fast?) conversion from machine code to IR
- Uses existing LLVM APIs and TableGen definitions
- Generic IR-based tools instead of one-off target specific solutions

DISCLAIMER:

The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

This research was sponsored by the Air Force Research Laboratory (AFRL), contract number FA8750-12-C-0261, as part of the DARPA High Assurance Cyber Military Systems (HACMS) program.