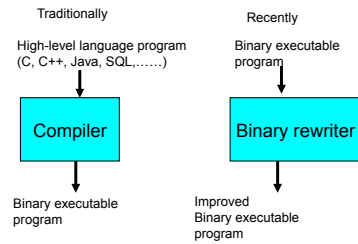


# SecondWrite: Better than first

Kapil Anand, Timothy Creech, Nathan Giles, Jim Gruen, Aparna Kotha, Padraig O'Sullivan, Matt Smithson, Pape Sylla, Khaled Wazeer, Greeshma Yellareddy, Rajeev Barua



## What is a Binary Rewriter



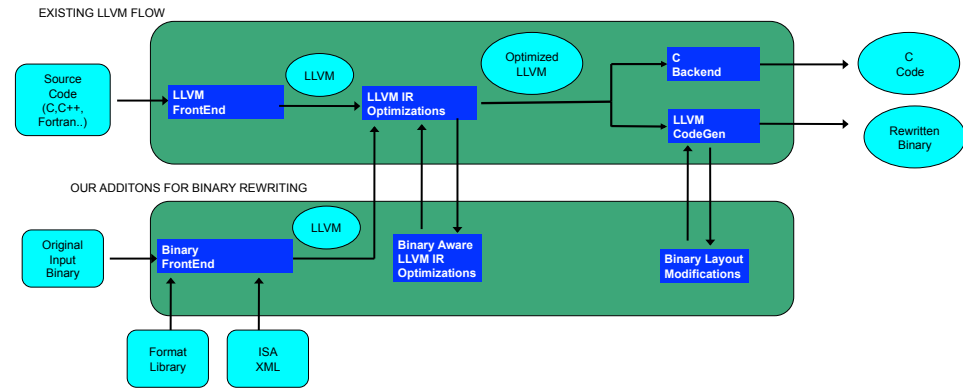
## Advantages of Binary Rewriter

- Whole program view allows interprocedural optimization
  - Not possible at compiler level
- Allows optimizations missed by compiler.
- Economically feasible
  - Portable across any language, compiler or ISA.
- Applicable to legacy codes and assembly level programs
- Security enforcement in binaries.

## Applications

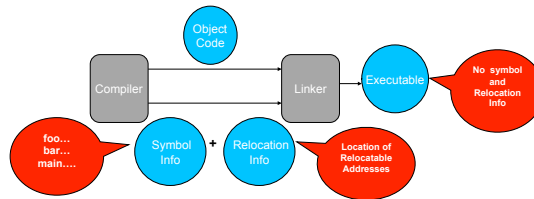
- Automatic Parallelization
  - Portable across any number of cores
- Improvement of Security and reliability
  - Protection against malicious attacks and enforcement of security policies
  - Implementing access control to data and services
- Whole program and binary specific optimizations
- Platform Specific optimizations
  - Customizing binary to underlying platforms

## Flow of Binary Rewriter



## Why is Secondwrite better than existing Binary Rewriters?

### Can Rewrite without relocation information



- Relocation information
  - Used by the linker to combine various object files into one executable binary
- Compilers discard Relocation Information
  - Not necessary for executing the binary
  - To avoid reverse engineering
- Secondwrite can rewrite all binaries, even without relocation or symbolic information
- Important since virtually all commercial binaries lack relocation information
- Security Policies enforcement not possible without the power to rewrite binaries without relocation information

### Employs Compiler Level Intermediate Format

- Secondwrite converts binaries to an existing compiler level intermediate format ( LLVM)
- Recovers high level information like function prototypes.
- Replaces register and memory accesses by symbols
- Advantages
  - Enables SecondWrite to do any code transformation since compiler IR is well known to support complex transformations.
  - Ability to apply source level automatic parallelization and security enforcement techniques without any modifications.
  - Reuse passes from mature compilers developed over multiple years of efforts.
  - Ability to employ all the existing and future compiler level research at binary level
  - Existing compiler backends can be used to convert the obtained compiler IR to source languages like C