Adding Compilation Metadata To Binaries To Make Disassembly Decidable

Daniel Engel, Freek Verbeek

Open Universiteit

Pranav Kumar, Binoy Ravindran

VIRGINIA TECH

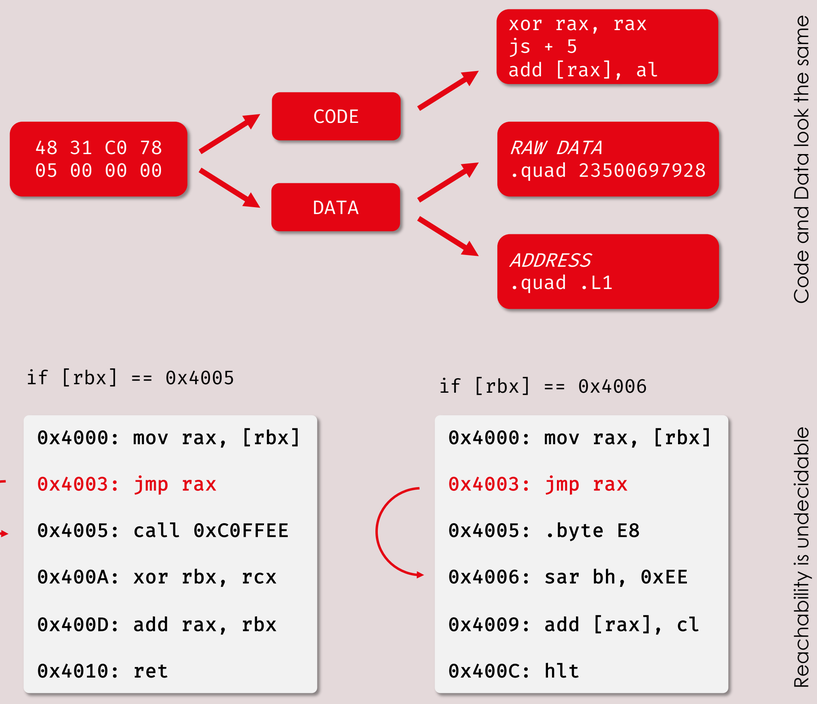
Reliable Software Decomposition - SIG

OpenSSF

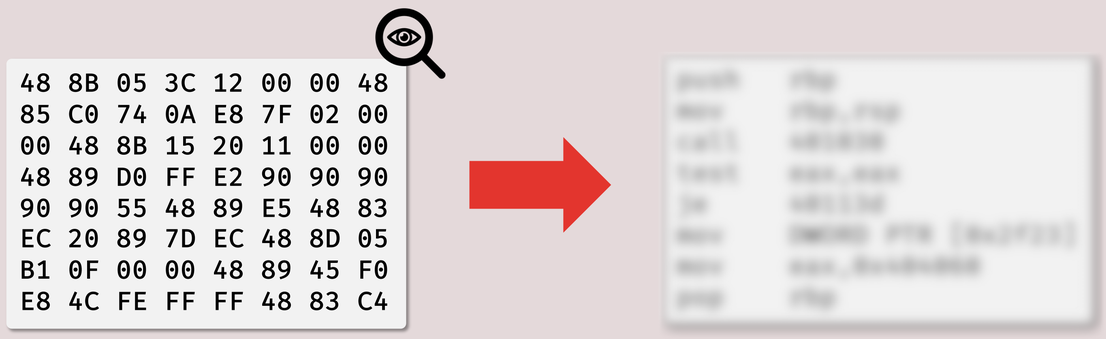
OPEN SOURCE SECURITY FOUNDATION

LLVM Dev Meeting 2026 - Dublin

LLVM COMPILER INFRASTRUCTURE



PROBLEM

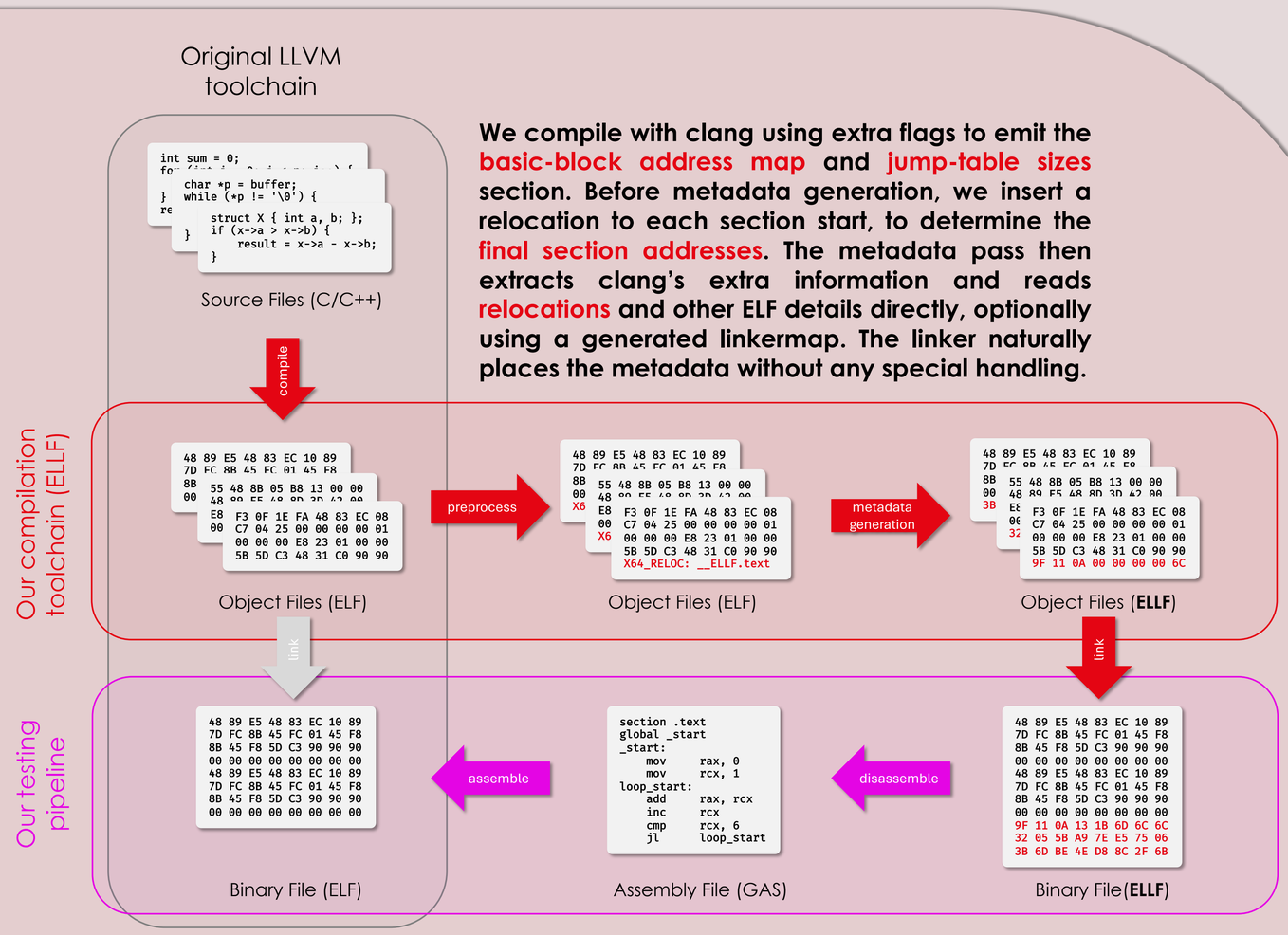


Binaries are the default format for distributing software, but they are inherently opaque. They execute efficiently on modern hardware, yet they reveal almost nothing about their structure or intent: code and data are indistinguishable, references look like raw bytes, and control flow is hidden in pointer indirections. As a result, even simple questions about what a program does become surprisingly hard to answer.

PROTOTYPE

Information	Known By: Bin	Obj	LLVM	Location
Instructions vs Data			■	`.llvm_bb_addr_map`
Raw Data vs References		■		`.rel.*, .rela.*, .symtab`
Code Section Layout			■	`.llvm_bb_addr_map`
Data Section Layout		■		`.rel.*, .rela.*,` Linker Map
Data Section Alignment		■		`.rel.*, .rela.*,` Linker Map
Jump Table Contents			■	`.llvm_jump_table_sizes`
Switch Table Contents	■			
Exception Information	■			`.eh_frame, .gcc_except_table`
Control Flow Graph			■*	LLVM IR
Call Graph			■*	LLVM IR
Section Locations			■	Linker Map

* while the CFG/CG is correct, it does not map 1-to1 to the BB address map



We compile with clang using extra flags to emit the basic-block address map and jump-table sizes section. Before metadata generation, we insert a relocation to each section start, to determine the final section addresses. The metadata pass then extracts clang's extra information and reads relocations and other ELF details directly, optionally using a generated linkermap. The linker naturally places the metadata without any special handling.

RESULTS

Name	Compile Time (seconds)		Binary Size (Kibibytes)			Exec Time (Milliseconds)		Test Result	
	Baseline	ELLF	Baseline	DWARF	ELLF	Baseline	ELLF	Baseline	Recompiled
alacconvert-decode	17.2	26.7	64	178	75	18	19	PASS	PASS
alacconvert-encode	16.8	27.2	64	178	75	25	24	PASS	PASS
burg	25.8	43.6	89	218	113	13	13	PASS	PASS
clamscan	141.7	222.6	977	2297	1104	82	81	PASS	PASS
…	…	…	…	…	…	…	…	…	…
frame_layout	2.7	4.4	145	937	193	16	17	PASS	FAIL
TOTAL	2875	4504 (+57%)	22534	58126 (+158%)	28523 (+27%)	181123	178261 (-1.6%)	0 FAIL	1 FAIL

We first compile the LLVM test suite at -O2 to establish a baseline for compiler performance and DWARF overhead. Next, we compile the same tests using our tool to measure its compile-time and binary size overhead. Finally, we disassemble the output of our tool, reassemble it with GCC, and rerun the tests to verify that the transformations preserve the correctness of the original programs.

The GOOD

`--llvm-bb-addr-map`

The basic block address map worked reliably across all test in both the binaries and object files

`--jump-table-sizes`

Early versions of our tool needed a heuristic to group the jump tables

The BAD

`--switch-table-sizes`

There is no flag similar to the jump table sizes flag for other compiler generated tables

`CFGs / CGs`

We could not get clang to emit any CFG/CG with the same BB layout as in the BB-Addr-Map

The UGLY

`-Wl,-Map=<bin>.map`

The linker map is extremely unreliable, especially for C++ programs. We needed to work around it with synthetic symbols + relocations.

`.rodata merging`

Our disassembler does not yet support merging different types of .rodata sections. There seems to be no reliable way to stop clang from merging them.

Source code of ELLF (Executable, Linkable and Liftable Format), the tool presented in this poster.

github.com/ssrg-vt/ELLF

Background theory: formal conditions required to make disassembly decidable.

link.springer.com/chapter/10.1007/978-3-031-64626-3_8

This work is supported by the Defense Advanced Research Projects Agency (DARPA) under Prime Contract No. HR001124C0492 and Naval Information Warfare Center Pacific (NIWC Pacific) under Contract No. N6600121-C-4028.