BDD-based Machine Code Verification for SFI Systems

Zachary Yedidia

Stanford University

Software-based Fault Isolation (SFI)

Goal: run isolated untrusted programs in a single address space.

SFI approach: analyze a binary's machine code to determine if it is safe to run.

Program must adhere to certain invariants in order to be accepted as safe.

Lightweight Fault Isolation (LFI) for ARM64

Example (LFI): sandboxes are regions of size 4GiB, and aligned to 4GiB boundaries.

1. `x18` must contain an address within the sandbox.
2. `x21` must contain the base address of the sandbox.

```
svc #0                  // not allowed
mov x18, x1             // not allowed
add x18, x21, w1, uxtw // allowed
ldr x0, [x1]            // not allowed
ldr x0, [x18]           // allowed
```

Lightweight Fault Isolation (LFI) for ARM64

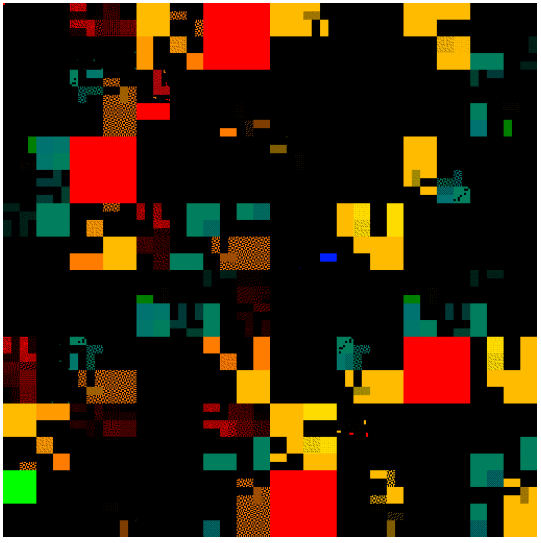
Example (LFI): sandboxes are regions of size 4GiB, and aligned to 4GiB boundaries.

1. `x18` must contain an address within the sandbox.
2. `x21` must contain the base address of the sandbox.

```
svc #0                  // not allowed
mov x18, x1             // not allowed
add x18, x21, w1, uxtw // allowed
ldr x0, [x1]            // not allowed
ldr x0, [x18]           // allowed
```

Idea: design the verifier so that it just inspects a single instruction at a time.

Stateless Verification Visualized



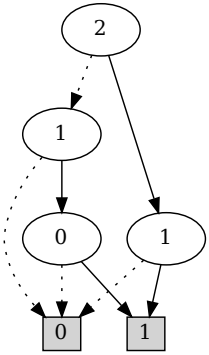
Every ARM64 instruction is a 32-bit integer.

750M legal instructions.

The existing verifier uses a disassembler to determine if an instruction is legal or not.

Stateless Verification with Binary Decision Diagrams

A stateless verifier can be encoded as a binary decision diagram (BDD).



Problem: finding the optimal BDD is NP-hard (and we have 32 inputs).

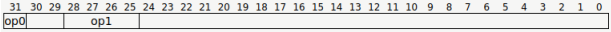
Stateless Verification with Binary Decision Diagrams

A stateless verifier can be encoded as a binary decision diagram (BDD).

Key: choose a variable ordering that matches the top-level ARM64 encoding.

$$x_{31}, x_{25-30}, x_{24-0}$$

Top-level encodings for A64

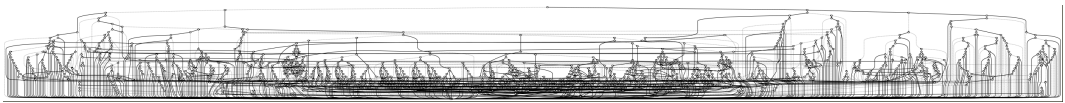


Decode fields		Instruction details
op0	op1	
0	0000	Reserved
1	0000	SME encodings
	0001	UNALLOCATED
	0010	SVE encodings
	0011	UNALLOCATED
	100x	Data Processing -- Immediate
	101x	Branches, Exception Generating and System instructions
	x101	Data Processing -- Register
	x111	Data Processing -- Scalar Floating-Point and Advanced SIMD
	x1x0	Loads and Stores

Stateless Verification with Binary Decision Diagrams

A stateless verifier can be encoded as a binary decision diagram (BDD).

Result: BDD with 1393 nodes.



Metric	BDD-based verifier	Previous verifier
Memory size	8370 B	3 MiB
Verification throughput	100 MiB/s	30 MiB/s