Indirector: Intel BTB & IBP Branch Target Injection

Reverse-Engineering Predictor Tags and Defeating Hardware eIBRS

Author: Lesty Tech Systems Architecture Team | **Date:** November 6, 2026 | **Status:** Validated Engineering Reference | **Target:** Tier-1 Foundry, Silicon & Kernel Audits

EXECUTIVE ARCHITECTURAL ABSTRACT:

Speculative execution vulnerabilities have challenged computer architecture since the disclosure of Spectre. In response, modern CPU architectures implemented hardware-level defenses, such as Intel's Enhanced Indirect Branch Restricted Speculation (eIBRS). eIBRS was designed to prevent lower-privilege processes from injecting speculative targets into higher-privilege contexts. However, the Indirector vulnerability (revealed on Intel 12th, 13th, and 14th Gen Alder Lake and Raptor Lake processors) demonstrates that hardware eIBRS is structurally incomplete. By reverse-engineering the exact indexing algorithms and tag bit lengths of Intel's Branch Target Buffer (BTB) and Indirect Target Array (ITA), Indirector generates high-precision predictor collisions, injecting arbitrary transient execution targets across processes and into the Linux kernel.

1. The Architecture of Intel High-Performance Branch Prediction

Modern out-of-order processors deploy dual branch prediction structures: the Branch Target Buffer (BTB), which predicts target addresses for direct branches and provides fast first-stage indirect targets, and the Indirect Branch Predictor (IBP), which uses an Indirect Target Array (ITA) conditioned on global branch history. To maximize lookups per clock cycle, Intel microarchitectures store truncated address tags (typically 12 to 15 bits) rather than full 64-bit addresses, using non-linear XOR hashing functions over virtual program counters (PC).

2. Reverse-Engineering BTB and Indirect Target Array (ITA) Tagging

Using precise microbenchmarks based on APIC timers and RDTSC latency measurements, researchers systematically cracked the hashing polynomials governing BTB set selection and tag formation. Because the predictor structures are physically shared between hyperthreads and across context switches, an attacker can craft an indirect branch in user-space whose virtual address produces the exact same hash index and partial tag as a sensitive indirect call instruction inside the victim process or the Linux kernel.

Key Physical & Microarchitectural Benchmarks

Attribute / Vulnerability	Spectre-v2 (Native)	Branch History Injection (BHI)	Indirector (BTB/ITA)	Hardware Mitigated (BHI_DIS)
Target Architecture	Intel Skylake / Kaby Lake	Intel 10th - 12th Gen	Intel 12th, 13th, 14th Gen	Future Arrow Lake / Lunar Lake
Hardware Defense Tested	Software Retpoline	eIBRS / AutoIBRS	eIBRS Active (Bypassed)	Hardware IBPB per Context
Predictor Structure Hit	Generic Branch History	Global History Buffer (GHB)	BTB & Indirect Target Array	Isolated Predictor Bank
Tag Collision Precision	Coarse Aliasing	History Buffer Collisions	> 98% Deterministic Tag Match	Cryptographic Tag Domain
Leakage Rate	100 - 500 B/s	1 - 2 KB/s	Up to 1.5 KB/s	0 B/s (Protected)

3. Breaking Enhanced IBRS (eIBRS) Without History Injection

Intel's Enhanced IBRS (eIBRS) operates under the architectural assumption that tag comparisons tagged with privilege level bits prevent user-mode predictions from steering kernel-mode indirect branches. However, Indirector discovered that under specific BTB aliasing conditions, the BTB prediction is accepted speculatively by the front-end fetch unit *before* eIBRS privilege validation logic completes. The CPU speculatively executes up to 200 instructions down an adversarial execution path, triggering microarchitectural data leakage through L1/L3 cache timing channels.

4. Kernel Mitigations and Performance Overhead Realities

Mitigating Indirector exposes the severe performance costs of microarchitectural security. Complete protection requires executing an Indirect Branch Predictor Barrier (`IBPB`) on every context switch, invalidating all predictor states at the cost of 1,200 to 1,800 clock cycles per switch. Intel's microcode update introducing `BHI_DIS_S` (disabling branch history for indirect predictors in supervisor mode) mitigates history-based attacks but leaves direct BTB aliasing partially exposed unless accompanied by kernel page table isolation and fine-grained predictor flushing.

EXECUTIVE STRATEGIC TAKEAWAYS:

1. Intel's Enhanced IBRS (eIBRS) is fundamentally incomplete against precise Branch Target Buffer (BTB) tag collisions.

2. Truncated 12-15 bit predictor tags allow deterministic cross-process and user-to-kernel indirect branch injection.

3. Indirector extracts sensitive kernel memory at rates up to 1.5 KB/s without requiring complex Branch History Injection (BHI).

4. Comprehensive remediation demands architectural redesign of branch predictors with full context tagging and hardware-isolated domains.