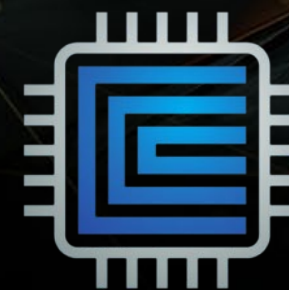




AUGUST 7-8, 2024
MANDALAY BAY/LAS VEGAS

LIBIHT: A Cross-Platform Library for Accessing Intel Hardware Trace Feature

Changyu Zhao, Di Wu, Guancheng Li



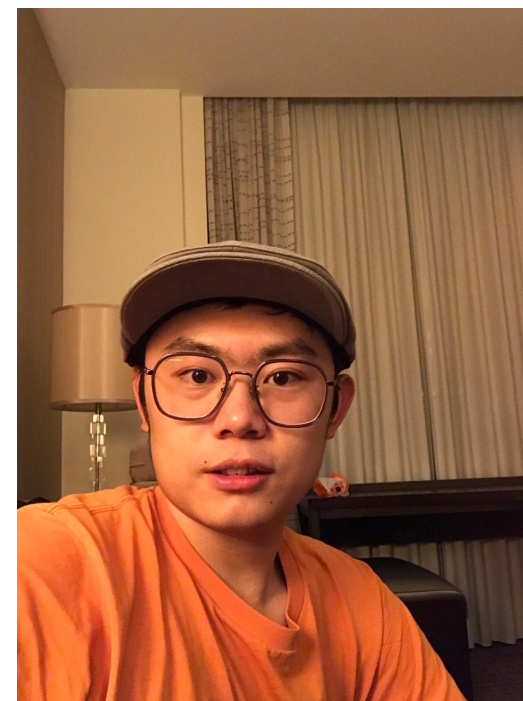
LibIHT

Intel Hardware Trace Library

Who Are We

Changyu Zhao (upper left)

- Undergrad at University of Wisconsin-Madison



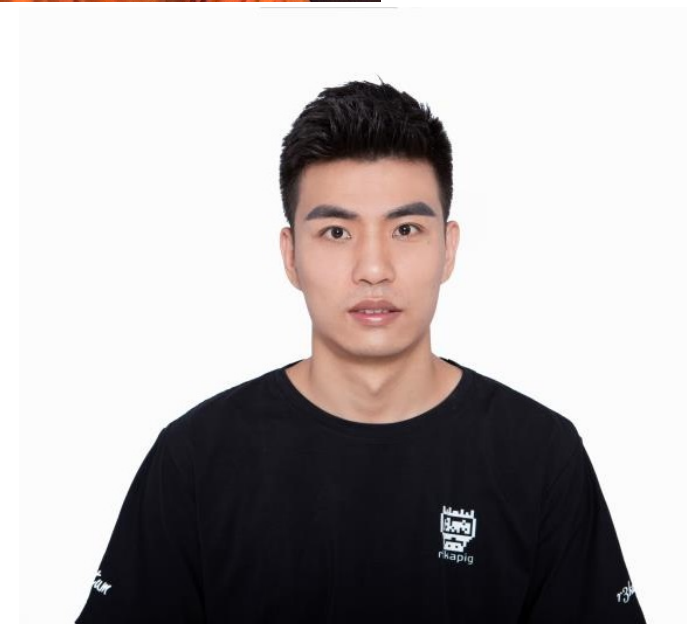
Di Wu (upper right)

- Undergrad at Peking University



Guancheng Li (bottom)

- Senior Security Researcher at Tencent Security Xuanwu Lab

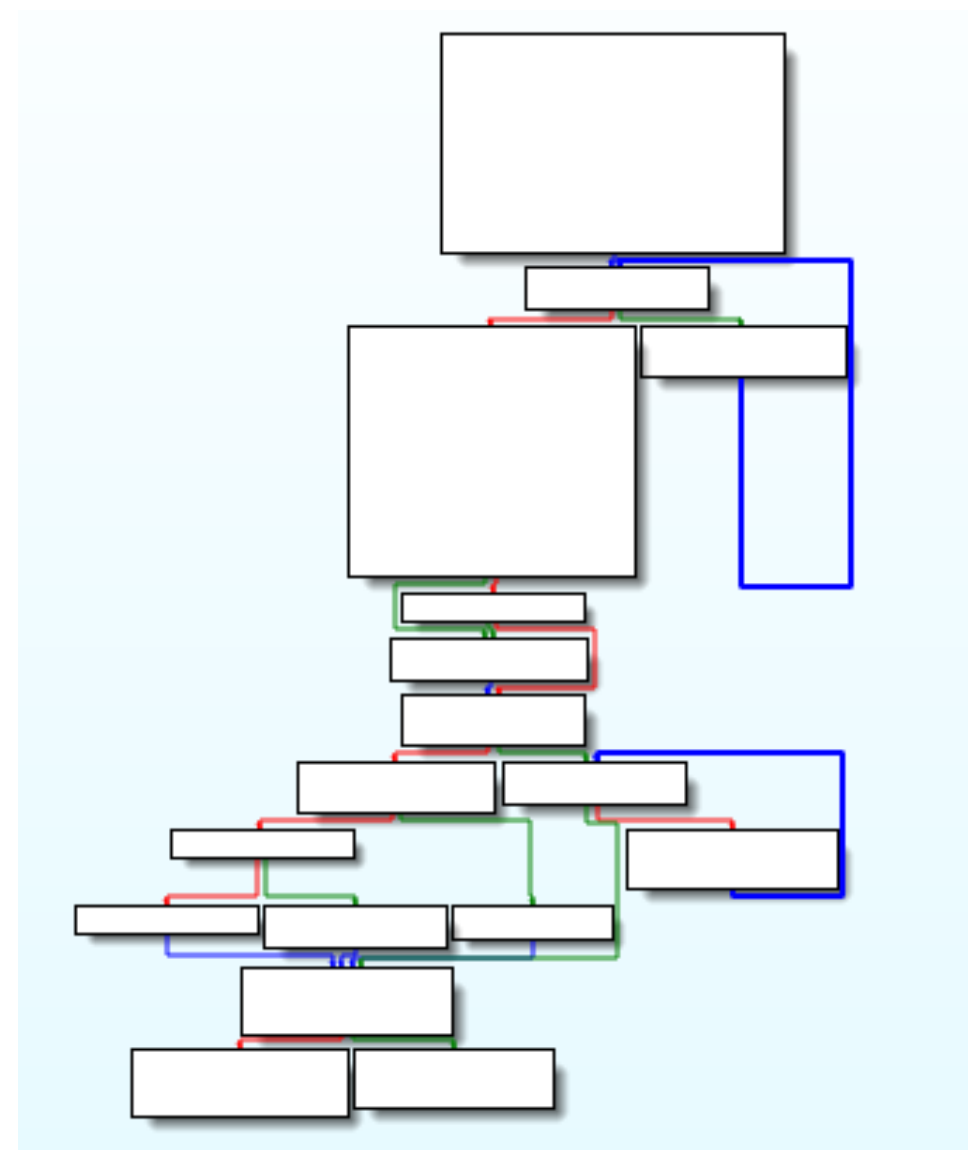


Agenda

- Introduction of Program Tracing
- Intel Hardware Tracing Features
- Traditional Tracing and Limitations
- LibIHT Architecture and Features
- Use Cases and Demonstrations
- Open Source and Future Development

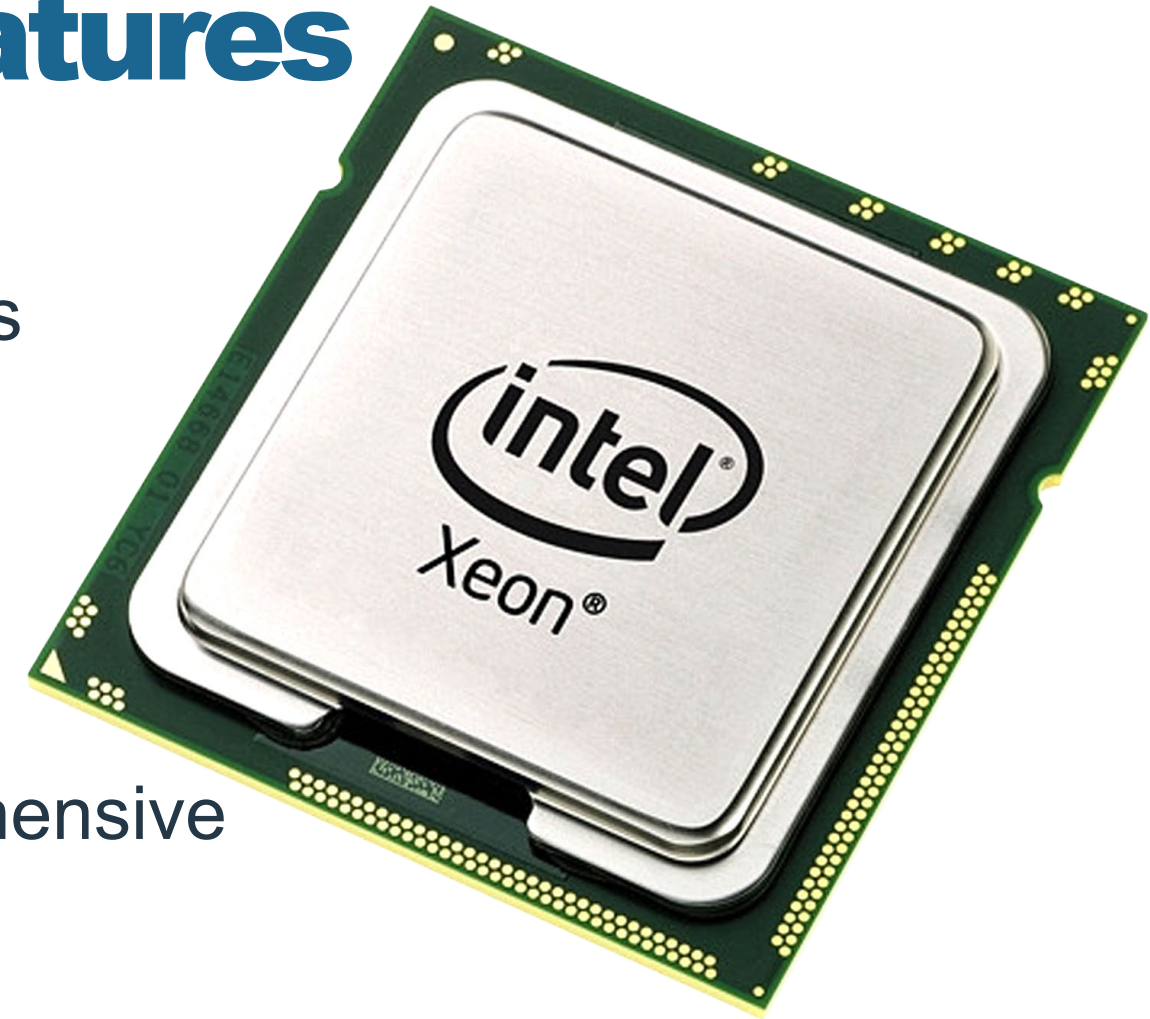
What is Program Tracing

- **Tracing** is the art of understanding and reconstructing the execution behavior of a program
- Trace Levels
 - Instruction, Function, and Basic Block Tracing
- Trace Techniques
 - Dynamic Binary Instrumentation
 - Debugger Based
 - Emulator Based
 - Hardware Based



Intel Hardware Tracing Features

- Last Branch Record (LBR)
 - Stores most recent branches taken in registers
 - Limited capacity but very low overhead
- Branch Trace Store (BTS)
 - Logs complete branch history in memory
 - Higher overhead than LBR, but more comprehensive
- Intel Processor Trace (Intel PT)
 - Provides extensive execution tracing capabilities
 - Extremely slow to decode tracing info



An Idea Tracing Technic



Efficiency



Integrity



Usability



Transparency

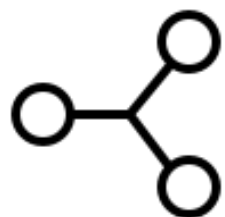


Stability

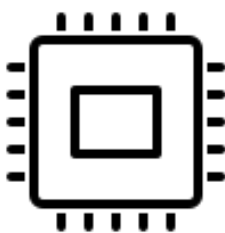
Limitations on Traditional Tracing

Tracing Techniques	Efficiency: Low Overhead & High Performance	Integrity: Accuracy & Completeness	Transparency: Minimal Interference	Stability: Reliability & Robustness	Usability: Friendly API & Flexibility
Dynamic Binary Instrumentation (e.g., Intel PIN, DynamoRIO, Frida)	✗	✓	✗	✗	✓
Debugger-Based Tracing (e.g., GDB, WinDbg)	✗	✓	✗	✓	✓
Emulator-Based Tracing (e.g., QEMU, Unicorn Engine)	✗	✓	✓	✓	?
Hardware-Based Tracing (e.g., Intel Hardware Tracing Features)	✓	✓	✓	✓	?

LibIHT: Intel Hardware Trace Library



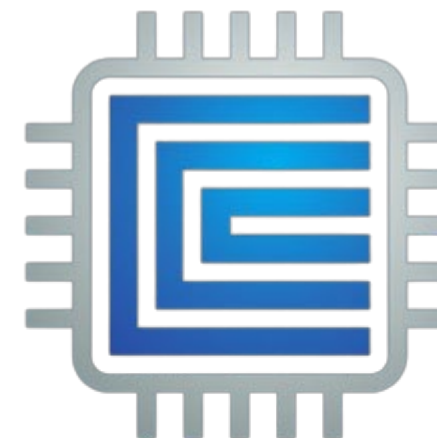
Bridges hardware and user needs



Simplifies hardware tracing



Efficient, low-overhead solution



LibIHT

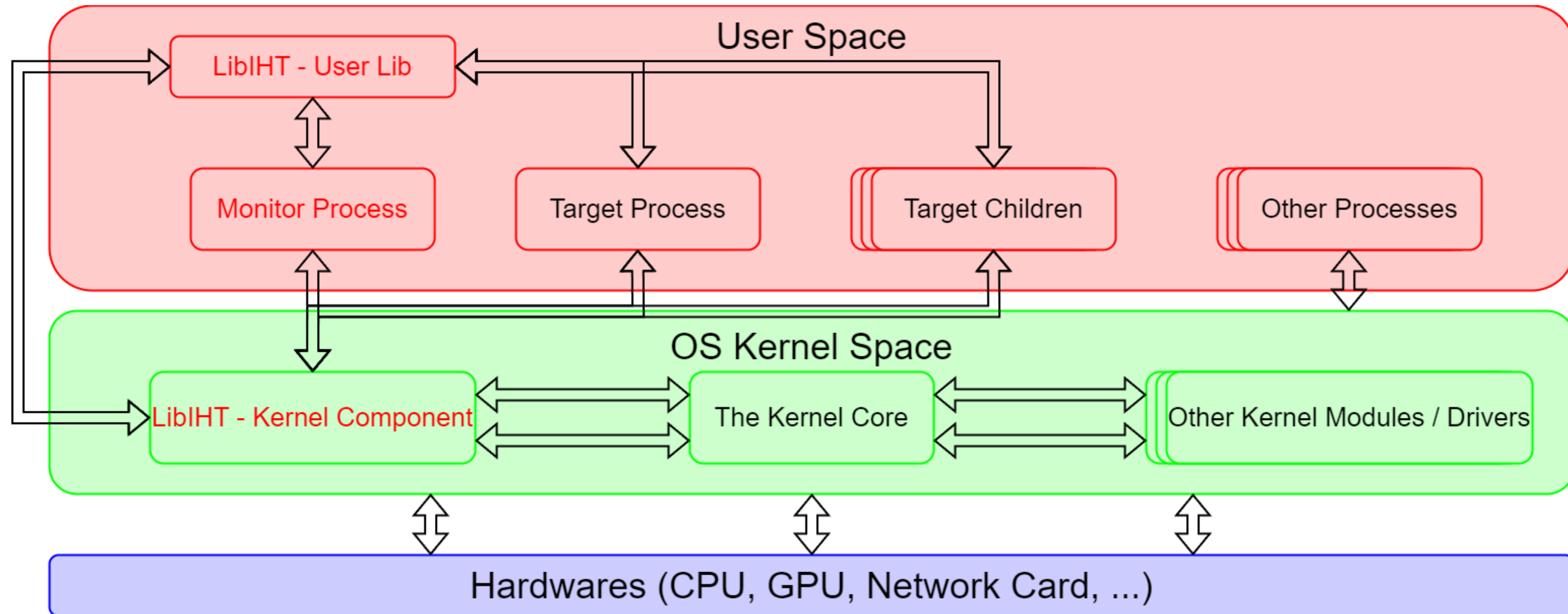
Intel Hardware Trace Library



GitHub

<https://github.com/libiht/libiht>

Introduction to LiblHT: Architecture



Introduction to LiblHT: Features

- Efficient Hardware-Assisted Tracing
 - Low-overhead tracing using Intel CPU features (LBR, BTS)
 - Selective tracing of specific code regions with fine-grained control
- Advanced Analysis Capabilities
 - Complete trace information for control flow reconstruction
 - Detailed execution info for performance and security analysis
- User-Friendly Design
 - Simple and convenient API for trace collection and analysis
 - Cross-platform support for Linux and Windows

Use Cases and Demonstrations

LibIHT Demo: LBR

```
thomason@flare: ~/libiht
[ +0.000000] LIBIHT-COM: Flush LBR on cpu core: 3
[ +0.000012] LIBIHT_LKM: Initilizing BTS...
[ +0.000002] LIBIHT-COM: Init BTS related structs.
[ +0.000001] LIBIHT-COM: Flushing BTS for all cpus...
[ +0.000002] LIBIHT-COM: Flush BTS on cpu core: 1...
[ +0.000002] LIBIHT-COM: Flush BTS on cpu core: 2...
[ -0.000001] LIBIHT-COM: Flush BTS on cpu core: 3...
[ +0.000000] LIBIHT-COM: Flush BTS on cpu core: 0...
[ +0.000005] LIBIHT_LKM: Initilized
[ +1.334831] LIBIHT_LKM: Exiting...
[ +0.000010] LIBIHT_LKM: Exiting BTS...
[ +0.000002] LIBIHT-COM: Flushing BTS for all cpus...
[ +0.000006] LIBIHT-COM: Flush BTS on cpu core: 3...
[ +0.000035] LIBIHT-COM: Flush BTS on cpu core: 0...
[ +0.000001] LIBIHT-COM: Flush BTS on cpu core: 1...
[ +0.000003] LIBIHT-COM: Flush BTS on cpu core: 2...
[ +0.000006] LIBIHT-COM: Freeing BTS state list.
[ +0.000003] LIBIHT_LKM: Exiting LBR...
[ +0.000002] LIBIHT-COM: Flushing LBR for all cpus...
[ +0.000002] LIBIHT-COM: Flush LBR on cpu core: 3
[ +0.000003] LIBIHT-COM: Flush LBR on cpu core: 0
[ +0.000000] LIBIHT-COM: Flush LBR on cpu core: 1
[ +0.000007] LIBIHT-COM: Flush LBR on cpu core: 2
[ +0.000014] LIBIHT-COM: Freeing LBR state list...
[ +0.000002] LIBIHT_LKM: Unregistering tracepoints...
[ +0.000164] LIBIHT_LKM: Removing helper process...
[ +0.000008] LIBIHT_LKM: Exit complete

thomason@flare:~/libiht/kernel/lkm$ ls
include      libiht_lkm.mod.c  Makefile      src
libiht_lkm.ko  libiht_lkm.mod.o  modules.order
libiht_lkm.mod  libiht_lkm.o      Module.symvers
thomason@flare:~/libiht/kernel/lkm$
```

[0] 0:bash* 1:make-

"flare" 07:12 03-Jul-24

LiblHT Demo: BTS

```
C:\projects\libiht\kernel\demo\kmd-demo\x64\Debug>
```



```
thomason@flare: ~
[ +0.000297] LIBIHT_LKM: Removing helper process...
[ +0.000009] LIBIHT_LKM: Exit complete
[ +2.377958] LIBIHT_LKM: Initializing...
[ +0.000005] LIBIHT-LKM: Creating helper process...
[ +0.000005] LIBIHT_LKM: Registering tracepoints...
[ +0.000019] LIBIHT_LKM: Registering tracepoint sched_s
witch
[ +0.000029] LIBIHT_LKM: Registering tracepoint task_ne
wtask
[ +0.000015] LIBIHT_LKM: Initilizing LBR...
[ +0.000001] LIBIHT-COM: DisplayFamily_DisplayModel - 6
_9cH
[ +0.000001] LIBIHT-COM: LBR capacity - 32
[ +0.000001] LIBIHT-COM: Init LBR related structs.
[ +0.000001] LIBIHT-COM: Flushing LBR for all cpus...
[ +0.000001] LIBIHT-COM: Flush LBR on cpu core: 2
[ +0.000000] LIBIHT-COM: Flush LBR on cpu core: 3
[ +0.000000] LIBIHT-COM: Flush LBR on cpu core: 0
[ +0.000000] LIBIHT-COM: Flush LBR on cpu core: 1
[ +0.000007] LIBIHT_LKM: Initilizing BTS...
[ +0.000001] LIBIHT-COM: Init BTS related structs.
[ +0.000001] LIBIHT-COM: Flushing BTS for all cpus...
[ +0.000001] LIBIHT-COM: Flush BTS on cpu core: 2...
[ +0.000000] LIBIHT-COM: Flush BTS on cpu core: 3...
[ +0.000000] LIBIHT-COM: Flush BTS on cpu core: 0...
[ +0.000000] LIBIHT-COM: Flush BTS on cpu core: 1...
[ +0.000002] LIBIHT_LKM: Initilized

thomason@flare:~/libiht/lib/demo/lkm-demo$ ls
gdb-demo  gdb-demo.c  lkm-demo  lkm-demo.c  Makefile
thomason@flare:~/libiht/lib/demo/lkm-demo$
```

[0] 0:bash- 1:bash* "flare" 18:27 02-Jul-24

What's Next?



Next in LibIHT:

- More tracing features (Architectural LBR, CET)
- Better and more powerful user space tool integration
- Community-driven feature requests and improvements

Join LibIHT Community!



<https://github.com/libiht/libiht>