black hat ARSENAL
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

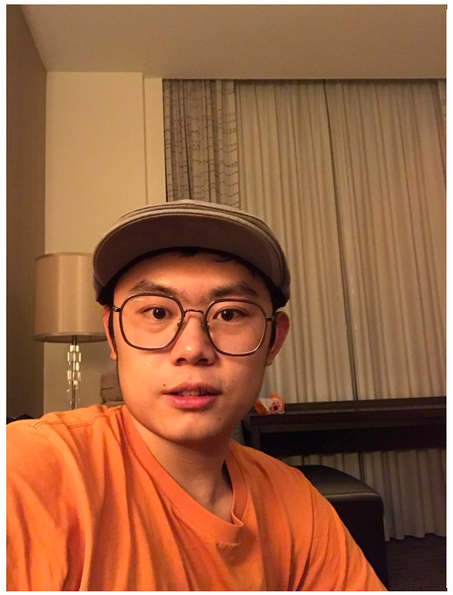
LibIHT
Intel Hardware Trace Library

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

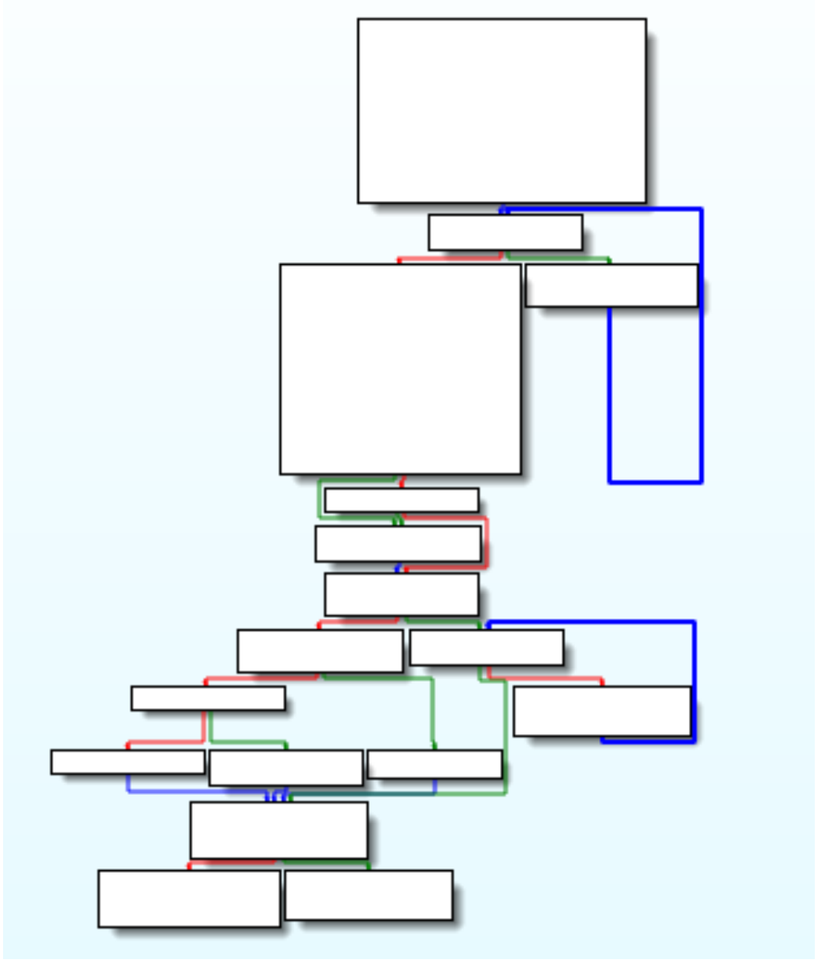


Agenda

- 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

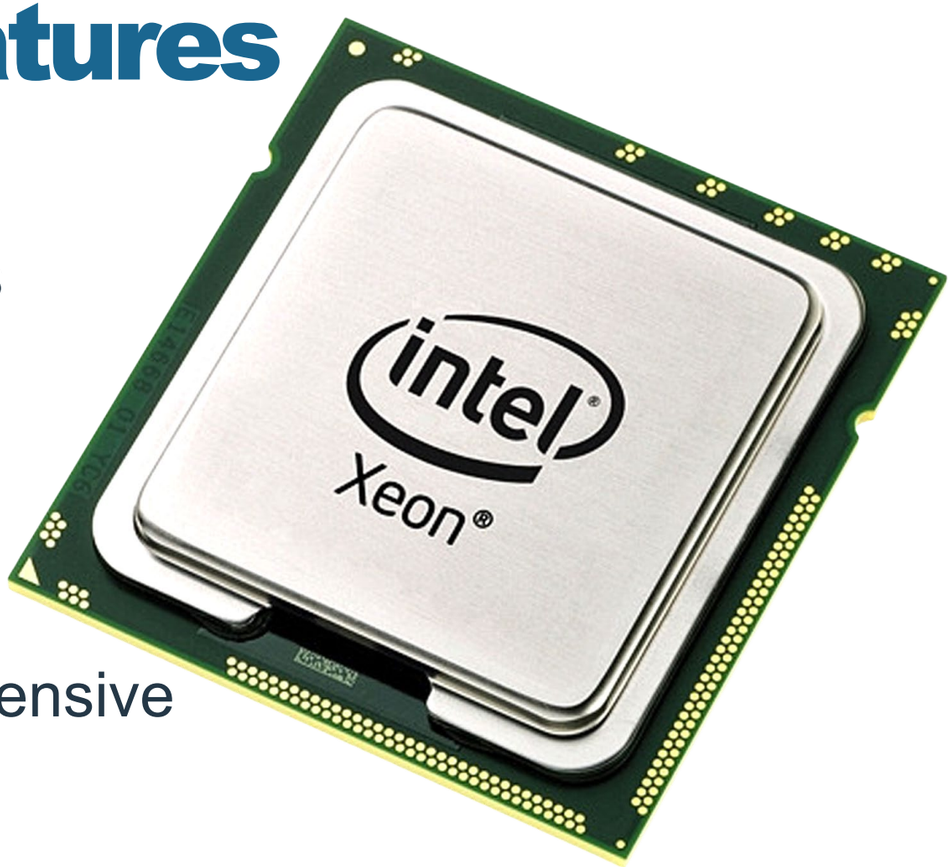
What is Program Tracing

- **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

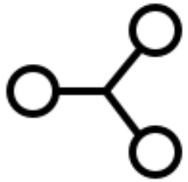
Transparency

Stability

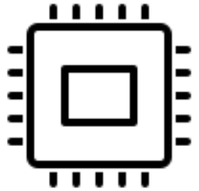
Limitations on Traditional Tracing

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

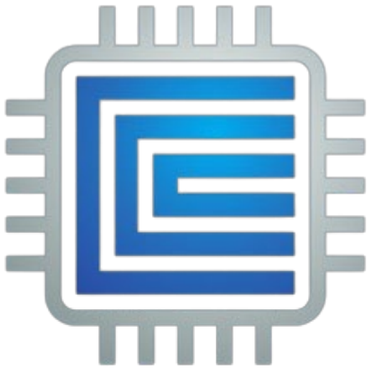


Bridges hardware and user needs



Simplifies hardware tracing

Efficient, low-overhead solution

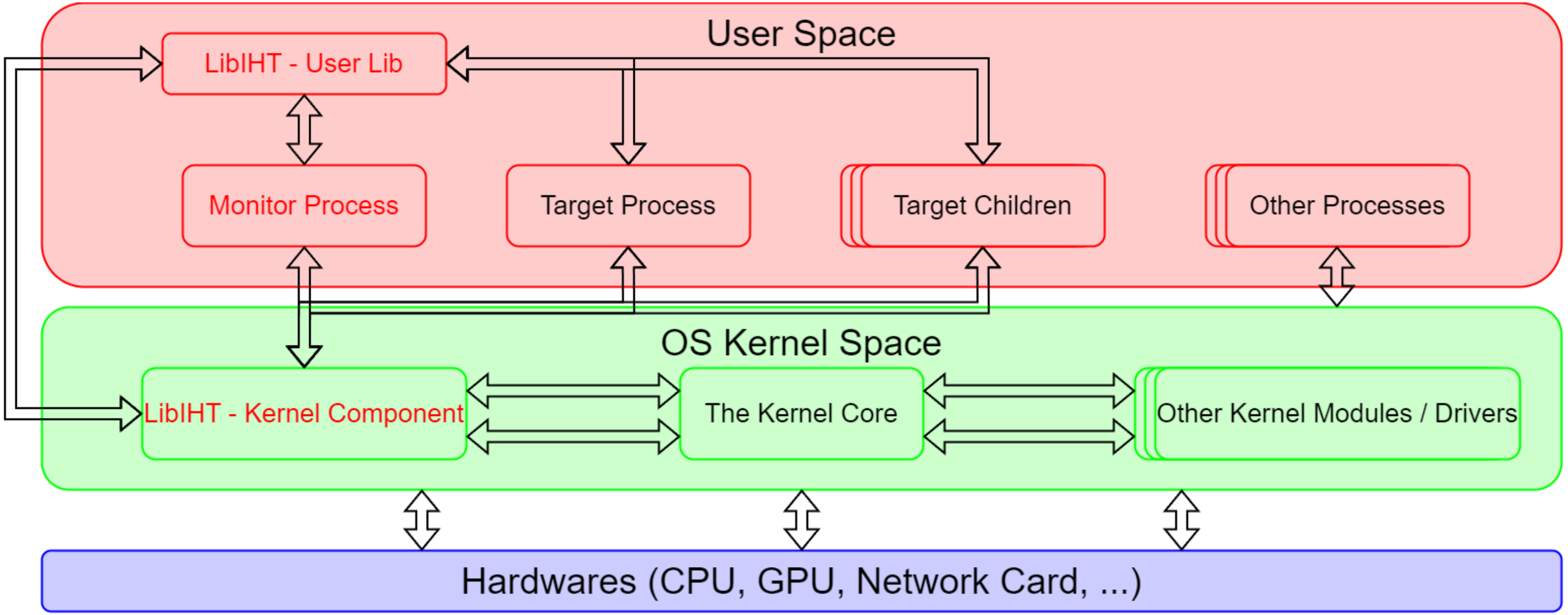


LibIHT

Intel Hardware Trace Library

https://github.com/libiht/libiht

Introduction to LibIHT: Architecture

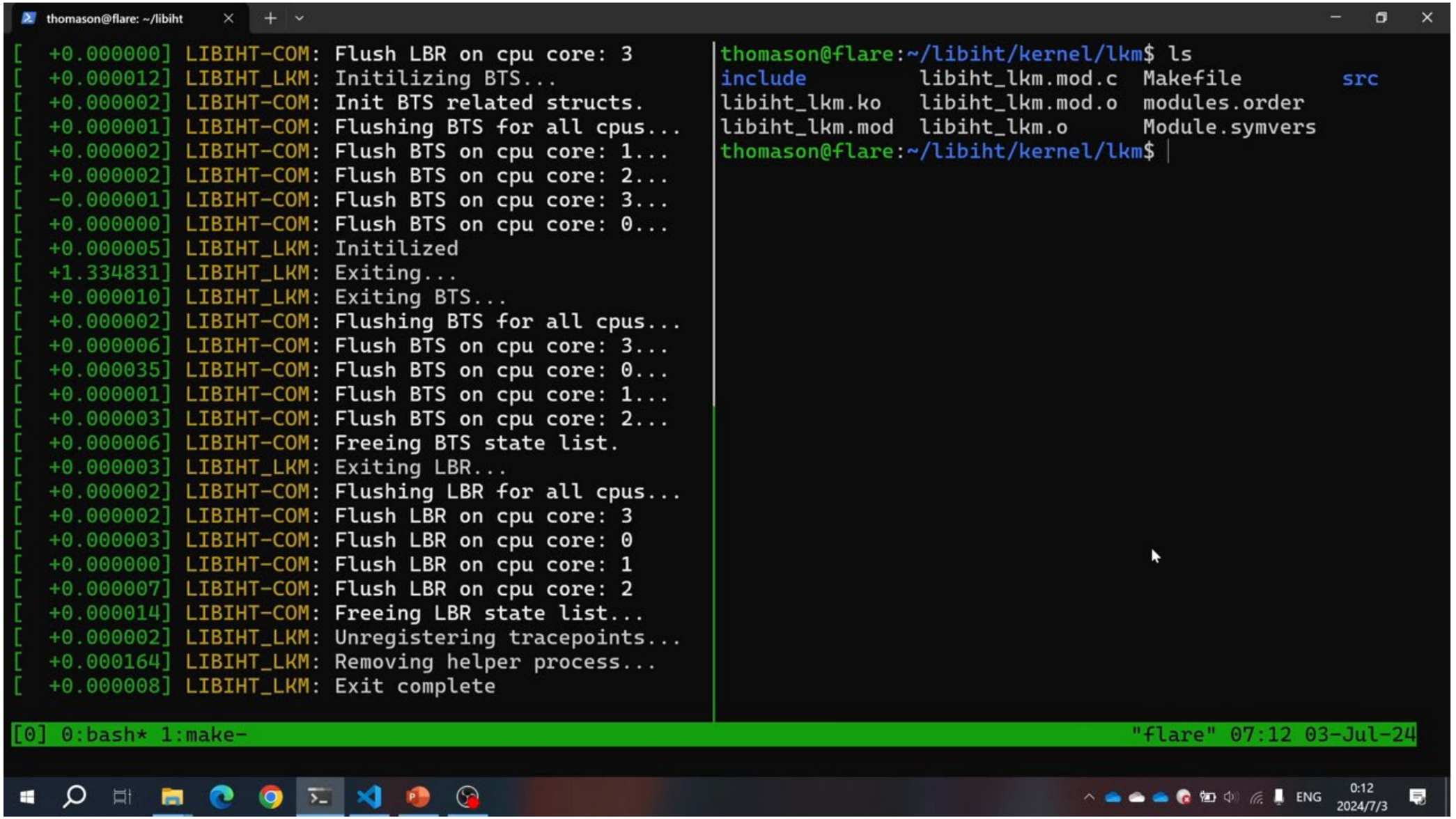


Introduction to LibIHT: Features

- 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

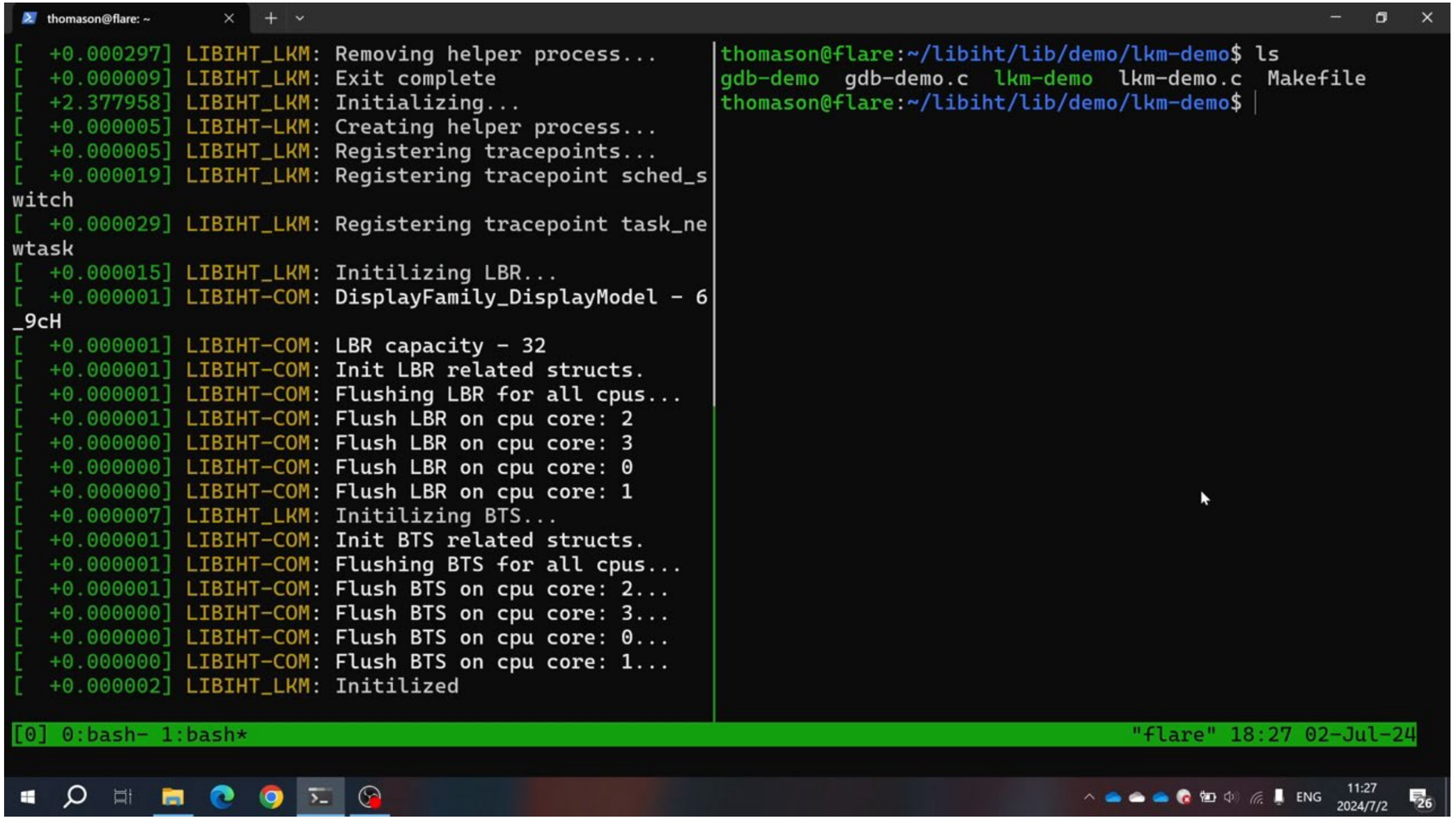
Use Cases and Demonstrations

LibIHT Demo: LBR



LibIHT Demo: BTS

LibIHT Demo: GDB



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!

GitHub

https://github.com/libiht/libiht