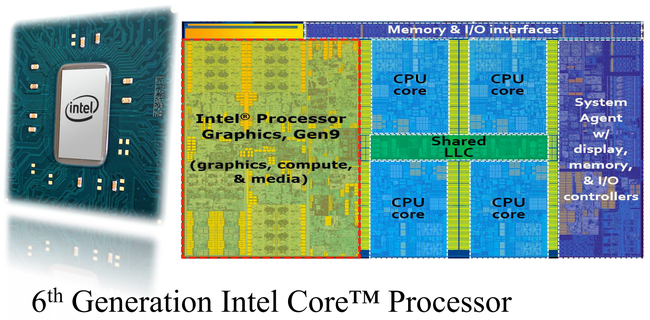
# ProRace: Practical Data Race Detection for Production Use

**Tong Zhang, Changhee Jung, Dongyoon Lee**

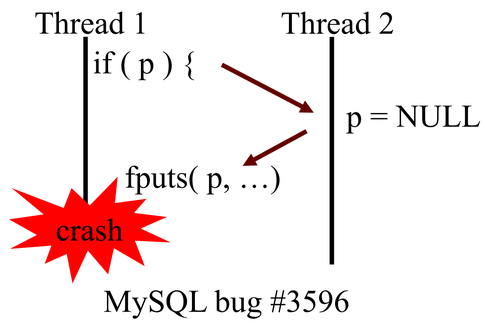
Department of Computer Science, Virginia Tech

## MOTIVATION

- Multithreaded programs are vulnerable to data race errors

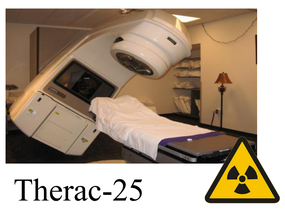



6th Generation Intel Core™ Processor



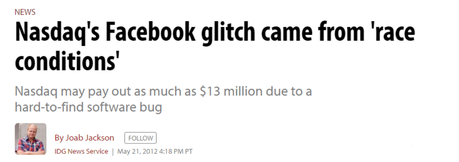

MySQL bug #3596

- Race Condition have caused severe real-world problems



Therac-25

Northeast Blackout (2003)



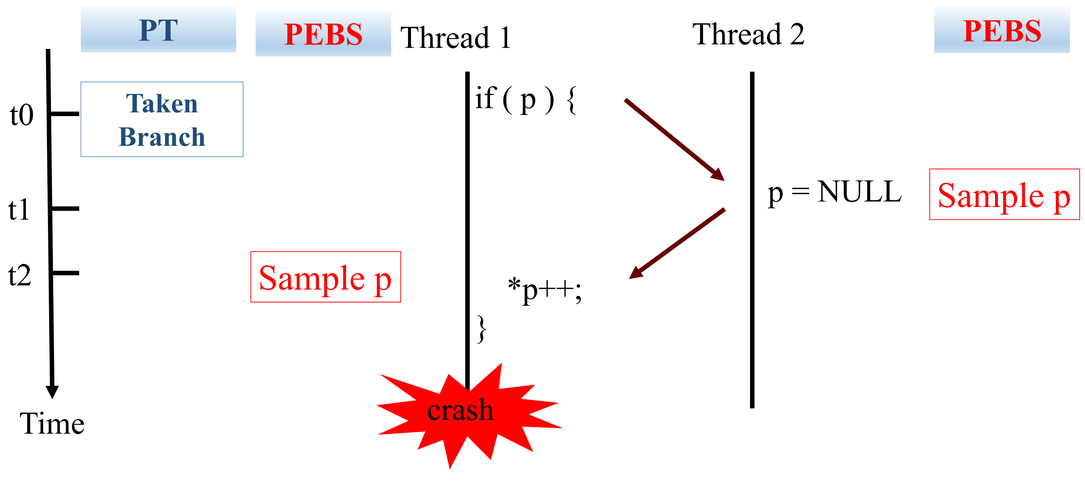

Nadaq stock price mismatch (2012)

Problem: • The state-of-the-art dynamic data race detectors are very slow
- e.g. FastTrack, Intel's Inspector XE, and Google's TSan: 10-100x

• Sampling based data race detector:
- high sampling overhead, e.g. LiteRace(1.47x), Pacer(1.86x)
- low detection capability, e.g. DataCollider, RaceZ

## BACKGROUND

- Sampling Based Detector
- **PT**: full program control flow **PEBS**: sample architecture state

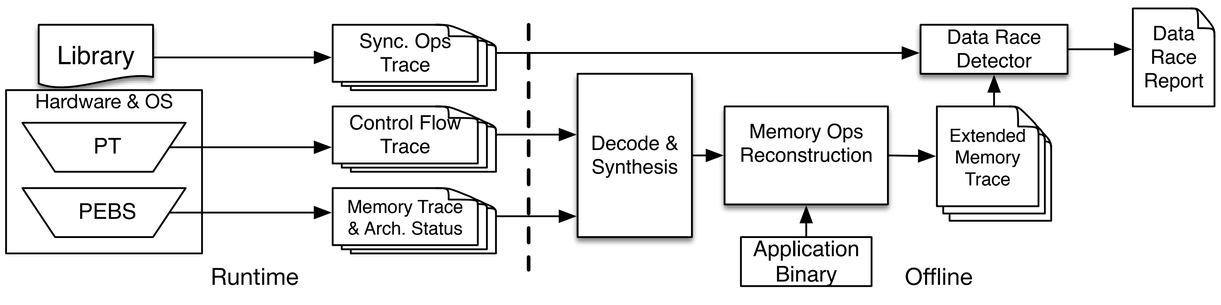



- Goal of data race detection in production environment
  - Low overhead, more samples for given performance budget
  - High detection capability

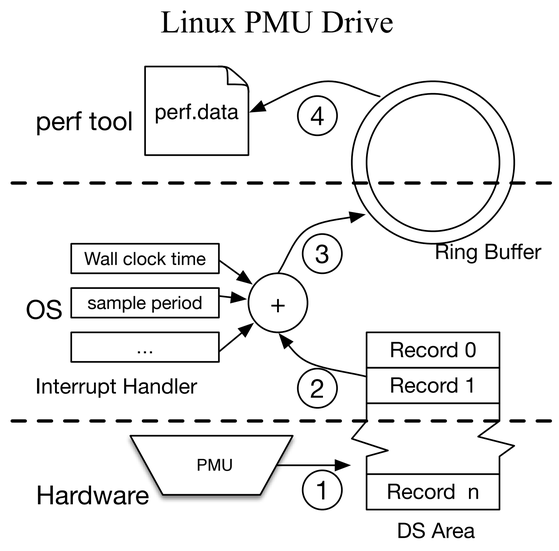
## OUR APPROACH

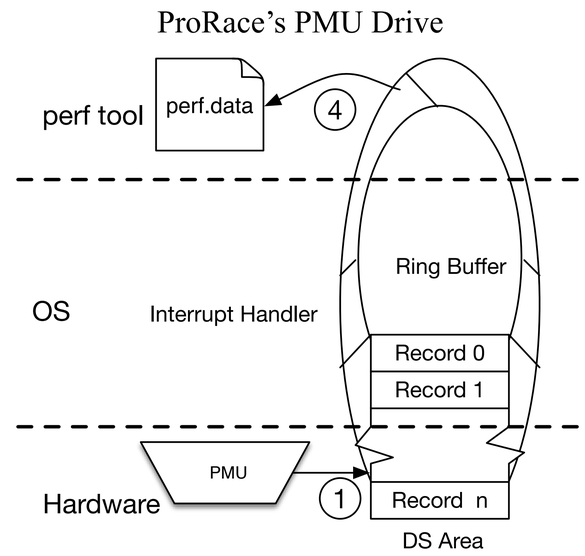
ProRace Overview:

- Enhanced PMU Driver
- Forward and Backward Replay to Reconstruct More Un-sampled Memory Operation

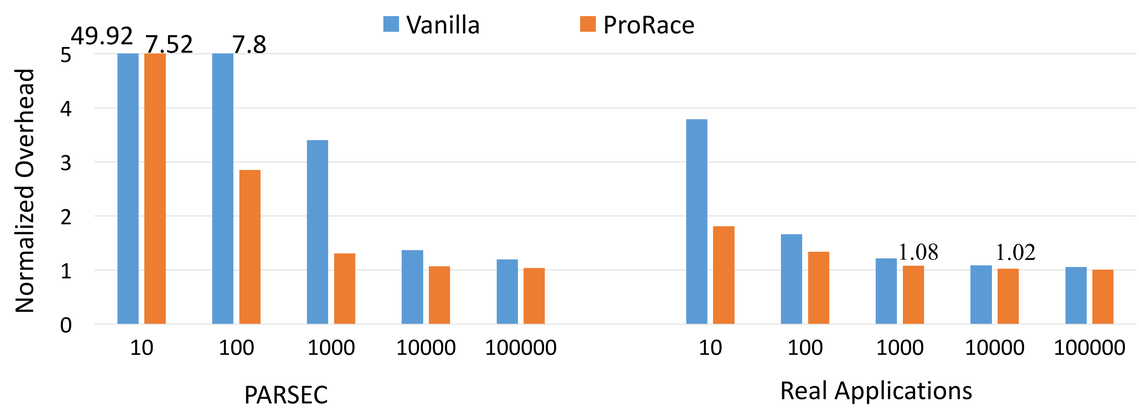



## LIGHTWEIGHT PROGRAM TRACING



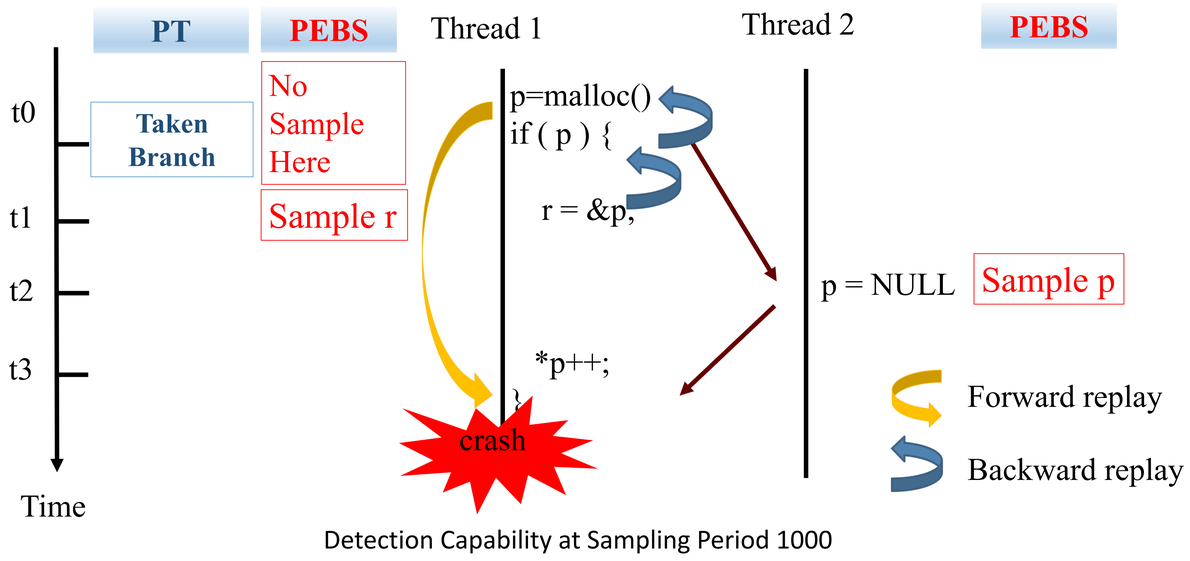




We compared ProRace's new PMU driver with Linux's PMU driver

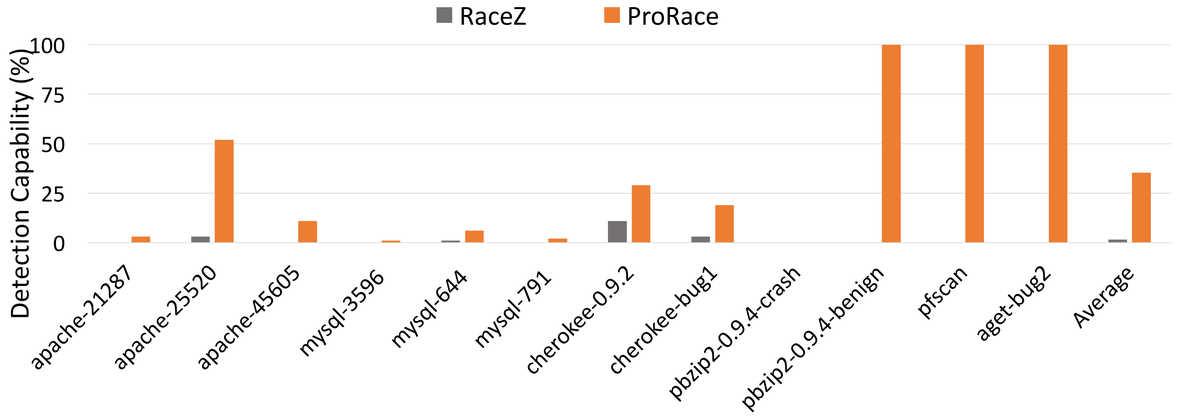



## HIGH DETECTION CAPABILITY VIA REPLAY

Recover unsampled memory accesses using backward and forward replay across basicblock, while RaceZ only support replay within single basicblock




Detection Capability at Sampling Period 1000




## CONCLUSION

ProRace lowers the runtime cost of sampling and introduced new technique to reconstruct un-sampled memory accesses, which enhanced data race detection coverage.

* Part of this research is supported by NSF and Google.