STAN – Current State and Future Work

Scalable Hardware-Aided Trusted Data Management

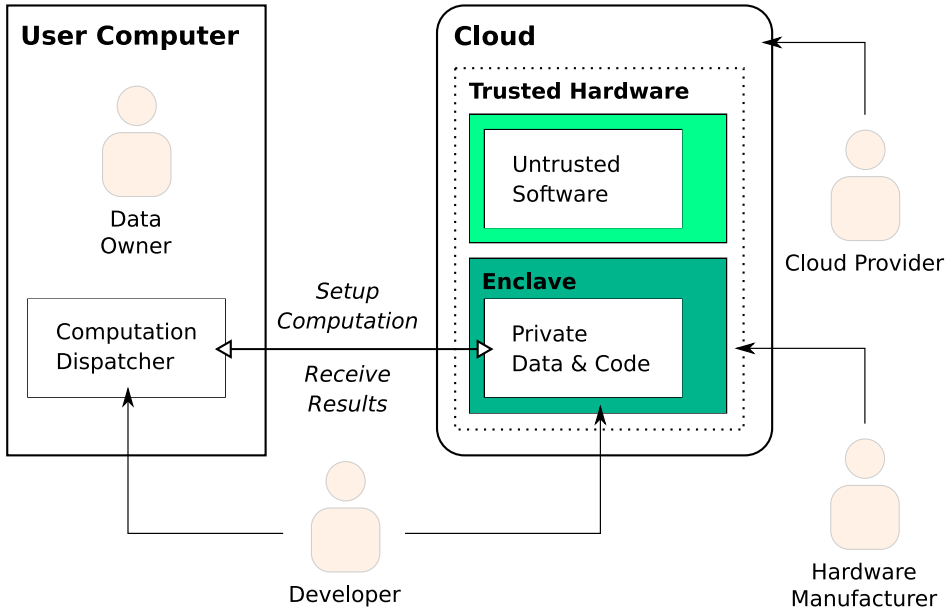
Nico Weichbrodt, 2019-09-03/04

Moving Databases to the Cloud

- Everyone moves to the cloud
- Higher scalability and availability
- But: no trust in the cloud provider

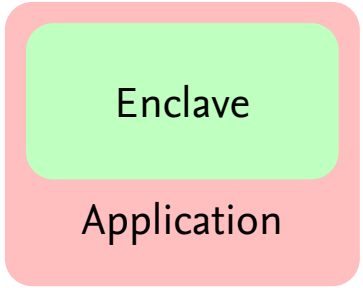
$\rightarrow$ Trusted Execution with Intel SGX

$\Rightarrow$ DBMS needs to be adapted



Intel SGX Recap

- Enclave are isolated compartments
- Confidentiality and integrity protection
- Special memory area for enclaves: EPC
- EPC size very small: ≈93 MiB on current hardware
  - There are plans to increase this
- Memory Layout of enclaves fixed after creation
- *NEW*: SGXv2
  - add/remove/change permissions of pages after enclave creation



Previous Research

- Efficiently handling data inside an SGX enclave
  - SGX-aware virtual memory engine
  - Prototype based on SQLite

  → STANlite (Sartakov et al., IC2E 2018)

- Analysing SGX performance overhead
  - Trace enclave transitions
  - Give recommendations on how to improve performance

  → sgx-perf (Weichbrodt et al. Middleware 2018)

- Reducing database memory footprint inside enclave
  - Decomposing of applications into features
  - Dynamically load and unload features into/out of enclaves

  → Adaptive SGX-enabled Systems (Krieter et al. VaMoS 2019)

Work Program Status – 1.5 Years Later

- WP3: Proactive Working Set Management (TUB)
  - Prefetching experiments, ≈20% performance inc. in best case
  - Not evaluated further
- WP4: Extended Code Generation for Secure Interaction (TUB/HSH)
  - No automatic partitioning
  - *sgx-perf* code recommendations
- WP5: System Support for Integrity Preservation (TUB)
  - STANlite virtual memory engine
- WP6: Trust-aware DBMS Architecture (HSH)
  - Adaptive SGX-enabled Systems with dynamic loading

$\Rightarrow$ First Half done

Current Research

- Current dynamic loading has a couple issues
  - ✗ Not thread safe, no global objects
  - ✗ Not a real dynamic linker (no linking of jumps/calls)
  - ✗ Function pointer ownership problems
  - ✗ Based on SGXv1 with executable heap

⇒ New version called *sgx-dl*

- Fix all issues above and use SGXv2
- *NEW:* Hot-Patching for functions

Architecture

- sgx-dl is a simple library linked into the enclave
- SXG SDK has preliminary support for SGXv2
  - We have our own additional changes (<100 SLOC driver, <1000 SDK)

- All issues fixed: thread safe ✓, support for global object ✓, no fixed memory layout ✓, no function pointers ✓, no executable heap ✓
- New issues
  - No function pointers and unloading results in a lot of checks $\rightarrow$ performance overhead ✗
  - Optimisation: Everything is always loaded mode

Language Support

- sgx-dl is language agnostic, it loads ELF files
  - In theory, everything that uses/supports C ABI should work

- Complete (tested) support for C and its features ✓
  - Loaded functions that are dynamically called need to have signature `void *fctname(void *args)` like pthread threads
- Some support for C++ ✓
  - SGX C++ standard library exists, but we did not really test this
- Rudimentary support for Rust ✗
  - No Rust standard library so only `[no_std]` code is supported
  - Baidu Rust SGX SDK might work here

Hot-Patching

Problem:

- Enclave is potentially large (almost or even bigger than EPC size)
- Patches need to be applied $\rightarrow$ restart needed
- Encrypt state, save to disk, restart enclave, reload state and decrypt

Solution:

- Hot-Patching of the affected functions, no restart needed
- Only possible for already dynamic functions

First Benchmarks

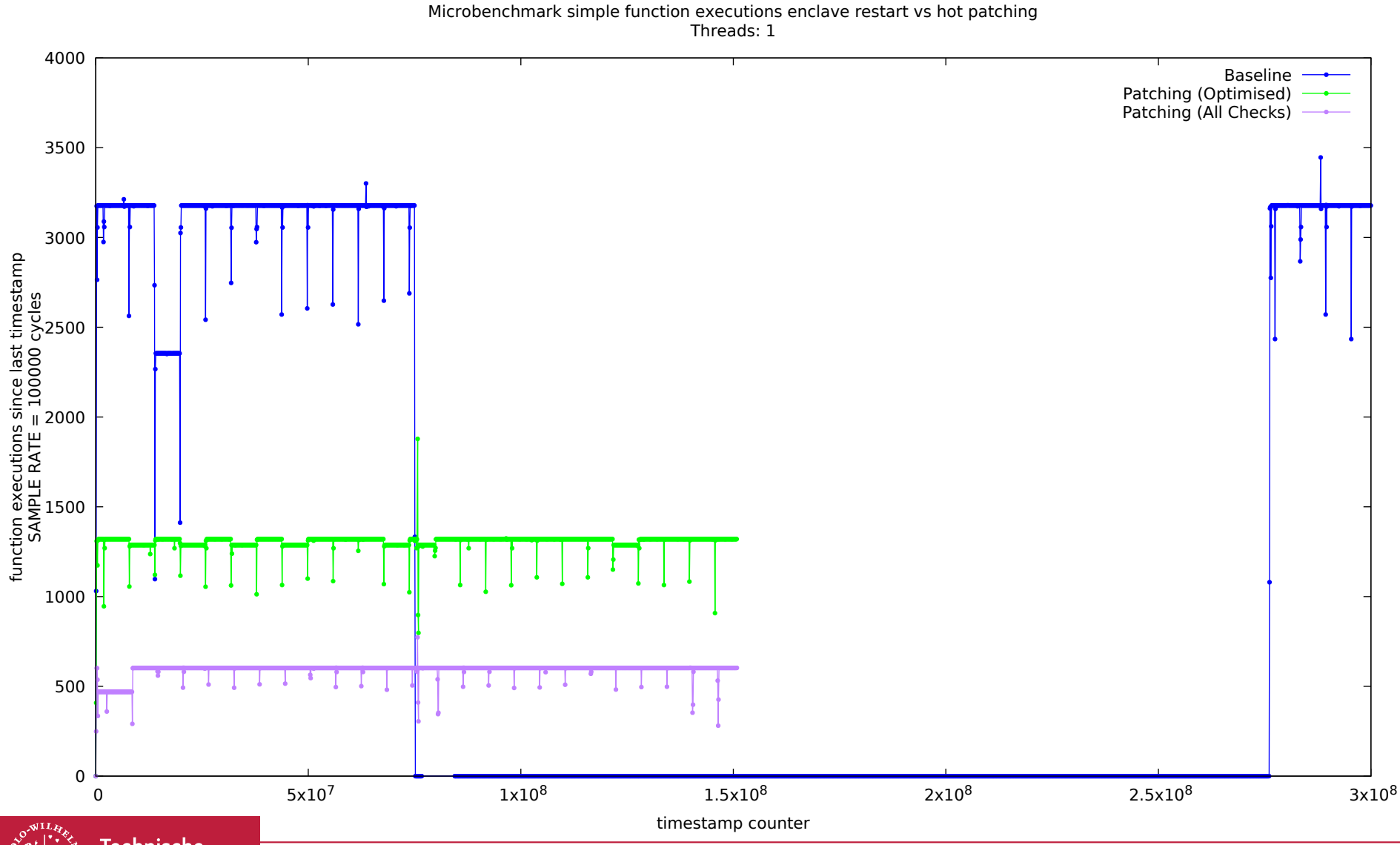
System:

- Intel NUC7PJYH2, Intel Pentium Silver, 4x 1.5GHz (SGXv2 capable)
  - SGXv2 only available with Gemini Lake so far
  - Skyhawk Lake: 2020
  - Cooper Lake: 2020, SGX on multi-socket servers
- 8 GiB memory, SSD

Microbenchmark:

- Tiny enclave, measure call overhead to dynamic functions
- Load three functions, call one, this one calls the other two
- Enclave restart time: 0.13 s
- **These enclaves are tiny so we don't save that much time**

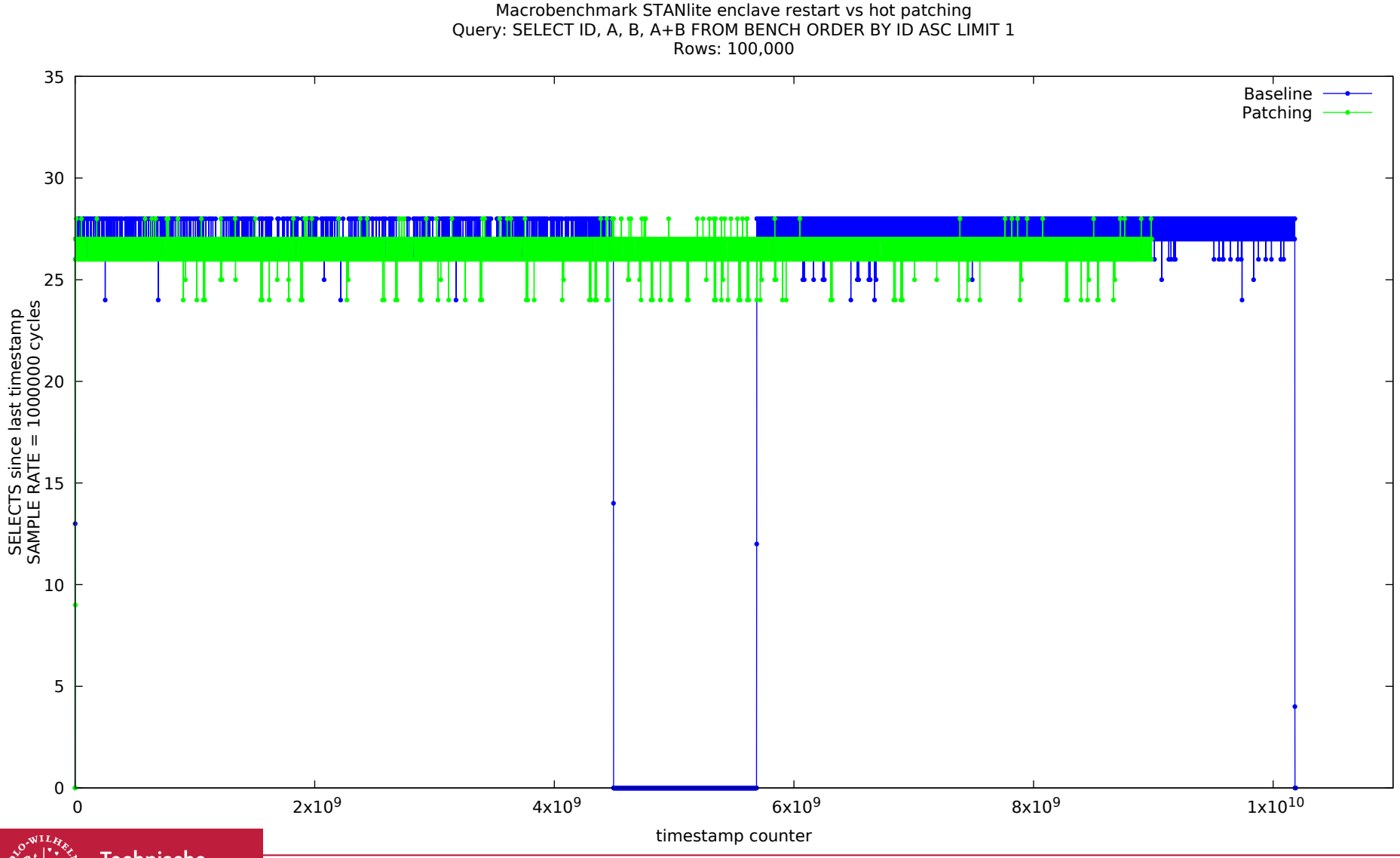
Microbechmark Hot-Patching/Calling Overhead



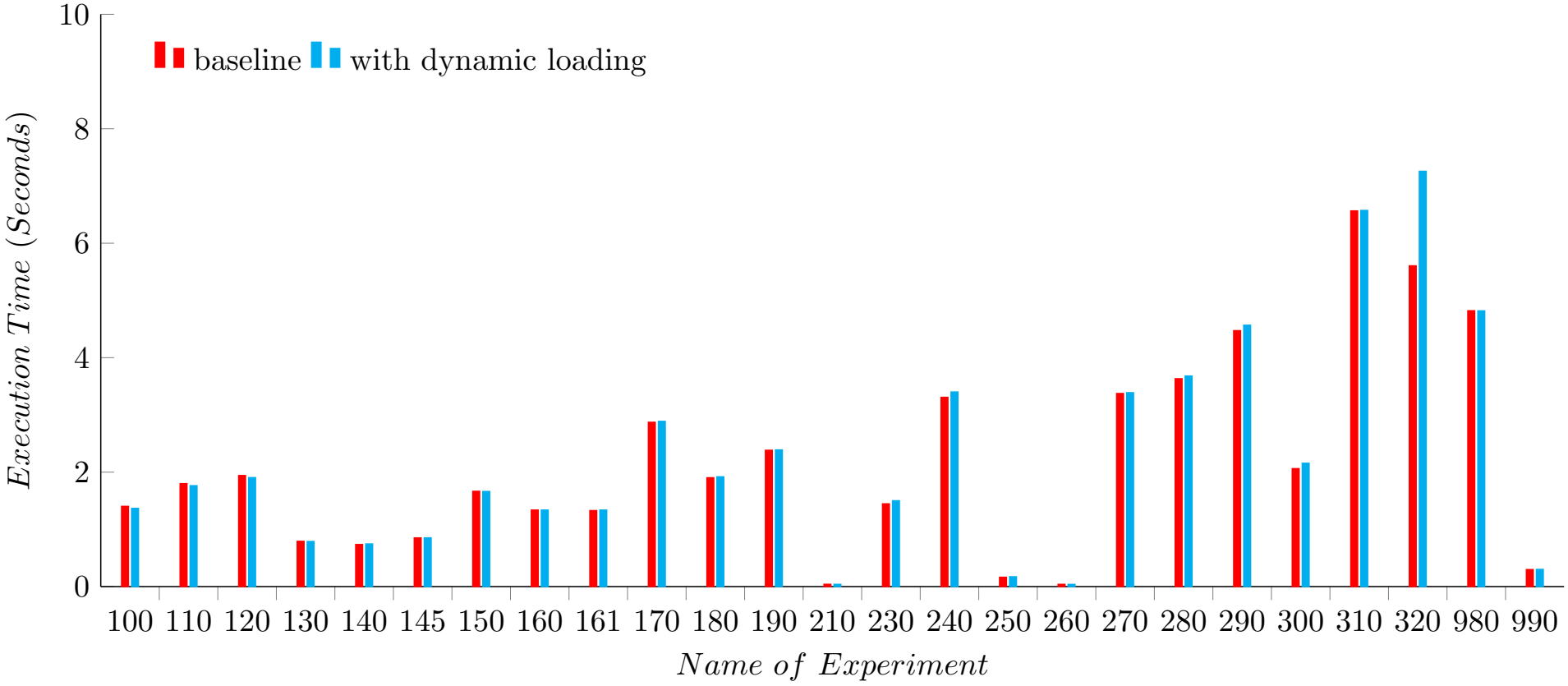
Macrobenchmark

- Small enclave (fits EPC)
- STANlite (SQLite) with dynamic functions in SQL-VM
  - add/sub/mul/div/mod
  - No changes made to state machine except argument un-/wrapping
- State is already encrypted in untrusted memory, so no state saving needed, just additional code to set the pointers right
  - This is actually bad for us, as it reduces the benefit of hot-patching
- Enclave restart time: 0.8 s

Macrobenchmark Hot-Patching/Calling Overhead



Macrobenchmark Speedtest2 (small dataset)



Adapted from STANlite

Next steps

- Publish sgx-dl
  - Current conference target: EuroSys 2020, deadline November
- Nicer benchmarks/more applications
  - Enclaves that don't fit EPC, make more functions of SQLite dynamic
- Integrate with a DBMS
  - e.g., MonetDB

- Next Round
  - Distributed Secure Computing
    - Intel SGX Card (3x E3-1585L v5 with PCIe interconnect)
  - SGX multi-socket servers
  - RDMA + SGX + DPDK or similar