UiT

THE ARCTIC UNIVERSITY OF NORWAY

Trusted Computing on Privacy Sensitive Data with Diggi

Anders Gjerdrum[1], Robert Pettersen[1], Håvard D. Johansen[1], Robbert van Renesse[2], and Dag Johansen[1]

[1]UIT: The Arctic University of Norway, Tromsø, Norway. [2]Cornell University, Ithaca, NY, USA.

/ MOTIVATION

- # of connected devices grow exponentially.
  - Generate and process personal and privacy sensitive data.
  - Increasing computational abilities.
- Host data outside primary domain.
  - Reduce cost (latency, bandwidth, compute) by moving computations closer to edge.

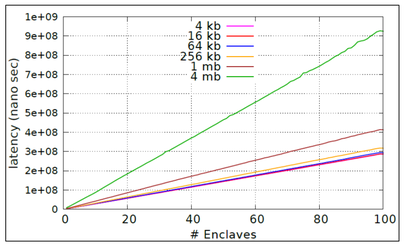
/ TRUSTED COMPUTING

- Confidentiality, integrity and authentication.
- Intel Software Guard Extensions (SGX)
  - Enables secure segments of code and data (enclaves) to run on untrusted platforms.
  - Minimizes trusted computing base.
- ARM TrustZone
  - Trusted Execution Environment(TEE).
  - For mobile devices.
  - Separate Trusted OS.
  - Physically isolated by the Memory Management Unit and bus architecture.

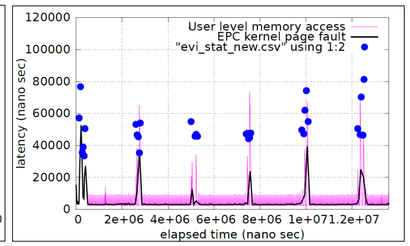
/ EXPERIMENTS

- Diggi architecture based on cost of SGX primitives.
- Quantify cost of provisioning, memory footprint, context switches and multithreading.
- Experimental setup
  - i5-6500 @3.20 GHz w/ 4 logical cores
  - 28 GB DDR3 DIMM DRAM.
  - Ubuntu 14.04
  - Instrumented Kernel driver
  - RDTSC instruction not available→ measurements include cost of entry-exit.

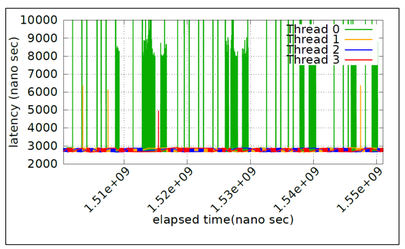
/ PROVISIONING



/ MEMORY FOOTPRINT



/ SINGLE CORE PAGEFAULT



/ MULTI CORE PAGEFAULT

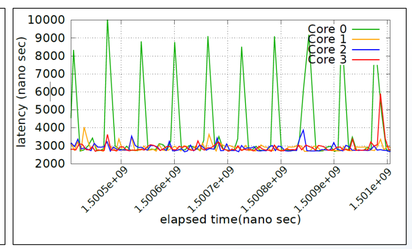


Figure 1: (Top-Left) Portioning latency per enclave. (Top-Right) Page fault overhead observed from kernel and user level measurements. (Bottom-Left) Page fault overhead observed in enclave by single core with multiple threads. (Bottom-Right) Page fault overhead observed in enclave by multi core with single thread.

/ IMPLICATIONS

- Keep enclaves small(< 64kb).
- Page fault handler prefetching scheme and Processor Reserved Memory exhaustion.
- Pre-provision enclaves.
- Pin threads to enclaves.
- Single thread per logical core.
- Asynchronous execution inside enclaves.

/ TRADE-OFF

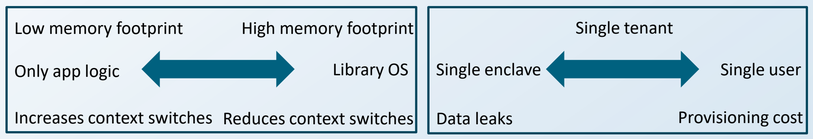


Figure 2: . (Left) Memory footprint. (Right) Isolation granularity

/ DIGGI AGENT ARCHITECTURE

- Distributed runtime for data storage/processing.
- Trusted Computing enables privacy and integrity on untrusted third-party platforms.
- Execution of service graphs optimized by placing processing entities, "**agents**", close to subject data.
- Built from the ground up for trusted computing.
  - Fully asynchronous execution environment.
  - Secure communication with program attestation.
  - Highly configurable.
  - Affinized, per enclave, thread scheduler.
  - Primitives built for low memory footprint.

```
void agent_run(ring_buffer_t *input_q,
                        ring_buffer_t *output_q)
{
        event_schedule(read_input, input_q);
        event_schedule(write_output, output_q);
        /*
                Application Logic
        */
        event_schedule(agent_run, input_q, output_q);
}
```

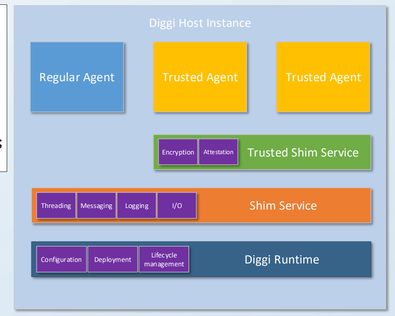


Figure 3: . (Left) Example logic of an asynchronous Diggi agent flow. (Right) High level overview of the Diggi agent host architecture.

Figure 4: (Right-Top) Backend shipping processing agent to untrusted mobile device (Left-Bottom) Mobile device shipping agent to untrusted backend