Safely and Efficiently Programming a 64kB Computer

Amit Levy\textsuperscript{a}, Branden Ghen\textsuperscript{b}, Bradford Campbell\textsuperscript{b}, Pat Pannuto\textsuperscript{b}, Prabal Dutta\textsuperscript{b}, Philip Levis\textsuperscript{a}

MSR UW Summer Institute
August 2, 2017
\textsuperscript{a}Stanford University  \textsuperscript{b}University of Michigan
Securing the Internet of Things

- Secure Internet of Things Project
  - 5 year project (just started second year)
  - 12 faculty collaborators
  - 3 universities: Stanford, Berkeley, and Michigan
- Rethink IoT systems, software, and applications from the ground up
- Make a secure IoT application as easy as a modern web application
There’s no such thing as a secure embedded OS today.
There’s no such thing as a secure embedded OS today.

Let’s research why and write one.
Embedded Systems

“An embedded system is a computerized system that is purpose-built for its application.”

Elicia White
Making Embedded Systems, O’Reilly
But the World is Changing…

“An embedded system is a computerized system that is purpose-built for its application.”

Elicia White
Making Embedded Systems, O’Reilly
But the World is Changing…

“An embedded system is a computerized system that is purpose-built for its application.”

Elicia White
Making Embedded Systems, O’Reilly
But the World is Changing…

“An embedded system is a computerized system that is purpose-built for its application.”

Elicia White
Making Embedded Systems, O’Reilly

A new class of embedded devices, that act as platforms supporting loadable programs within a particular application domain.
Tock Operating System

- Safe, multi-tasking operating system for memory-constrained devices
- Core kernel written in Rust, a safe systems language
  - Small amount of trusted code (can do unsafe things)
    - Rust bindings for memory-mapped I/O
    - Core scheduler, context switches
- Core kernel can be extended with capsules
  - Safe, written in Rust
  - Run inside kernel
- Processes can be written in any language (asm, C)
  - Leverage Cortex-M memory protection unit (MPU)
  - User-level, traps to kernel with system calls
Tock Architecture

Process (Any language)
- heap
- stack
- data
- text

Kernel (Rust)
- SPI
- I2C
- UART
- Console
- GPIO
- Timer

Core kernel (Trusted)
- HAL
- Scheduler
- Config

Capsules (Untrusted)
- Process Accessible Memory
- RAM
- Flash

Heap
- Text
- Stack
- Data
- Grant
Challenge: System Calls

- System calls need to dynamically allocate memory
  - Create a timer, kernel needs to keep timer’s state
  - Enqueue a packet to send, kernel needs reference to packet
- For dependability, kernel has no heap
  - Otherwise a process can exhaust kernel memory
  - Fragmentation
  - Cleaning up after process failures
- How does the kernel handle system calls if it has no heap?
System Call Insight

- Processes given block of memory
- Dynamically allocated when process loaded
- Kernel can allocate memory from process

Processes (Any language)

Kernel (Rust)

Capsules (Untrusted)

Core kernel (Trusted)
Memory Grants

- Each process has a growable container of *grant memory*
- Kernel can allocate objects from the grant block
- References to objects cannot escape the block
  - Process failure/crash does not lead to dangling pointers
- Users pass a function to the container with `enter`

self.apps.enter(appid, |app, _| {  
  app.read_buffer = Some(slice);  
  app.read_idx = 0;  
  0  
}).unwrap_or(-1)
Programs to the Edge

- Firmware
- WWW
- lua
- python
- C

Application programming model
Application-to-application authentication
Network security policies
20-year cryptography/software update
Tock Operating System

- Safe, multi-tasking operating system for memory-constrained devices
- Core kernel written in Rust, a safe systems language
  - Small amount of trusted code (can do unsafe things)
    - Rust bindings for memory-mapped I/O
    - Core scheduler, context switches
- Many new system design and research challenges
  - Writing a kernel in a type safe, not garbage collected language
  - Memory isolation and allocation
- Come learn how to use it!
Thanks!

https://www.tockos.org/
Amit Levy <levya@cs.stanford.edu>

↑ Amit will be on the job market this year - help me make him smile!