seL4 Present and Future
@GernotHeiser & Team
NICTA and UNSW Australia
What is seL4?

seL4: The latest (and most advanced) member of the L4 microkernel family – 20 years of history and experience
What is seL4?

seL4: The world’s most (only?) secure OS kernel – provably!

GPLed
2014-07-29
seL4: Mathematical *Proof of Security*

**Confidentiality**
- No buffer overflow, un-init vars, NUL-deref, stack smashing, code injection, ROP…

**Integrity**
- Functional correctness
- World First!

**Availability**
- Isolation properties
  - World First!

**Abstract Model**
- Proof

**C Implementation**
- Translation correctness
  - World First!
- Proof

**Binary code**
- Proof

**Exclusions (at present):**
- Initialisation
- Privileged state & caches
- Multicore
- Covert timing channels

**Timeliness**
- World First!

**No need to trust C compiler!**
- Real upper bounds for IRQ latencies
- No buffer overflow, un-init vars, NUL-deref, stack smashing, code injection, ROP…
What seL4 is NOT: An Operating System

All device drivers, OS services are usermode processes

seL4 microkernel

Processor

IPC

VM

Linux

App

IPC

App

File System
NW Stack
Device Driver
Process Mgmt
Memory Mgmt

Strong Isolation

Controlled Communication

©2015 Gernot Heiser, NICTA
What’s Different to Other L4 Microkernels?

Design for isolation: No memory allocation in the kernel

Resources fully delegated, allows autonomous operation

Strong isolation, No shared kernel resources
High-Assurance System on seL4

DARPA HACMS Program:
- Provable vehicle safety
- “Red Team” must not be able to divert vehicle

Boeing Unmanned Little Bird (AH-6) Deployment Vehicle

SMACCMcopter Research Vehicle
Architecting System-Level Security/Safety

- Architecture Specification
  - Requirements (specific set of security/safety properties)
  - Component Model
    - Untrusted
    - Trusted

- Automatic Analysis
  - (Requirements fulfilled)

- Automatic Generation of Glue code

- Component Implementations
  - Verified Glue Code
    - Communication
    - Init

- seL4 Kernel
  - Correctness
    - Formal proof
    - Synthesis
  - Security
  - Functional correctness
  - Glue Code Proof
    - Verified Proof

- Correctness
  - Formal proof
  - Synthesis

- Glue Code Proof
  - Verified Proof

- seL4 Proof
  - Functional correctness
  - Security

- Verified Glue Code
  - trusted
  - Untrusted

- seL4 Kernel
  - trusted
  - Untrusted
Current NICTA Work on seL4

- High-performance multicore support
  - Release ETA: few months (ARM, x86)

- Full support for virtualisation extensions
  - Release ETA: few months (ARM, x86)

- 64-bit support
  - Release ETA: few month (x86), ??? (ARM64)

- Mechanisms for eliminating timing channels
  - ETA: 2015 (ARM and x86)

- Temporal isolation and mixed-criticality scheduling
  - ETA: 2015 (ARM and x86)

- Hardware failure resilience (DMR/TMR on multicore)
  - ETA: 2015 (ARM and x86)
What Else Is Cooking?

- **Aim:** Cost reduction by automation and abstraction
  - Present seL4 cost: $400/SLOC, high-assurance, high-performance
  - Other “high” assurance: $1,000/SLOC, no proof, poor performance
  - Low assurance (Pistachio): $200/SLOC, no proof, high performance

- **Device driver synthesis**
  - Synthesise driver code from hardware and OS interface specs
  - works already for simple devices

- **Code and proof co-generation**
  - High-level spec in DSL describes logic, generate C code and proofs
  - File systems as case study

- **Type- and memory-safe high-level languages**
  - Do verification cheaper in HLL semantics
  - Requires verified HLL run-time and compilers
seL4 Ecosystem: Kernel Development

Private branch → release → seL4 experimental (Commitment to verify! (Eventually…)) → Proofs → seL4 master

Public, Github, GPLv2
How Can YOU Contribute?

• Libraries presently extremely rudimentary
  – POSIX! ...

• Platform ports
  – Especially popular ARM boards: Tegra, RK3188, Beaglebone, ...

• Drivers!!!!!!
  – Very few available ATM

• Network stacks and file systems
  – Presently have lwIP, incomplete functionality

• Tools
  – Have component system (CAmkES), glue generators

• Languages
  – Core C++ support just released, lacks std template lib
  – Haskell presently in progress (with Galois) – stay tuned
  – Python would be awesome!
Why NOT Use seL4?

• Very rudimentary programming environment!
  – Fair enough
  – You can help to fix this!
• I like unsafe/insecure systems!
  – Ok, go shoot yourself
• I like the thrill of danger!
  – Like getting sued for building a critical system on outdated technology
• Actually, I want to use seL4!
  – Right answer ;-)

http://seL4.systems

gernot@nicta.com.au

http://microkerneldude.wordpress.com

@GernotHeiser