OS - Retrospection

Tid-bits from course outline

This course is oriented towards exposing students to the essential concepts and issues that underly operating systems and their design.

- **Technical**
  - Make students understand the key concepts and mechanisms of modern operating systems:
    - processes and process management,
    - memory management techniques,
    - on-line storage methods (file systems),
    - concurrency issues,
- **Educational**
  - Make students understand the reasons why operating systems are built the way they are, and what the implications and lessons are for other software systems. Specific learning objectives are:
    - appreciation of design trade-offs and design decisions and their dependence on the target environment;
    - exposure to low-level code;
    - exposure to current trends in operating systems research and development.
- **Professional**
  - The tutorial formats will give students practice in the presentation of solutions to an audience of peers, and will challenge them to critique peer technical presentations. Furthermore, the whole course encourages critical examination and analysis of "standard" solutions.
  - The assignments give students an opportunity to develop skills required to work as a team on a technical project, and the opportunity to work with a substantial body of code created by a third-party.

Operating Systems @ CSE.UNSW

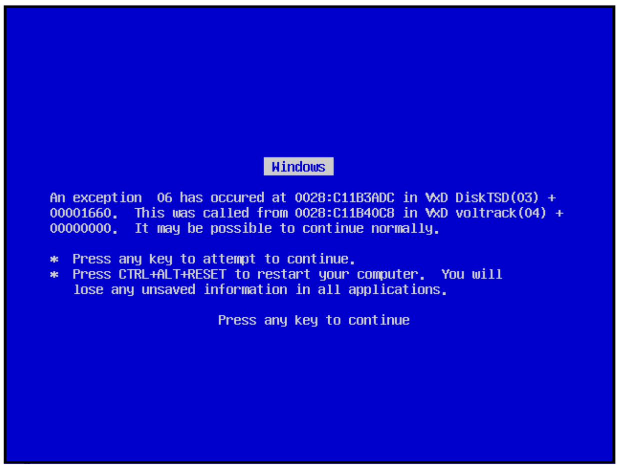
Systems Courses

- COMP9242 Advanced Operating Systems
  - In-depth coverage of OS implementation issues
  - Learn more about what makes OS fast and what makes them slow
  - Learn how the OS deals with multiprocessors, caches, virtualisation, etc, etc....
  - Write your own OS on a microkernel
- In Session 2 taught by Prof. Gernot Heiser and Assoc. Prof. Kevin Elphinstone

- Distributed systems COMP9243 (Session 1 2018)
  - Examines issues in building distributed systems and infrastructure
  - Peer-to-peer, web services, network file systems, name services, ……

OS Research Trustworthy Systems Group, Data61

http://ts.data61.csiro.au/
- 10-ish researchers (PhDs)
- 10-ish research engineers / research assistants
- 10-ish PhD students


```
Windows

An exception  06 has occured at 0028:C11B3ADC in VxD DiskTSD(03) +
00001660.  This was called from 0028:C11B40C8 in VxD voltrack(04) +
00000000.  It may be possible to continue normally.

*  Press any key to attempt to continue.
*  Press CTRL+ALT+RESET to restart your computer.  You will
   lose any unsaved information in all applications.

                    Press any key to continue
```

The Problem

THE UNIVERSITY OF NEW SOUTH WALES

Embedded System Scenarios

- Increasing usability requirements
  - Patient-operated (wearable) medical devices
  - GUIs next to life-critical functionality
- On-going integration of critical and entertainment functions
  - Automotive infotainment and car control
  - Mutually untrusted SW vendors
- Cost pressure
  - COTS devices for national security use
- No longer closed systems
  - Download SW

THE UNIVERSITY OF NEW SOUTH WALES 9

Embedded Systems Software

Present Approaches 1: Real-time Executives

- Small, simple operating system
  - optimised for fast real-time response
  - suitable for systems with very limited functionality
- No internal protection
  - every small bug/failure is fatal
  - no defence against viruses, limited defence against crackers

THE UNIVERSITY OF NEW SOUTH WALES 10

Embedded Systems Software

Present Approaches 2: Linux, Windows Embedded

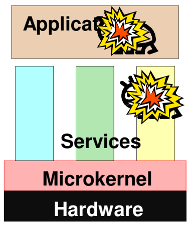
- Scaled-down version of desktop operating system
  - operating system protected from application misbehaviour
  - excessive code base for small embedded system
  - too much code on which security of system is dependent
- Dubious or non-existent real-time capabilities
  - unsuitable for hard real-time systems

THE UNIVERSITY OF NEW SOUTH WALES 11

Embedded Systems Software

Our Approach: Microkernels

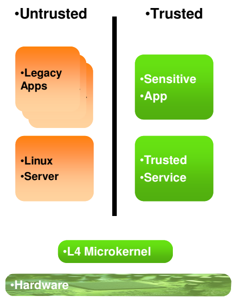
- Extremely small kernel
  - microkernel only contains code that must run in privileged mode
  - all other "systems" code runs as unprivileged servers
  - microkernel protected from application and other systems code
  - microkernel provides protection of all components from each other
  - services can be restarted



THE UNIVERSITY OF NEW SOUTH WALES 12

Microkernel Approach – L4

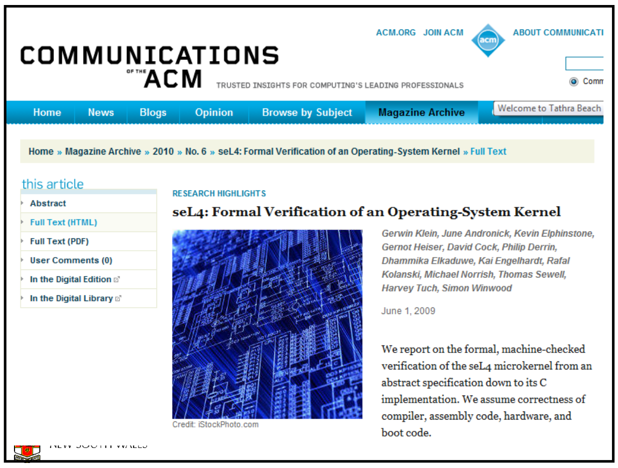
- Small trustworthy foundation
  - Applications:
    - Fault isolation
    - Fault identification
    - IP protection
    - Modularity
    - ...
  - High assurance components in presence of other components
- Provides a trustworthy foundation



L4 (UNSW/NICTA/Data61) Impact

- Licensed to OK-labs
  - NICTA spinout
  - L4 on >1500 million Handsets
    - Including Android Phones, Windows Phone, iPhone (security processor)
  - Acquired by General Dynamics

Open Kernel Labs™
Be open. Be safe.



ACM.ORG JOIN ACM ABOUT COMMUNICATI

COMMUNICATIONS OF THE ACM
TRUSTED INSIGHTS FOR COMPUTING'S LEADING PROFESSIONALS

Home News Blogs Opinion Browse by Subject Magazine Archive

Welcome to Tathra Beach

Home » Magazine Archive » 2010 » No. 6 » seL4: Formal Verification of an Operating-System Kernel » Full Text

this article
- Abstract
- Full Text (HTML)
- Full Text (PDF)
- User Comments (0)
- In the Digital Edition
- In the Digital Library

RESEARCH HIGHLIGHTS

seL4: Formal Verification of an Operating-System Kernel

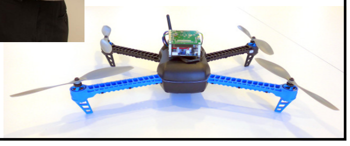
Gerwin Klein, June Andronick, Kevin Elphinstone, Gernot Heiser, David Cock, Philip Derrin, Dhammika Elkaduwe, Kai Engelhardt, Rafal Kolanski, Michael Norrish, Thomas Sewell, Harvey Tuch, Simon Winwood

June 1, 2009

Credit: iStockPhoto.com

We report on the formal, machine-checked verification of the seL4 microkernel from an abstract specification down to its C implementation. We assume correctness of compiler, assembly code, hardware, and boot code.

Current Projects



Why am I telling you this?

Does the following Interest you?

- Gaining in-depth experience in OS research
- Working on a very challenging projects
- Collaborating closely with active researchers
- Getting a high thesis mark
- International travel
- Fame and fortune

Prerequisites

- Keen interest in OS
- Demonstrable background/ability in OS
- Sharp Intellect
- Committed to working on a project

Still Interested?

- Check out

http://ts.data61.csiro.au
specifically the education section, and the student section
http://ts.data61.csiro.au/students

Apply for a Taste of Research Summer Scholarship
https://www.engineering.unsw.edu.au/study-with-us/scholarships/taste-of-research-summer-scholarships

On-line Course Surveys

- The on-line course survey will be available
  - My one – in addition to UNSW one
- Please make time to do it
- Award 2 bonus class marks to everyone who completes ***my*** survey.
  - You will be emailed an invite

Final Exam

- Separate papers for OS (3231/9201) and Extended OS (3891/9283)
- Sat, 24th June, 9:00
- Two Hours
- No examination materials allowed
  - Uni approved calculators okay
- Don't trust me – check the timetable yourself

Exam Format

- Read the instructions on the exam
  - The following details are approximate (read the exam instructions on the day)
- 5 questions
  - 3 should be answered in separate books
  - 1 must be ***answered on the exam paper*** itself.
  - 1 must be answered on the multiple choice answer sheet provided
  - 100-ish Marks in total (total will be scaled to 100)
  - 2 marks for following exam instructions

Exam Format

- Q1 is true/false choice (40% marks)

  You will receive one mark for each correct classification, and lose one mark for each incorrect classification. You gain zero marks for each answer left unclassified. The overall mark for this question will not be negative, i.e. the minimum mark is zero.
- Intended to be hard!
  - Some questions are tricky, and may appear ambiguous if you don't know material.

Exam Format

- Q2..Q5, roughly:
  - half working out a solution to a problem
  - half written answers to a question

For written answers

- Be clear and concise (get to the point quickly)
  - Long, rambling answers will be penalised

Sample Question

- What are the four conditions required for deadlock to occur? For each condition, state a method of deadlock prevention that prevents the condition occurring, if such a method exists.

- Sample Marking Scheme (out of 8)
  - 2 Marks for each condition (1 for the condition, 1 for the prevention with "why/understanding")

Reasonable answer

- Mutual exclusion
  - The need for mutual exclusion cant be avoided as it would introduce race conditions.
- No Preemption
  - Preemption of critical sections is not possible while retaining correctness – locks/resource need to be used for the duration of the critical section.
- Hold and wait
  - This can be prevented by never holding resources if waiting is required, i.e. locks/resources that are held are released prior to waiting
  - This can live-lock, i.e. not guarantee that all required locks are ever acquired.
- Circular wait
  - This can be practically prevented numerically ordering all resource/locks and always acquiring them in numerical order. It prevent a thread that has a higher number resource ever waiting on a lower numbered resource, thus prevent circular waiting.

Poor answers

- FIFO, Threads, Locks and Scheduling
  - Don't just as add names of acronyms you can remember

Poor answers

- Deadlock is where the computer stops. Four conditions are required for deadlock to occur. CPU must be running, locks are required, and one lock must need another lock, and more than one thread is a condition as well.
- Stopping the CPU is not a feasible condition.
- We can't avoid locks as well
- We can stop one lock from acquiring other locks to prevent deadlock
- We can prevent deadlock by only running one program at a time. It prevents the more than one thread condition.

Answer the question!!!

- Don't repeat the question, we set the exam, we know what it is!!!!
- Don't just write what you know (or don't know) about the topic area
  - You make us have to search for the real answer.
  - You may be correct, but say a lot of unrelated incorrect stuff in the process.
- Don't contradict yourself
  - X is better/faster/more efficient than Y, and later Y is better than X
- Marks are awarded for stating WHY an answer is correct.
  - Demonstrates understanding

Exam Content

- For structure and style, look at the sample exam from past years.
- For content, the tutorial questions are a reasonable *guide*.
- Will be releasing 100-ish sample questions (with student answers).
  - Will also answer questions on the forum
    - sometimes difficult to answer without a whiteboard

The questions attempt to examine understanding rather than particular implementations

- Don't expect
  - "Describe OS/161's exception handling on a timer interrupt"
- But you may get
  - "Describe (in general) a feasible sequence of steps that occur in response to a timer interrupt that results in the current process being pre-empted and another process running"

Examinable Content

- All Lectures, Tutorials, Assignments.
- More specifically
  - Anything related to learning outcomes