Wednesday, 28 April 2021
Available from 14:00 BST
Expected Duration: 2 hours
Time Allowed: 4 hours
Timed exam within 24 hours

DEGREES OF MSc, MSci, MEng, BEng, BSc, MA and MA (Social Sciences)

ADVANCED SYSTEMS PROGRAMMING (H) COMPSCI 4089

Answer 3 out of 4 questions

This examination paper is an open book, online assessment and is worth a total of 60 marks.

1. **(a)** A common approach to concurrent programming is to use multiple threads of execution, with locks protecting concurrent access to shared state. As an alternative, some languages offer support for concurrency via transactions with automatic roll-back and retry, while others support a concurrent actor model with message passing communications. Discuss the advantages and disadvantages of these approaches, when applied to developing operating systems code and device drivers. State which you think is most suitable for this problem domain. Justify your answer. [12]

   (b) Message passing systems with synchronous communication force the message sender and receiver to rendezvous to exchange a message. In asynchronous message passing systems, the sender continues after sending the message, that is buffered for later delivery. Discuss how these two different approaches affect the concurrency and reliability of a message passing system. [8]

2. **(a)** Operating systems typically organise the virtual address space used by a process so that the memory used by the stack is separate to the memory used by the heap. Outline what data is stored on the stack and what is stored on the heap. Explain why the stack and the heap need to be separate regions of memory. [6]

   (b) The C programming language provides low-level control over data layout, allowing the programmer to precisely control how data structures are represented in memory. It is also not a memory safe language, permitting arbitrary pointer arithmetic and casts between data types, and allowing a program to interpret a region of memory as many different types. Considering the implementation of operating systems and device drivers, highlight areas where this combination of precise control over data layout and weak type checking is a strength of C, and areas where it is a weakness. Discuss whether you believe C makes the right trade-off between safety and control. [10]

   (c) The Rust programming language provides `unsafe` blocks, that allow certain aspects of the type system to be circumvented. One of these aspects is the ability to dereference raw pointers, that can provide unsafe memory access. Discuss why Rust provides for unsafe memory access in this way, and whether the presence of this feature indicates a problem with the language. [4]

3. **(a)** Studies have shown that most security vulnerabilities in deployed systems are due to a lack of memory safety. The lectures described the classic, stack-smashing, buffer overflow attack as an example of a memory safety vulnerability. Briefly describe how this buffer overflow attack works. [3]

   (b) Two mitigations for the classic buffer overflow attack are to randomise the location of the top of the stack, and to randomise the addresses at which shared libraries are loaded. Explain how these changes reduce the effectiveness of buffer overflow attacks, and why such mitigations are more effective on 64-bit systems than on 32-bit systems. [3]

   (c) Safely parsing protocol data units is one of the key challenges in building secure networked systems. A long-standing principle when designing parsing code in such systems is expressed in Postel's law, that states "Be liberal in what you accept, and conservative in what you send". Discuss to what extent Postel's law is still applicable to modern networked systems. [8]

(d) In addition to memory safety, a key benefit of modern systems programming languages is that they have expressive type systems, capable of modelling features of the problem domain. Discuss to what extent the principle of type-driven development can help to improve the security of networked systems, giving examples to illustrate such approaches. [6]

4. **(a)** It has been suggested that the C programming language is no longer appropriate for writing systems code, and that new programming languages can improve systems programming. Discuss the extent to which you believe that changing the programming language will help to address the challenges in building secure, high performance, and robust systems programs, and to what extent the challenges of building systems programs are inherent in the problem domain. Give examples to show what features of modern programming languages are beneficial for systems programs, and what are harmful. [12]

(b) Briefly describe what is meant by a *type-driven development* approach to designing and implementing programs. Discuss whether the level of abstraction of operating systems and device driver code is such that type-driven approaches to development are likely to offer benefits compared to more traditional approaches, or will result in excessive cognitive overhead in designing types. [8]