IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2018-2019

MEng Honours Degree in Mathematics and Computer Science Part IV
MEng Honours Degrees in Computing Part IV
MSc in Advanced Computing
MSc in Computing Science (Specialist)
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute

PAPER C410

SCALABLE SYSTEMS FOR THE CLOUD

Tuesday 11th December 2018, 10:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators required

Section A (Use a separate answer book for this section)

1 a Briefly explain each of these terms and state where they are used:

i) trap-and-emulate

ii) dynamic binary translation

iii) shadow page tables

iv) power usage effectiveness.

b Efficient datacentre networking requires support across the system stack.

i) Compare a traditional network model with a software defined network (SDN) model.

ii) Draw the SDN architecture and explain the different roles of the layers (planes) and the interface(s) between them.

iii) List two types of hardware support for network I/O virtualization. Explain what each of them does and discuss the benefits of using it.

iv) Assume that for an application running in a virtual machine it is important to have efficient communication with its clients over the network. What are the main bottlenecks for packet processing in a traditional network I/O stack? What kind of hardware/library support would help you optimise the I/O performance? Explain if any modifications in the application code are needed.

The two parts carry, respectively, 25% and 75% of the marks.

2 a Briefly answer the following questions.

i) What is the difference between server- and resource-centric datacenters?

ii) What are the key properties of Virtual Machines?

iii) What are the roles of a Virtual Machine Monitor (VMM) and a hypervisor?

iv) What are the restrictions (limitations) that one needs to account for when using Serverless (Function as a Service)?

b Please provide more detailed answers to the following questions.

i) Describe the difference between hardware- and OS-level virtualization. List the advantages and disadvantages of using one over the other.

ii) What techniques do containers use to achieve isolation? Please provide a one-sentence explanation for each technique.

iii) In class we covered a few key virtualization techniques: (1) hardware support, (2) para-virtualization, and (3) binary translation. Which one of them can most easily adapt to changes in the underlying hardware, and which one the hardest? Explain your reasoning.

iv) Draw the architecture of Kubernetes and describe each component. Explain why and how Kubernetes uses Pods to manage a group of containers.

The two parts carry, respectively, 25% and 75% of the marks.

Section B (Use a separate answer book for this section.)

3 a Briefly explain each of the following concepts, and give an example where each could be applied in the context of scalable distributed systems.

i) Commit log

ii) Service level agreement

iii) Combiner function

iv) Job scheduler

b The *ZooKeeper* system provides the following client API:

```
create(path, data, flags)
delete(path, version)
exists(path, watch)
getData(path, watch)
setData(path, data, version)
getChildren(path, watch)
sync(path)
```

i) Briefly explain the semantics of each API call and its parameters.

ii) Describe the simplest way in which you could implement a mutual exclusion lock using the above API.

iii) When you have a large number of clients competing for the same lock, explain the *performance issue* that your simple lock implementation would suffer from.

iv) Describe, using pseudo code, a more sophisticated lock implementation using the ZooKeeper API that does not suffer from the issue identified under iii) above.

v) Explain how you could extend your solution under iv) to support lock acquisition by clients that have different *priorities*. A client with a higher priority should always acquire a lock before a client with a lower priority.

The two parts carry, respectively, 20% and 80% of the marks.

4 You work as a software engineer for FBAY, an e-commerce company that offers a trading platform for users to sell and buy goods. Your boss would like to record all transactions that have occurred on the platform as part of an *immutable log*.

You are tasked with proposing the design of a new system that creates and maintains such an immutable log. The immutable log should be stored on disk. The log should record various information about a transaction, such as a timestamp, the identities of the seller and the buyer, a description of the item, and the purchase price. The system must support a client API to issue (i) read queries for past entires in the log and (ii) write queries to append new entries to the log.

Entries in the log should be immutable by forming a *hash chain*: each entry should include the hash of the previous entry and be cryptically signed to detect changes.

You should propose the design of a new system that maintains such an immutable log. Your proposed design should satisfy the following requirements:

(R1) The system must handle a large number of concurrent transactions that query existing entries in the immutable log and add new entries to the log.

(R2) The system must support arbitrary growth of the log over time. Assume that the log may consume petabytes of storage.

(R3) The system must be incrementally scalable.

(R4) The design must be fault tolerant.

(You should make justified decisions about any aspects that are left unspecified.)

a Describe the design of a system that satisfies the above requirements.

Draw a diagram of your system design, clearly labelling all distributed components. For each component, explain its operation and justify its function.

b Explain the steps that your system takes to execute (i) a *read* query and (b) a *write* query.

c For each of the requirements, (R1)–(R4), explain how your design achieves it.

d Discuss any bottlenecks that exist in your design that may limit the query throughput.

The four parts carry, respectively, 40% 20%, 20%, and 20% of the marks.