CIS 508 (Spring 2014) Assignment 1

Name: Yanan Xiao

Masdar Email: yxiao@masdar.ac.ae

Deadline: **30 Jan 2014, 11:59pm (Late submissions of assignments will not be marked)**

Submission Instructions:

1. Fill in this document
2. Save as a pdf file, with filename as “cis508-2014-asg1-[MasdarEmailUserID].pdf” (replace [MasdarEmailUserID] by your own Masdar Email User ID)
3. Send as an attachment to sidckchau@gmail.com, with email subject as “[CIS 508] Assignment 1 (2014)” before the above deadline

Questions:

Part 1:

Xen is a brainchild from computer laboratory, University of Cambridge (led by Ian Pratt). Xen realizes the idea of paravirtualization.

1. What is paravirtualization? What are the differences between paravirtualization and other former virtualization technology? (2 Marks)

Definition: Paravirtualization works by utilizing modules in system kernel and provides virtualization support for computers that do not support hardware-assisted virtualization, like Intel-VT and AMD-V.

Differences: Paravirtualization does not emulate a whole set of computer hardware and its instructions. It runs a control domain on a hypervisor, which runs on top of hardware, and uses PV backend (in control domain) PV frontend (in guest OS) to communicate. With both the frontend and backend support, guests can collaborate in virtualization thus reduing the overhead when compared with running full virtualization.

2. Why the development of Xen is important in the context of operating systems? (2 Marks)

Systems: The host systems compatible with Xen is diversified. This means that Xen can be configured smoothly without worrying about specific hardware.

Architecture: Xen supports multiple system architectures like IA-32, X86 and even ARM, so it is highly scalable with respect to deployment. Almost all kinds of operating systems can be run as guest os.

Virtualization: The support for paravirtualization, hardware-assisted virtualization and even the old full virtualization makes Xen even more versatile. And the invention of Paravirtualization speeds up the guest OS for orders of magnitude.

3. Xen was acquired by Citrix in 2007 for US$500M. Why did Citrix buy Xen? Based on the present observations, was the acquisition worthwhile? (2 Marks)

Reason: In 2012, the annual revenue for Citrix is $2.56 billion. So even in the 2007 senario, Citrix can afford a $500 million offering to buy a technology-intensive company like Xen. Citrix's main “responsibility” is to empower people to work and collaborate everywhere, thus a reliable and virtualized backend, in this case, Xen serves becomes very important. In purchasing rather than developing virtualization technologies from scratch, Citrix can greatly improve its service quality in no time.

Justification: By acquiring a company like Xen which is of high technology, a $500 million offering may seem a bit high in the beginning. But by actually deploying this technology and still making it open source, Citrix could gain a long term growth (as reflected by annual report) and good reputation among open source community.

Part 2:

PlanetLab was conceived as a new breed of distributed computer systems.

4. Explain the concept of PlanetLab? Compare the similarities and differences between PlanetLab to other similar systems, like Emulab, Orbit, GENI (2 Marks)

Concept: PlanetLab could be regarded as the prototype of modern cloud computing. Briefly, it's a geographically distributed network service using overlay-network.

Similarities: They are all distributed network systems and mainly used for scientific research.

Differences: Unlike other “similar” systems, PlanetLab is more than a distributed system providing more computing power. It also serves as an overlay-network testbed for researchers to experiment new services. Moreover, there are many services that can be seen on today's Amazon Web Service, like CDN and DHT. To conclude, PlanetLab has implemented more functionalities.

5. What are the applications that can run on PlanetLab? Why are advantages for running those applications on PlanetLab?(2 Marks)

Applications: As described in the former question, PlanetLab and the like are the pioneers of today's cloud computing service. It could also be regarded as another form of virtualization. Therefore, broadly speaking, almost all kinds of services can be run on systems like this. But the mainstream applications are network-related. Like CDN (content distributed networks), DHT (distributed hash table) and network measurement services like ScriptRoute.

Advantages: One of the most important features are scalability. By running services like this, computing and storage resources could be allocated dynamically and intelligently than traditional manners. The other big advantage when using distributed systems is redundancy. This feature is almost enabled “by nature” in distributed systems with a simple setup.

Part 3:

Consider three transactions (T1, T2, T3) as follows:

T1	T2	T3
ReadLock(B) B_ = Read(B) B_ = B_ + 100 ReadLock(A) A_ = Read(A) A_ = A_ - 100 WriteLock(A) Write(A, A_) WriteLock(B) Write(B, B_) WriteUnlock(A) WriteUnlock(B)	ReadLock(A) A' = Read(A) ReadLock(B) B' = Read(B) Print(A' + B') ReadUnlock(A) ReadUnlock(B)	ReadLock(B) B'' = Read(B) WriteLock(A) Write (A, 20) Print(B'') WriteUnlock(A) ReadUnlock(B)

We assume that

- Time is divided into discrete timeslots (e.g. t=1, 2, 3, ...).
- Each operation can be executed within one timeslot.
- Simultaneous operations from different transactions may be executed at the same timeslot, if not prevented by locking.
- However, unlocking operation cannot be executed simultaneously with other locking operations on the same data at the same timeslot.
- Also, WriteLock operation cannot be executed simultaneously with other locking/unlocking operations on the same data at the same timeslot.

6. Considering T1 and T2, give an instance of execution, such that T1 can be executed successfully, but T2 fails. And explain why T2 fails in the instance. (1 Mark)

T1	T2
ReadLock(B) B_ = Read(B) B_ = B_ + 100 ReadLock(A) A_ = Read(A) A_ = A_ - 100 WriteLock(A) Write(A, A_) WriteLock(B) Write(B, B_) WriteUnlock(A) WriteUnlock(B)	ReadLock(A) A' = Read(A) ReadLock(B) B' = Read(B) Print(A' + B') ReadUnlock(A) ReadUnlock(B)

In this scenario, when T2 starts a little bit later, it will not be able to obtain the ReadLock of A whereas T1 has obtained it ahead. So T2 will fail.

7. Considering T1 and T2, give an instance of execution, such that both T1 and T2 can be executed successfully, using the minimal number of timeslots. (1 Mark)

T1	T2
ReadLock(B) B_ = Read(B) B_ = B_ + 100 ReadLock(A) A_ = Read(A) A_ = A_ - 100 WriteLock(A) Write(A, A_) WriteLock(B) Write(B, B_) WriteUnlock(A) WriteUnlock(B)	ReadLock(A) A' = Read(A) ReadLock(B) B' = Read(B) Print(A' + B') ReadUnlock(A) ReadUnlock(B)

In this scenario, both T1 and T2 will operate without disturbing each other. And total time slots would be 12.

8. Considering T1, T2 and T3, give an instance of execution, such that all T1, T2 and T3 can be executed successfully, using the minimal number of timeslots. (2 Marks)

T1	T2	T3
ReadLock(B) B_ = Read(B) B_ = B_ + 100 ReadLock(A) A_ = Read(A) A_ = A_ - 100 WriteLock(A) Write(A, A_) WriteLock(B) Write(B, B_) WriteUnlock(A) WriteUnlock(B)	ReadLock(A) A' = Read(A) ReadLock(B) B' = Read(B) Print(A' + B') ReadUnlock(A) ReadUnlock(B)	 ReadLock(B) B'' = Read(B) WriteLock(A) Write (A, 20) Print(B'') WriteUnlock(A) ReadUnlock(B)

In the above case, the total time slots would be 19.