Cloud Computing
Summer Term 2012

Björn Lohrmann
bjoern.lohrmann@tu-berlin.de
FG CIT

Project Assignment No. 1

DUE: 10.05.2012 23:59

The primary goal of this exercise is to gain insight into performance characteristics of XEN paravirtualization using Amazon AWS as a testbed and comparing these values to a setup without virtualization. The secondary goal is to also familiarize students with Amazon AWS and the respective tools as an example of today's Infrastructure-as-a-Service clouds.

Note

The tutorial session on 27.04.2012 mentioned comparing XEN hvm (virtualization with unmodified guest kernel) and paravirtualization. Due to limitations of Amazon AWS that we were not aware of at that time, it is currently not possible to create hvm instances in Amazon EC2. Therefore we will only compare benchmark measurements of paravirtualized instances with measurements from an actual, physical machine (these are provided). You do not have to install an unmodified (non-paravirtual) kernel, however you still have to create your own AMI.

The benchmark measurements we provide („novirt" in the csv files) give an impression of non-virtualized benchmark performance and have been measured on the following hardware:

- *Intel(R) Xeon(R) CPU X5355 @ 2.66GHz*
- *32GB DDR2-RAM PC667*
- *400GB Seagate SATA II 7200 rpm*
- *Intel Gbit Ethernet*

1.Prerequisites

- Sign up for a group in the course website in ISIS if you have not already done so. There should be as many groups of three students as possible, only one smaller/larger group is allowed.
- If your group is full and has not already received an Amazon AWS Grant Code via the ISIS messaging system, please contact us.
- One member of your group must be in possession of a credit card and create an Amazon AWS account in order to redeem the grant code. If necessary, he/she can allow access to the account using AWS Identity and Access Management (IAM) via the AWS Management Console. How to use IAM is described here. The grant should be more than sufficient to successfully complete this project assignment.
- Setup the Amazon API and AMI tools on your workstation or laptop. They are required to complete the assignment. Make sure you always use the eu-west-1 region endpoint URL in your EC2_URL environment variable (see totorial slides).
- Using the Amazon EC2 online documentation you can find out how to setup and use the API and AMI tools, how to work with instances etc.

2.Creating an AMI

First, create your own custom AMI, that has all the necessary benchmarking tools installed. Use the Amazon API tools to create an initial instance and then follow the AMI creation guide in the AWS documentation to create your custom AMI. Make sure to keep a listing of the API/AMI shell commands you executed, as they are part of your submission (see section „Submission Deliverables). Remember not to include any private information such as your access key or secret key in your submission (replace them with dummy strings). Your resulting AMI must...

- ...be a 64bit AMI.
- ...be instance-store backed, not EBS-backed. This is important for the disk benchmark.
- ...be uploaded in the „eu-west-1" region.
- ...run a recent paravirtual linux kernel (anything newer than 3.0 is new enough).
- ...contain the benchmark scripts provided on ISIS and the tools they depend on (gcc, iperf and R).

Hints:

- The easiest way to do this is to create an instance in the eu-west-1 region from an existing AMI and install the tools in it.
- We suggest you use the official „Ubuntu Server 11.10" AMI (ami-895069fd) however if you want to use a different pre-existing AMI you may do so. Additional packages to be installed for benchmarking on Ubuntu (requires multiverse repository): build-essential r-base iperf ec2-api-tools ec2-ami-tools

3.Single Instance Benchmarks

Based on your custom AMI, create three instances in region „eu-west-1" and availability zone „eu-west-1a", using the instance types „m1.small", „m1.medium" and „m1.large". Use the Amazon API tools to create the instances and keep a listing of the API/AMI commands as they are part of your submission.

On each instance run the benchmarks provided in the additional material provided in ISIS. Execute only one benchmark per instance at any given time and make sure that there is no other CPU or IO intensive process running on the instance. Rerun each benchmark at least three times to make sure you minimize temporary effects. The averages of your benchmark measurements shall be entered into the provided csv files. Measurements from the m1.small instance are ented into the lines beginning with „pv-small", those from m1.medium into the lines beginning with „pv-medium" etc. The following benchmarks should be executed on each instance:

- `linkpack.sh`
  - LINPACK is a popular benchmark in the supercomputing community
  - Enter the average KFLOPS value of you runs into `results/flops.csv`
- `memsweep.sh`
  - Enter the average measured memsweep time of your runs into `results/memsweep.csv`
- `syscall.sh`

    - Enter the average measured time for 50M syscalls of your runs into `results/syscall.csv`
- `fork.sh`
    - Enter the average measured time for 1M forks of your runs into `results/fork.csv`
- `disk.sh`
    - Pause for 15min between each run of this benchmark, otherwise caching effects will give the false impression of very high disk speed (which in fact is the speed of the memory bus).
    - Enter the average measured reading speed of your runs into `results/disk.csv`

Use the provided R script in `results/plots.r` to produce a PDF plot of your measurement data. You can plot the data in one of your instances by executing the following shell command ($ stands for the shell prompt):

```
$ R CMD BATCH plots.r
```

This produces a preliminary file `plots.pdf` that contains barplots of all your measurements. Make sure the R script does not produce any errors and that you have the csv files and API/AMI command listings on your local computer as they are part of the submission.

When you are done, shut down the m1.medium instance. The remaining instances will be required by the next exercise.

4. Network Throughput Benchmark

You should have two remaining instances now: m1.small and m1.large. Create two additional instances, one with type m1.small and one with type m1.large.

Now, run the network throughput benchmark in `network-client.sh/network-server.sh` in the following configurations:

- With the m1.small instances as client/server
- With the m1.large instances as client/server

Rerun the benchmarks at least three times and copy the averages (seven average Mbits/s values per configuration) into the provided file `results/network.csv`.

Use the R script to produce the final plot of your measurement data. Make sure you have all the files for the ISIS submission on your local machine and shut down all four instances after completing this section.

5.Evaluation of Benchmark Results

Based on your benchmarks and your knowledge about XEN paravirtualization, provide short answers to the following questions.

1. LINPACK Benchmark

1. Find out what the LINPACK benchmark measures (try google :-). Would you expect paravirtualization to affect the LINPACK benchmark? Why?
2. Look at your LINPACK measurements. Are they consistent with your expectations. If not, what could be the reason?

2. Memsweep Benchmark
    1. Find out how the memsweep benchmark works by looking at the C code. Would you expect paravirtualization to affect the memsweep benchmark? Why?
    2. Look at your memsweep measurements. Are they consistent with your expectations. If not, what could be the reason?
3. Syscall Benchmark
    1. Find out how the syscall benchmark works by looking at the C code. Would you expect paravirtualization to affect the syscall benchmark? Why?
    2. Look at the syscall measurements. Are they consistent with your expectations. If not, what could be the reason?
4. Fork Benchmark
    1. Find out how the fork benchmark works by looking at the C code. Would you expect paravirtualization to affect the fork benchmark? Why?
    2. Look at the fork measurements. Are they consistent with your expectations. If not, what could be the reason?
5. Disk Benchmark
    1. Find out how the disk benchmark works by looking at the shell code. Would you expect paravirtualization to affect the disk benchmark? Why?
    2. Look at the disk measurements. Are they consistent with your expectations. If not, what could be the reason?
6. Network Throughput Benchmark
    1. Find out how the network benchmark works by looking at the shell code (the iperf manpage should be helpful). Would you expect paravirtualization to affect the network benchmark? Why?
    2. Look at the network throughput measurements. Are they consistent with your expectations. If not, what could be the reason?

6.Submission Deliverables

Your submission on ISIS should be a single .zip file containing the following:

- A folder „results" containing all the .csv files from Sections 3 and 4.
- The final plots.pdf produced by the R script in Section 4.
- A .txt or .pdf file containing
    - The Amazon API/AMI tool commands used to create your AMI (Section 2)
    - The Amazon API tool commands used to create the additional instances (Section 3)
    - The evaluation of the benchmark results (Section 5).