Final Exam Exercise: Cloud Computing Performance Testing

Objective

Evaluate and compare the performance of virtual machines (VMs), containers.

General Instructions

- Students can choose any Linux distribution for their virtual machines and containers.
- Any virtualization and containerization system may be used; **VirtualBox** and **Docker** are recommended.
- Use performance testing tools discussed during the lectures such as **HPL (High-Performance Linpack)**, **stress-ng**, and **sysbench**, **IOZone**, etc.
- **Optionally**, test I/O performance on both **local filesystems** and **NFS filesystems**.
- **Optionally**, test performance of the host machine (when possible).

Part 1: Virtual Machines Performance Test

Setup

1. Create a set of virtual machines (e.g., 2-3 VMs).
2. Connect them using a virtual switch and local IPs.
3. Allocate limited resources to each VM (e.g., 2 CPUs and 2GB of RAM).

Performance Tests

- **CPU Test**: Use **HPC Challenge Benchmark** for high-performance computation benchmarking (availabe for Ubuntu).
- **General System Test**: Use **stress-ng** or **sysbench** to evaluate CPU and memory performance.
- **Disk I/O Test**: Use `IOZone` to test local filesystem I/O. Optionally, configure an NFS filesystem and test its I/O performance.
- **Network Test**: Use tools like `iperf` or `netcat` to measure network throughput and latency between VMs.

Note on HPC Challenge Benchmark

HPCC is a suite of benchmarks that measure performance of processor, memory subsytem, and the interconnect. For details refer to the HPC~Challenge web site (\ url{http://icl.cs.utk.edu/hpcc/}.)

In essence, HPC Challenge consists of a number of tests each of which measures performance of a different aspect of the system: HPL, STREAM, DGEMM, PTRANS, RandomAccess, FFT.

If you are familiar with the High Performance Linpack (HPL) benchmark code (see the HPL web site: http://www.netlib.org/benchmark/hpl/}) then you can reuse the input file that you already have for HPL. See http://www.netlib.org/benchmark/hpl/tuning.html for a description of this file and its parameters. You can use the following sites for finding the appropriate values:

- Tweak HPL parameters: https://www.advancedclustering.com/act_kb/tune-hpl-dat-file/
- HPL Calculator: https://hpl-calculator.sourceforge.net/

The main parameters to play with for optimizing the HPL runs are:

- NB: depends on the CPU architecture, use the recommended blocking sizes (NB in HPL.dat) listed after loading the toolchain/intel module under $EBROOTIMKL/compilers_and_libraries/linux/mkl/benchmarks/mp_linpack/readme.txt, i.e
  - NB=192 for the broadwell processors available on iris
  - NB=384 on the skylake processors available on iris
- P and Q, knowing that the product P x Q SHOULD typically be equal to the number of MPI processes.
- Of course N the problem size.

To run the HPCC benchmark, first create the HPL input file and then simply exeute the hpcc command from cli.

Part 2: Containers Performance Test

Setup

1. Create a Docker Compose file to define a set of containers (e.g., 2-4 containers).
2. Connect the containers using internal Docker network.
3. Limit resources for each container using Docker Compose options (e.g., 2 CPUs and 2GB of RAM).

Performance Tests

- **CPU Test**: Run **HPC Challenge Benchmark** for CPU performance and/or **stress-ng** or **sysbench** for general system performance.
- **Disk I/O Test**: Use `IOZone` to test local filesystem performance within containers.
- **Network Test**: Use `iperf` or `netcat` to measure network throughput between containers.

Final Comparison

Analyze

- Compare the performance results across VMs, containers, and optionally with the host machine.
- Discuss the impact of resource allocation, virtualization overhead, and network/file system efficiency.

Deliverables

- Submit a report containing:
  - Steps for setting up VMs and containers.
  - Performance metrics (HPL, stress-ng, sysbench, disk I/O, and network throughput).
  - Observations and analysis of the performance tests.