Cloud Computing

Winter Term 2020/2021
Tutorial Session 3

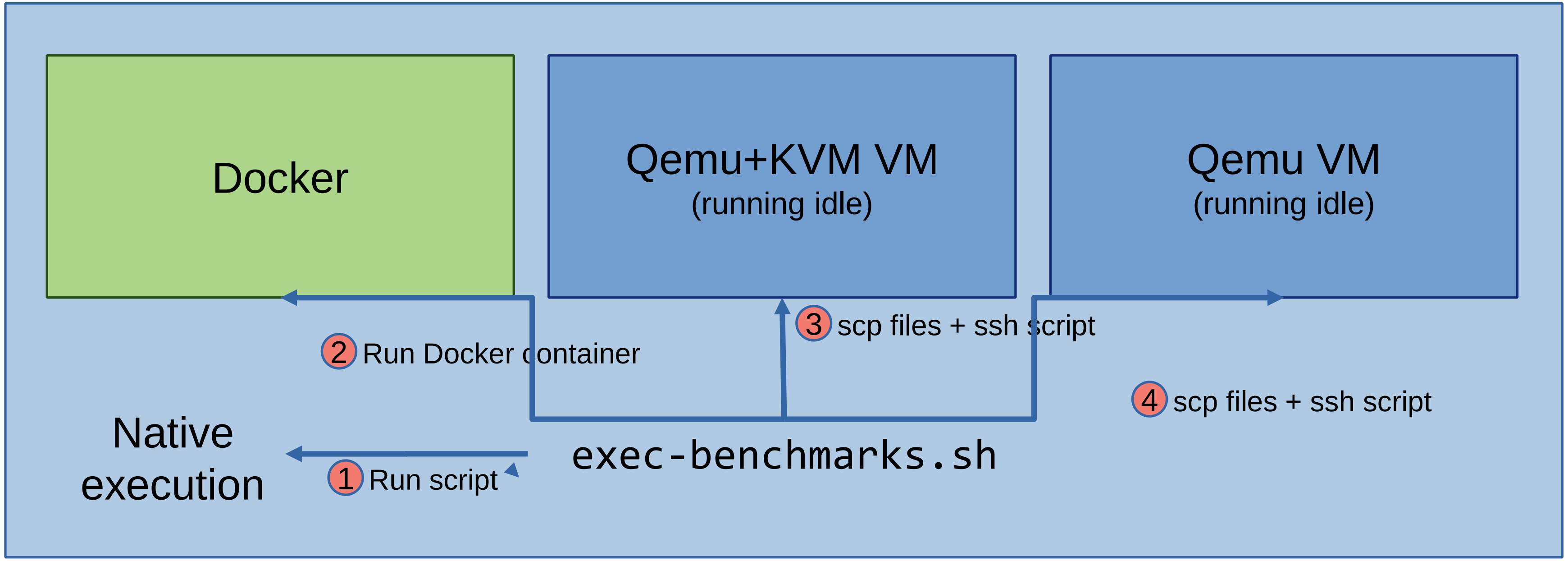
Ilja Behnke, Alexander Acker
Distributed and Operating Systems

i.behnke@tu-berlin.de

Practical Assignment 3

- Due: 20.12.2020
- Summary:
  - Work on 1 host machine
    - Preferably your laptop (physical machine). If you don’t have Linux, use a VM.
  - Prepare virtualization environments:
    - Qemu/KVM
    - Docker
  - Write 2 new benchmarks:
    - Forksum
    - Iperf3 (uplink speed)
  - Execute benchmarks on different virtualization platforms
- Public cloud platforms are not mandatory for this assignment, but do not yet delete your accounts (shut down your VMs if not used for the assignment)

Benchmark Setup



Experiment host: Laptop or (Public Cloud) VM

Virtualization Platforms

- Native execution
    - Simply execute `./benchmark.sh`
- Docker
    - Write Dockerfile that executes benchmark script
    - The container image must contain all tools and files for all benchmarks, execute the benchmark when started without parameters, and exit after printing the results
- Qemu (with and without KVM)
    - Use an Ubuntu 18.04 cloud image
    - Either work directly with qemu-system-* executable, or use a management program such as libvirt

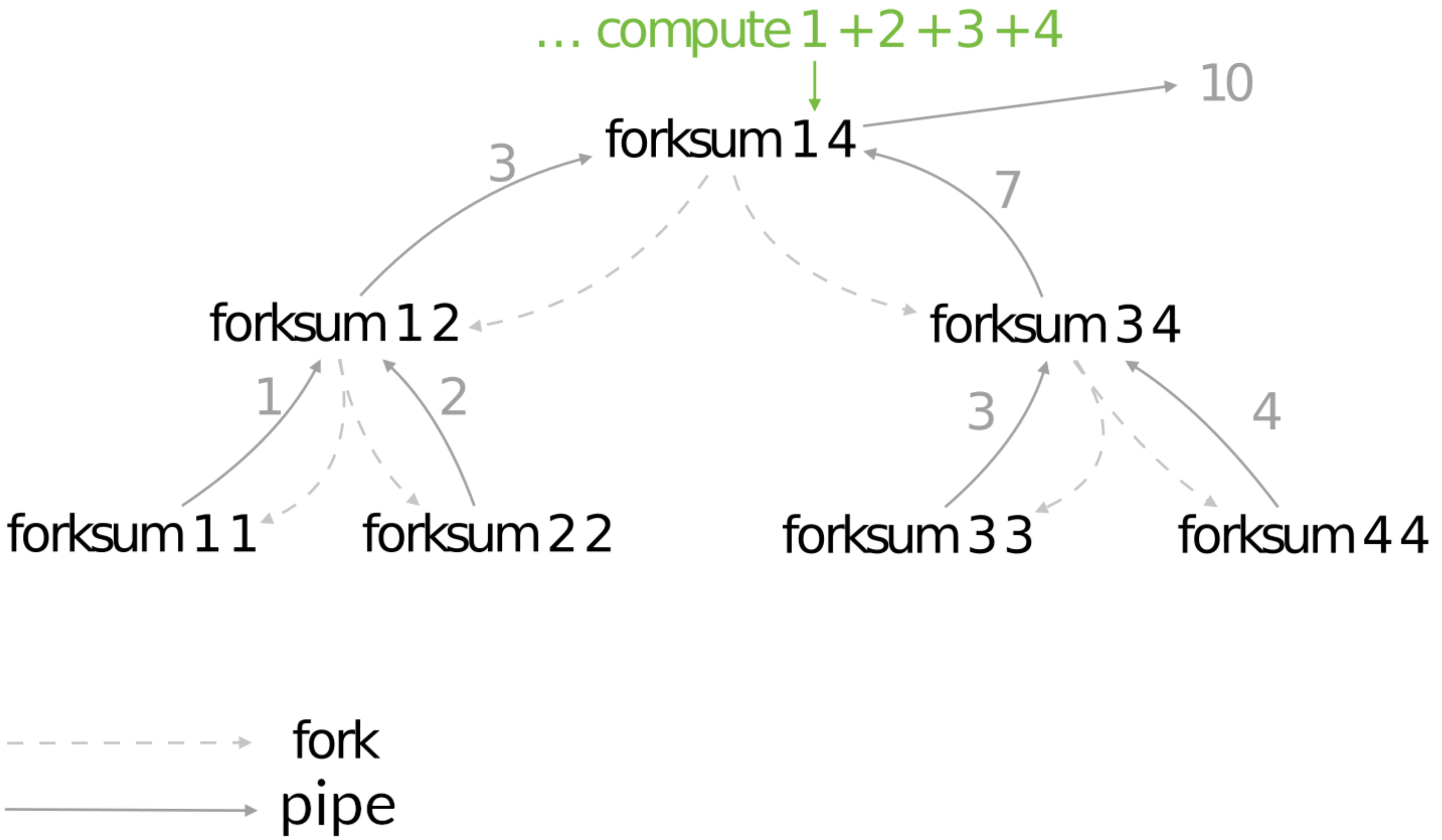
Benchmarks

- Basic resources: CPU, RAM and disk access
  - Reuse from assignments 1 & 2
  - Option: use the benchmark.sh script provided on ISIS
- New benchmark: fork
  - Creates many parallel processes to calculate sum of an interval
  - Main benchmark target: system calls
- New benchmark: iperf3
  - Measure uplink speed to host machine
  - Main benchmark target: networking impact of VMs

Fork Benchmark

- Program receives 2 parameters: start and end of integer range
    - Task: compute sum of all integers within the range
    - Example: **`./forkbench 100 1000`** should print **`495550`**
- Every sum is executed by a separate child process
    - If start == end: output value (end of the recursion)
    - Else: spawn 2 child processes: one for lower sub-range and one for upper sub-range
        - After child-processes return their results, parse them and output the sum

Forksum: Example



Forksum: required C functions

- **`fork()`**: continue program as two separate processes
    - Return value of `fork()` tells the program if it is the child or parent process
- **`pipe()`**: Create a bidirectional pipe that can be used to write in the child process, and to read in the parent process
- **`fdopen()`**: Open file descriptor as a stream for reading and writing
- **`fprintf()`**: Write formatted text to a stream, can be used to write to stderr
- **`getline()`**: Read a line of text from a stream
- Other useful functions: wait(), perror(), read(), strotl(), printf(), close(), exit()

iperf3 Uplink Benchmark

- iperf3 measures network performance
- Requires a server to establish a connection
  - Install iperf3 on host and guest machines
  - Use host as iperf3 server
  - See if there are differences depending on type of VM
  - Compare with results using a public iperf3 server