

# CSCI315 – Operating Systems Design

Department of Computer Science  
Bucknell University

## Virtual Machines Examples

**Ch 18.7-18.8**

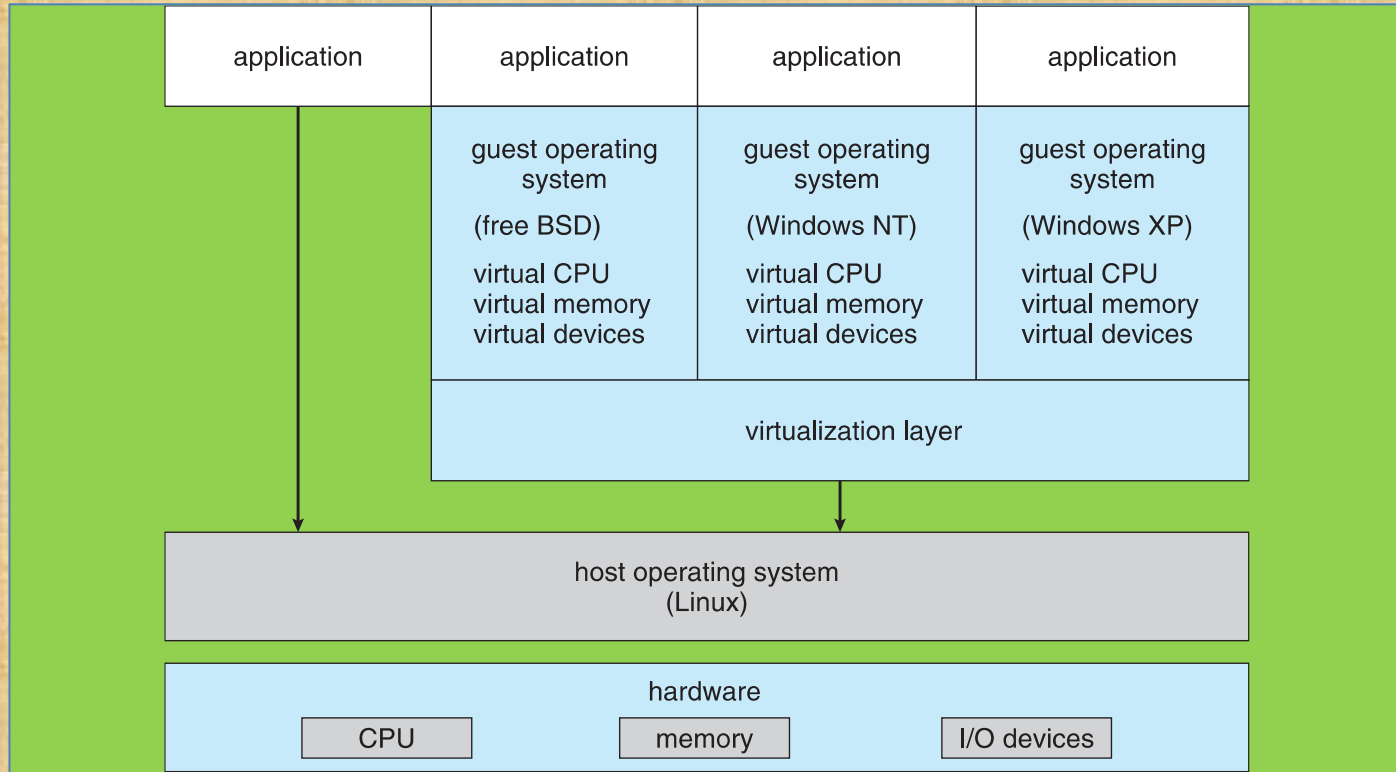
*This set of notes is based on notes from the textbook authors, as well as L. Felipe Perrone, Joshua Stough, and other instructors.*

*Xiannong Meng, Fall 2021.*

# Examples - VMware

- VMware Workstation runs on x86, provides VMM for guests
- Runs as application on other native, installed host operating system -> Type 2
- Lots of guests possible, including Windows, Linux, etc. all runnable concurrently (as resources allow)
- Virtualization layer abstracts underlying HW, providing guest with its own virtual CPUs, memory, disk drives, network interfaces, etc.
- Physical disks can be provided to guests, or virtual physical disks (just files within host file system)

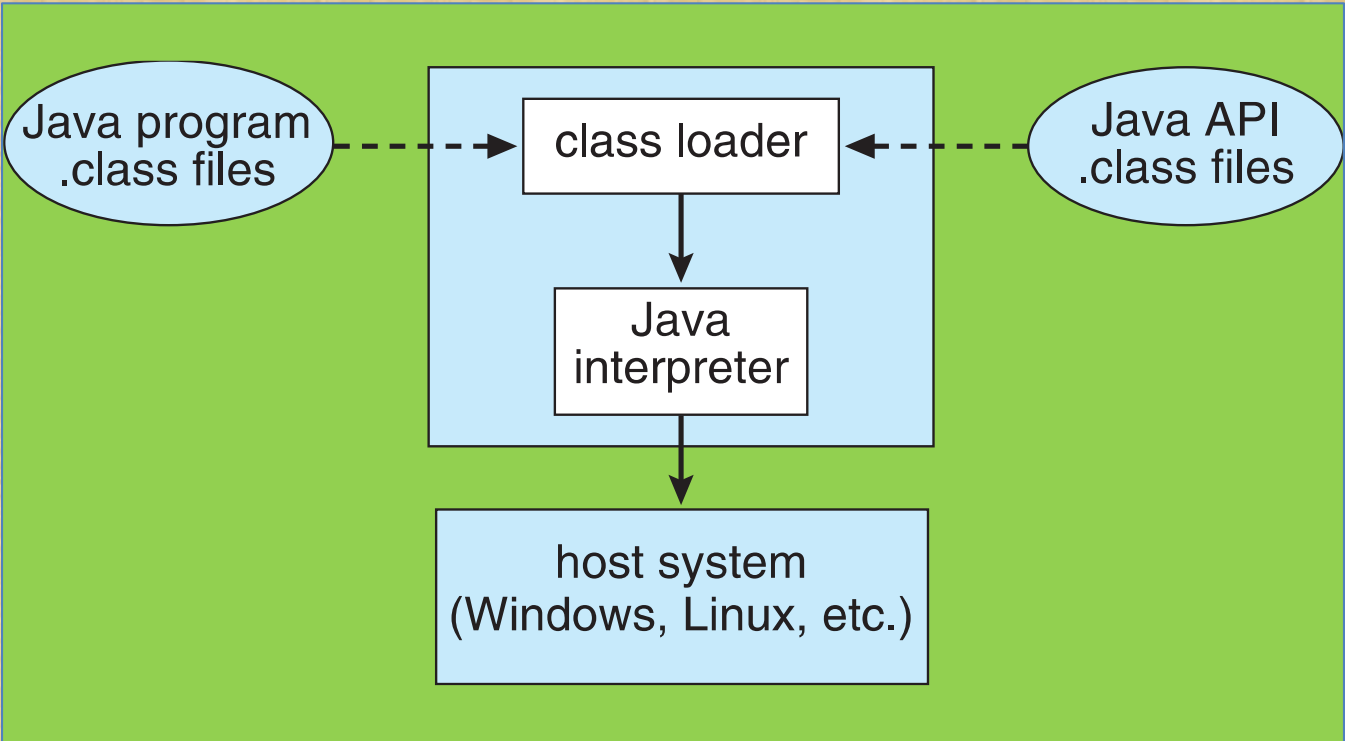
# VMware Workstation Architecture



# Examples – Java Virtual Machine

- Example of programming-environment virtualization
- Very popular language / application environment invented by Sun Microsystems in 1995
- Write once, run anywhere
- Includes language specification (Java), API library, Java virtual machine (JVM)
- Java objects specified by class construct, Java program is one or more objects
- Each Java object compiled into architecture-neutral **bytecode** output (`.class`) which JVM **class loader** loads
- JVM compiled per architecture, reads bytecode and executes
- Includes **garbage collection** to reclaim memory no longer in use
- Made faster by **just-in-time (JIT)** compiler that turns bytecodes into native code and caches them

# The Java Virtual Machine



# Virtualization Research

- Very popular technology with active research
- Driven by uses such as server consolidation
- **Unikernels**, built on **library operating systems**
  - Aim to improve efficiency and security
  - Specialized machine images using one address space, shrinking attack surface and resource footprint of deployed applications
  - In essence, compile application, libraries called, and used kernel services into single binary that runs in a virtual environment

<https://en.wikipedia.org/wiki/Unikernel>

<http://unikernel.org/>

# Virtualization Research

- Better control of processes available via projects like **Quest-V**
  - Real time execution and fault tolerance via virtualization instructions
  - Partitioning hypervisors partition physical resources amongst guests, fully-committing all resources (rather than overcommitting)
  - For example a Linux system that lacks real-time capabilities for safety- and security-critical tasks can be extended with a lightweight real-time OS running in its own VM

<http://www.cs.bu.edu/fac/richwest/quest.php>

# Virtualization Research (Cont.)

- Separation hypervisors like Quest-V, each task runs in a virtual machine
  - Hypervisor initializes system and starts tasks but not involved in continuing operation
  - Each VM has its own resources the task manages
  - Tasks can be real time and more secure
  - Other examples are Xtratum, Siemens Jailhouse
  - Can build chip-level distributed system
  - Secure shared memory channels implemented via extended page tables for inter-task communication
  - Project targets include robotics, self-driving cars, Internet of Things



# Current Popular List of VM Software (1)

- Based on a list by TechRadar (dated 09/24/2020)
  - 1. [VMware Workstation Player](#)
  - 2. [VirtualBox](#)
  - 3. [Parallels Desktop](#)
  - 4. [QEMU](#)
  - 5. [Citrix Hypervisor](#)
  - 6. [Xen Project](#)
  - 7. [Microsoft Hyper-V](#)

<https://www.techradar.com/best/best-virtual-machine-software>

# Current Popular List of VM Software (2)

- Based on a list by Software Testing Help (dated 10/1/2020)
  - [SolarWinds Virtualization Manager](#)
  - [V2 Cloud](#)
  - [VM Ware Fusion](#)
  - [Parallels Desktop](#)
  - [Oracle Virtualization](#) (a.k.a. Virtual Box)
  - [VM Ware Workstation](#)
  - [QEMU](#)
  - [Virtual PC](#)
  - [Microsoft Hyper-V](#)
  - [Redhat Virtualization](#)
  - [Veertu-for MAC](#)
  - [Apple-Boot Camp](#)

<https://www.softwaretestinghelp.com/virtualization-software/>

# Virtual Machine: QEMU

- Here is a running example of VM: QEMU.
  - hardware emulator of x86 processor
- I used it to study the bootstrap programs, which was demonstrated early in the course.
- One can specify the disk capacity and memory configuration for each run.
- <https://www.qemu.org/>
- <https://en.wikipedia.org/wiki/QEMU>

# Some Screenshots Running QEMU

```
c:\Users\xmeng\qemu-data>dir
Volume in drive C is Windows
Volume Serial Number is 3A36-7B83

Directory of c:\Users\xmeng\qemu-data

08/02/2020  06:19 PM    <DIR>          .
08/02/2020  06:19 PM    <DIR>          ..
11/15/2020  11:16 AM       1,576,534,016  alpine.qcow2
06/11/2020  10:10 AM           512  boot_go_pm.bin
06/16/2020  09:16 AM           512  boot_sect.bin
06/20/2020  10:01 AM       8,192  os-image
05/31/2020  07:19 AM           71  qemu-alpine.bat
06/01/2020  08:37 AM           21  qemu-simple.bat
08/02/2020  06:19 PM           11  users.txt
              7 File(s)  1,576,543,335 bytes
              2 Dir(s)  145,420,840,960 bytes free

c:\Users\xmeng\qemu-data>type qemu-alpine.bat
@echo off
echo.
qemu-system-x86_64 -m 512 -nic user -hda alpine.qcow2
c:\Users\xmeng\qemu-data>
```

QEMU files on my Windows computer. “alpine.qcow2” is a virtual Linux brand OS.

<https://alpinelinux.org/>

# Some Screenshots Running QEMU

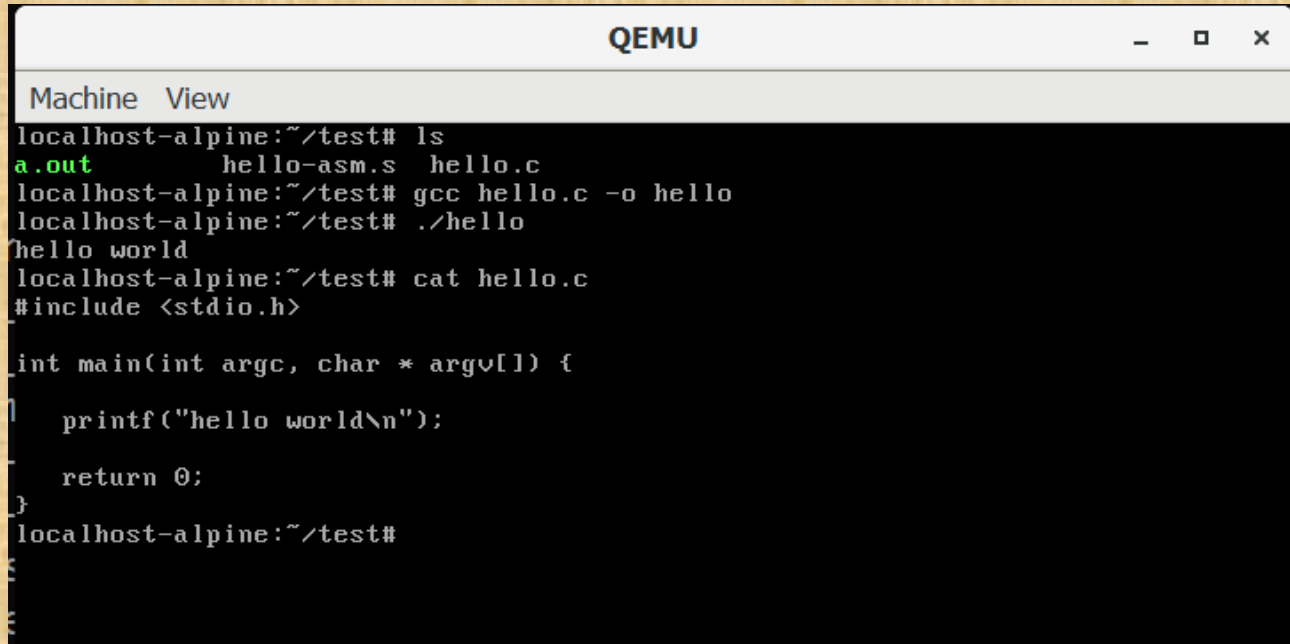
```
QEMU - □ ×
Machine View
* Checking local filesystems ...
/dev/sda3: clean, 9574/456960 files, 379712/1826048 blocks
/dev/sda1: clean, 24/25688 files, 30440/102400 blocks [ ok ]
* Remounting root filesystem read/write ... [ ok ]
* Remounting filesystems ... [ ok ]
* Activating swap devices ... [ ok ]
* Mounting local filesystems ... [ ok ]
* Configuring kernel parameters ... [ ok ]
* Creating user login records ... [ ok ]
* Wiping /tmp directory ... [ ok ]
* Setting hostname ... [ ok ]
* Setting keymap ... [ ok ]
* Starting networking ...
*   lo ... [ ok ]
*   eth0 ... [ ok ]
* Starting busybox syslog ... [ ok ]
* Initializing random number generator ... [ ok ]
* Starting busybox acpid ... [ ok ]
* Starting busybox crond ... [ ok ]
* Starting sshd ... [ ok ]

Welcome to Alpine Linux 3.11
Kernel 5.4.34-0-lts on an x86_64 (/dev/tty1)

localhost-alpine login: _
```

booting up Alpine Linux on my Windows computer under QEMU

# Some Screenshots Running QEMU



```
QEMU
Machine View
localhost-alpine:~/test# ls
a.out      hello-asm.s  hello.c
localhost-alpine:~/test# gcc hello.c -o hello
localhost-alpine:~/test# ./hello
hello world
localhost-alpine:~/test# cat hello.c
#include <stdio.h>

int main(int argc, char * argv[]) {
    printf("hello world\n");

    return 0;
}
localhost-alpine:~/test#
```

compiling and running hello.c

# Some Screenshots Running QEMU

```
QEMU - Press Ctrl+Alt+G to release grab
Machine View
Retrieving /home/xmeng/csci315/Labs/Lab05/solution/bin
Retrieving /home/xmeng/csci315/Labs/Lab05/solution/include
/home/xmeng/csci315/Labs/Lab05/solution/inclu 100% 2666 32.7KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/solution/inclu 100% 2279 27.4KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/solution/inclu 100% 2547 31.7KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/solution/inclu 100% 2694 32.5KB/s 00:00
Retrieving /home/xmeng/csci315/Labs/Lab05/solution/obj
Retrieving /home/xmeng/csci315/Labs/Lab05/solution/src
/home/xmeng/csci315/Labs/Lab05/solution/src/c 100% 3562 42.8KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/solution/src/p 100% 2735 34.1KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/solution/src/c 100% 1140 14.2KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/solution/src/a 100% 1524 18.8KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/solution/src/c 100% 3621 44.9KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/solution/src/p 100% 5266 64.7KB/s 00:00
Fetching /home/xmeng/csci315/Labs/Lab05/student/ to student
Retrieving /home/xmeng/csci315/Labs/Lab05/student
Retrieving /home/xmeng/csci315/Labs/Lab05/student/bin
/home/xmeng/csci315/Labs/Lab05/student/Doxyfi 100% 66KB 44.3KB/s 00:01
Retrieving /home/xmeng/csci315/Labs/Lab05/student/include
/home/xmeng/csci315/Labs/Lab05/student/includ 100% 2279 28.1KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/student/Makefi 100% 1328 16.3KB/s 00:00
Retrieving /home/xmeng/csci315/Labs/Lab05/student/src
/home/xmeng/csci315/Labs/Lab05/student/src/ci 100% 1140 14.1KB/s 00:00
/home/xmeng/csci315/Labs/Lab05/student/src/pr 100% 3188 39.6KB/s 00:00
sftp>
```

retrieving files from linuxremote

```
sftp user@linuxremote.bucknell.edu
cd csci315/lab05
mget -r *
```

# Some Screenshots Running QEMU

```
QEMU - Press Ctrl+Alt+G to release grab
Machine View
localhost-alpine:~/lab05/solution# make clean
/bin/rm -rf ./obj/* ./bin/* core* *~ ./src/*~ ./doc/*
localhost-alpine:~/lab05/solution# make
gcc -I ./include -std=gnu99 -Wall -g -c ./src/circular-list.c -o ./obj/circular-list.o
gcc -I ./include -std=gnu99 -Wall -g -c src/adt-test.c -o ./obj/adt-test.o
gcc -I ./include -std=gnu99 -Wall -g -c src/prodcons.c -o ./obj/prodcons.o
gcc -I ./include -std=gnu99 -Wall -g -c ./obj/circular-list.o ./obj/adt-test.o ./obj/prodcons.o -o ./bin/prodcons -lpthread
localhost-alpine:~/lab05/solution# _
```

compiling lab05 on QEMU

running lab05 on QEMU

```
QEMU - Press Ctrl+Alt+G to release grab
Machine View
consumer waiting for buffer
sleep amount 861
consumer waiting for buffer
sleep amount 268
producer waiting for buffer
producer waiting for lock
producer released lock
consumer waiting for lock
consumer released lock
consumer signaled full
CONSUMER: consumed value 0.524886
producer signaled buffer
PRODUCER: produced value 0.524886
sleep amount 32
producer waiting for buffer
producer waiting for lock
producer released lock
consumer waiting for lock
consumer released lock
consumer signaled full
CONSUMER: consumed value 0.165771
producer signaled buffer
PRODUCER: produced value 0.165771
sleep amount 783^C
localhost-alpine:~/lab05/solution# _
```