CS4516 Class Project Overview

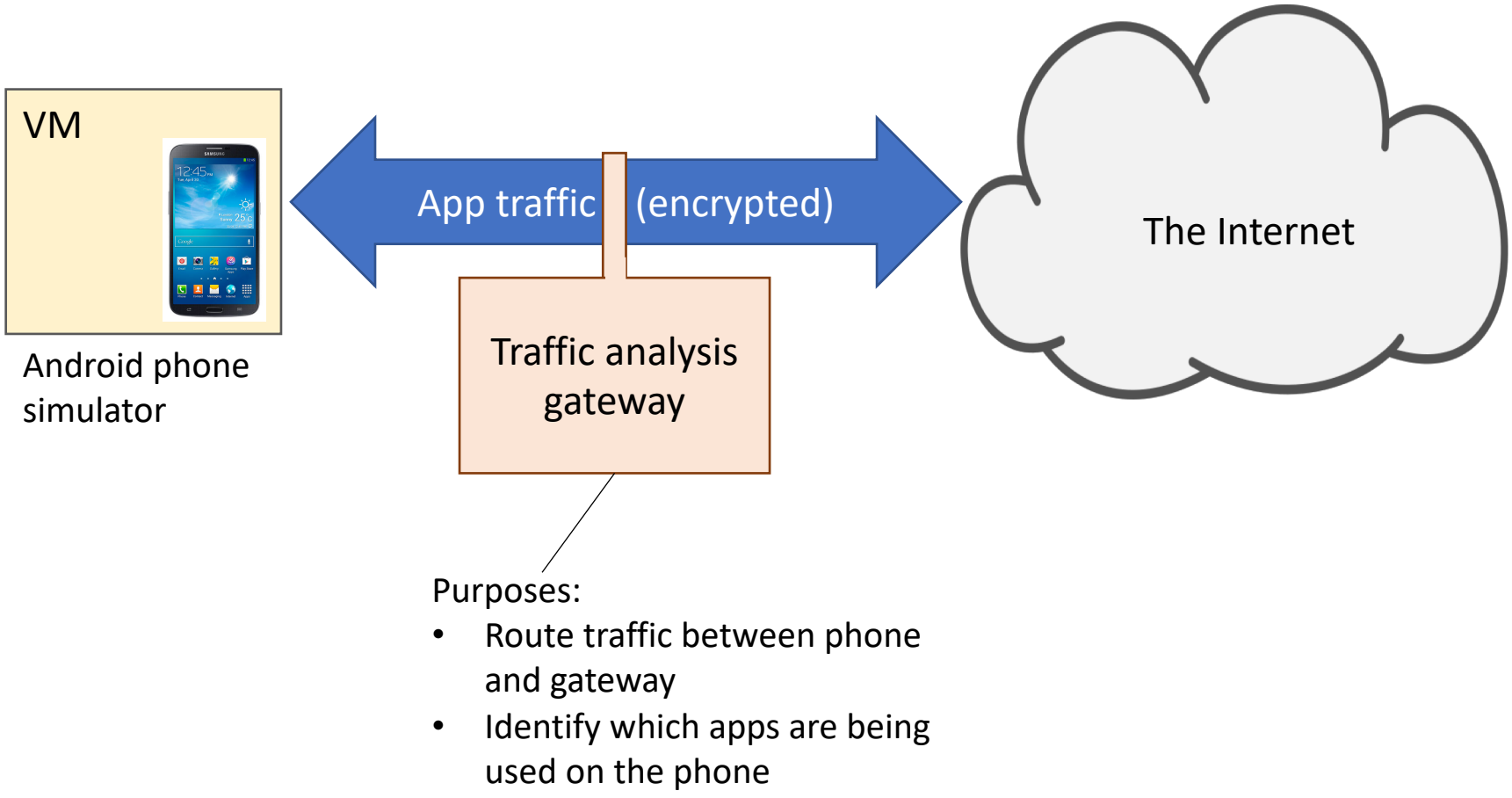
WPI CS4516 Spring 2019 D term

Instructor: Lorenzo De Carli (ldecarli@wpi.edu)

Core points

- Project is organized in four phases
- Each phase ends with a deliverable that must be uploaded on Canvas
- Together with the last deliverable you must also upload a project report
  - Refer to class schedule page (https://web.cs.wpi.edu/~ldecarli/teaching/cs4516D19/schedule.html) for due dates
- You must form 2-person teams

What's the project about?



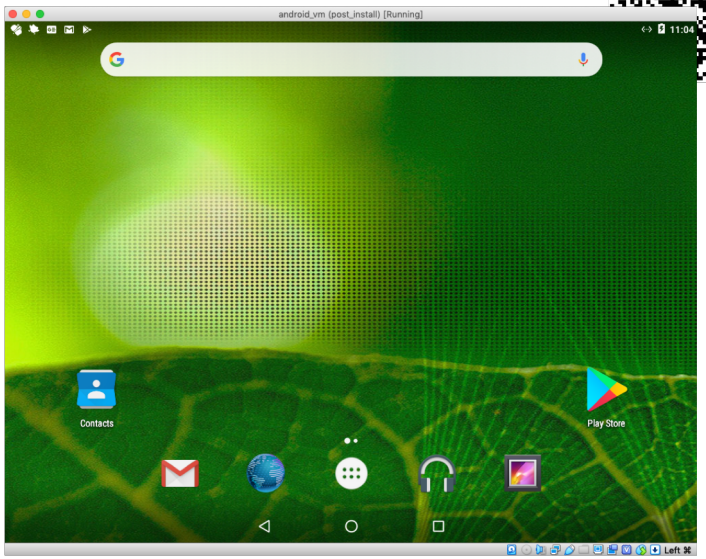
Components you need to implement

- Android phone simulator (virtual machine)
- Internet gateway/analysis box (virtual machine)
- Machine learning model for application identification

Tools you need

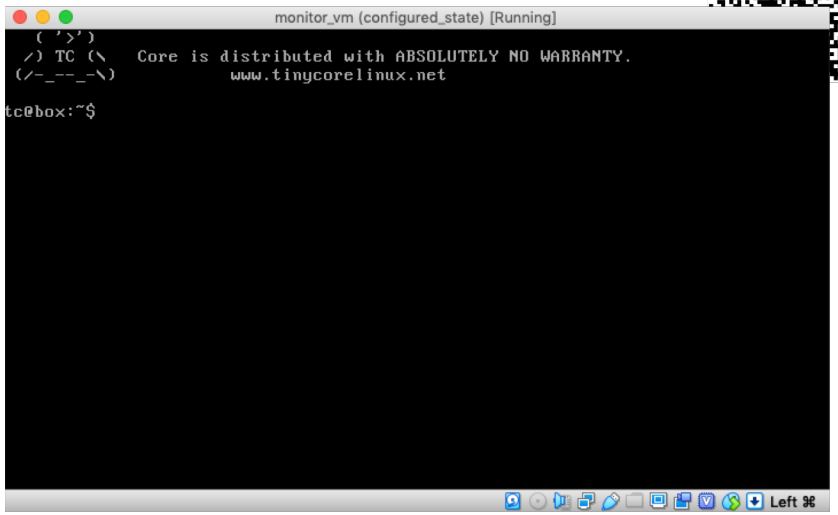
- VirtualBox
  - Virtual machine hypervisor/manager
  - Free
  - Available for Windows/Linux/MacOs
  - **If you don’t have access to a machine that can run it, contact me**
- VM with Android x86
  - We will provide ISO image and installation instructions
- VM with TinyCore Linux (for gateway)
  - We will provide VM and configuration instructions

Android x86



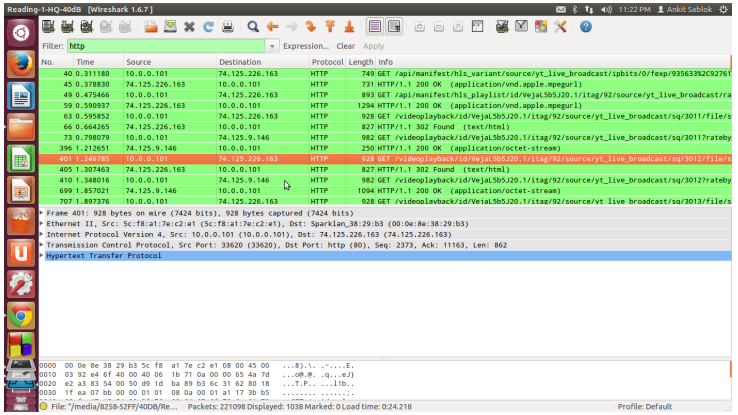
- Full version of Android, suitable for use on laptops and PCs
- We will use version 6.0 (less demanding than recent ones)
- You will need to create your own VM using the provided ISO image and configure it appropriately
- No need to deliver the Android VM with your assignments - we will use our own for testing

TinyCore Linux



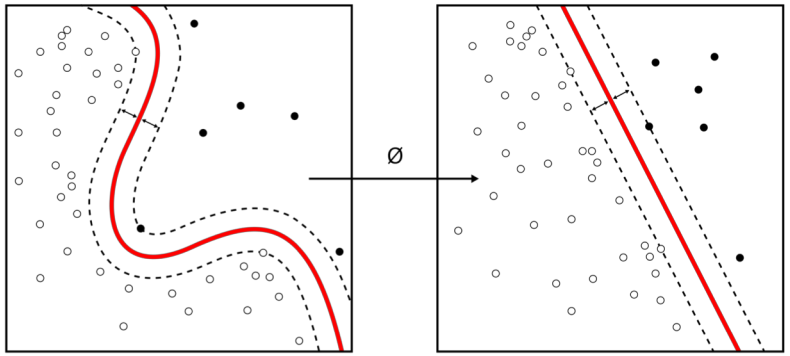
- Version of Linux with minimal memory/processor /disk requirements
- Ideal for efficient implementation of network functions
- You will use it to implement a gateway routing the Android VM's traffic to/from the Internet
- We will provide a base VM and you will need to configure it appropriately

Traffic capturing in the gateway



- Once the infrastructure is in place, you will implement code to capture network packets as they are in transit through the gateway
- You will use libpcap – industry-standard tool for capturing network traces
- Bindings for Python (and probably other languages – but you are required to use it)

Traffic classification



- The core part of the project consists of implementing a classifier to distinguish between different applications (without peeking into packet payloads)
- You will need to collect traffic samples from a set of apps that we will give you, extracts *features* from them, and train a classifier to recognize each app
- We will provide a tutorial on scikit-learn for those unfamiliar with machine learning

Live classification

- Once you figured out how to distinguish between different applications, you will:
  - Plug the model you developed into the gateway to analyze live traffic
  - Perform live traffic classification

Phases and deliverables

Phase 1

- **Goal:** create & configure VMs
- **Deliverable:** TinyCore VM configured to act as gateway for Android VM
- **What we will test:** whether Android VM can connect to the Internet
- **Deadline:** Mon 3/25 10:45 AM

Phase 2

- **Goal:** implement packet capturing infrastructure
- **Deliverables:** TinyCore gateway VM configured to log network flows going between Android VM and the Internet
- **What we will test:** if the VM correctly generates flow logs when the Android VM generates traffic
- **Deadline:** Mon 4/1 10:45 AM

Phase 3

- **Goal:** implementing traffic classification model
- **Deliverable:** Python code receiving as input a PCAP trace and returning as output the application that generated it
- **What we will test:** we will run your code on a set of traces and check the accuracy of the classification
- **Deadline:** Mon 4/15 10:45 AM

Phase 4

- **Goal:** implementing live traffic classification
- **Deliverable:** TinyCore gateway configured to identify applications generating the traffic from the Android VM
- **What we will test:** we will run application on the Android VM and verify whether they get identified by the classifier on the gateway
- **Deadline:** Mon 4/22 10:45 AM

Phase 4+ (only for BS/MS extra credit)

- **Goal:** block flows for certain applications
- **Deliverable:** TinyCore gateway configured to block flows from selected applkications
- **What we will test:** whether running a blacklisted application results in appropriate firewall rules being generated
- **Deadline:** Mon 4/29 10:45 AM

Resources

- We will provide a brief in-class tutorial on Thursdays before the beginning of each phase
- We will provide images w/ Android and TinyCore on Canvas to get you started

Expectations

- This is a 4000-level class, so you are expected to be largely independent
  - We will provide enough to get you started – the rest is up to you
- This is an open-ended project – you will find that traffic classification is messy and difficult
  - We will not expect 100% accuracy, but you will need to justify your design choices and potential limitations in your final project report

Why are we doing this?

Two reasons:

1. Learn skills: traffic capture, analysis and classification
2. Reflect on the fact that even strong end-to-end encryption may not be enough to prevent people from learning something about you

First things to do

- Organize in teams of two students. Send me an email with members of your team. I will assign you a team ID that you will need to use in delivering your assignments.
  - **Do this by Monday 3/18**
- If you need to be paired, send me an email so I can arrange it

Next up: phase #1 tutorial

First things first

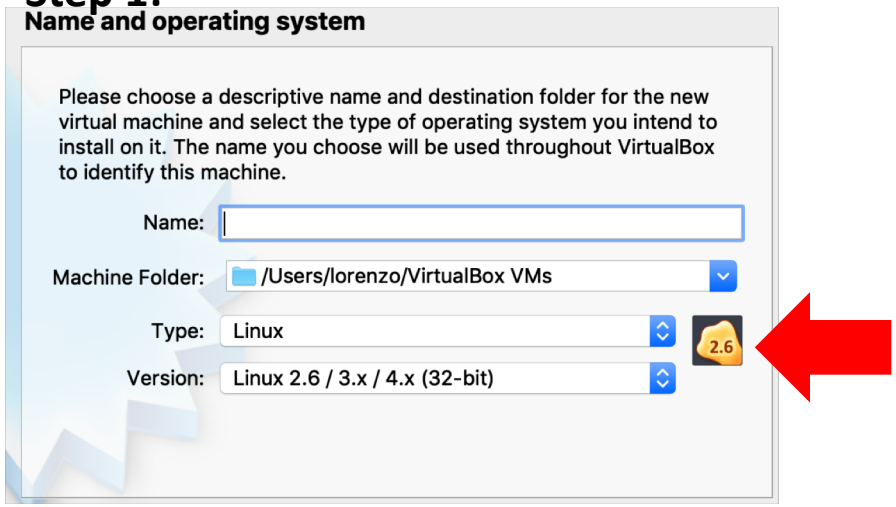
- Download and install VirtualBox
  - https://www.virtualbox.org/
- Download supporting material
  - Go to the Canvas course page, under the ”Files” tab, “Project material” folder, you will find:
    - Android x86 ISO image
    - TinyCore VM image

First, configure Android x86 VM

- Note that we'll only provide the ISO (installation) image – you'll need to create and configure the VM itself

VM creation

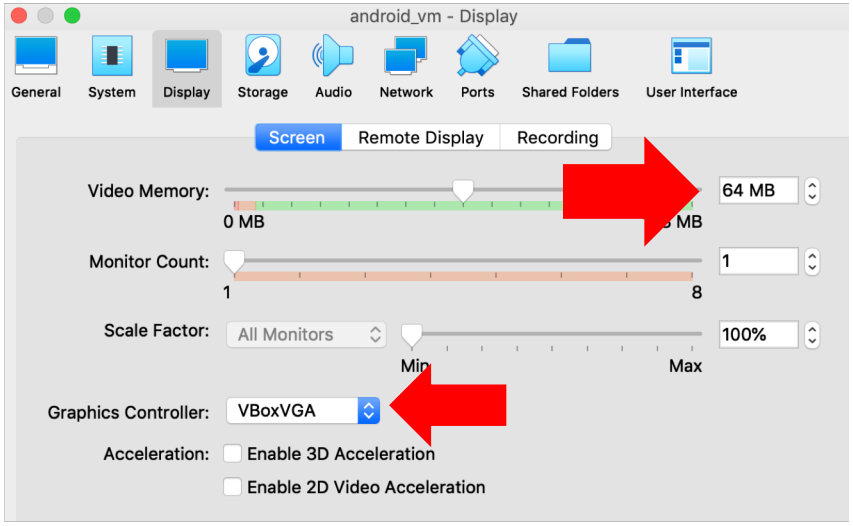
Step 1:



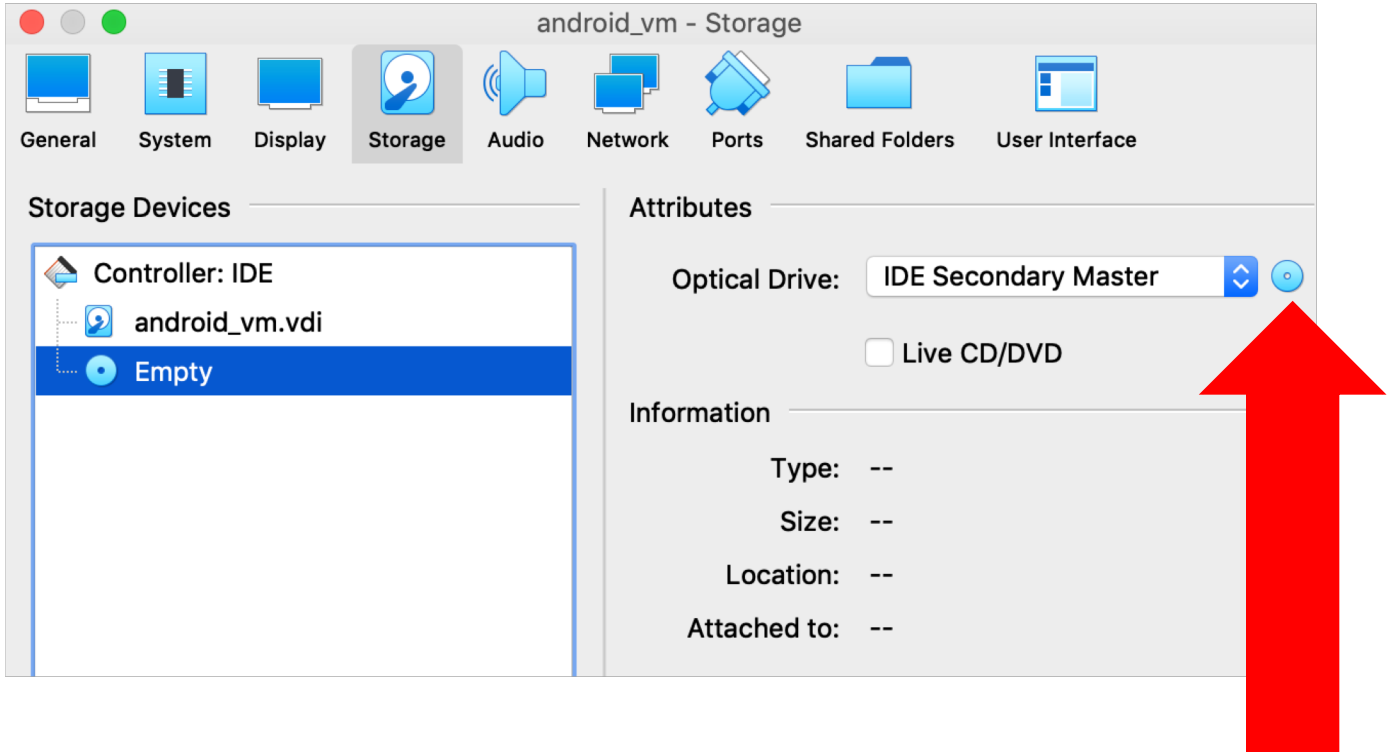
Step 2:

- Set memory to 1024MB
- Create a 8GB virtual disk
  - Format: VDI
  - Storage: dynamically allocated
  - Size: 8GB

Step 3:

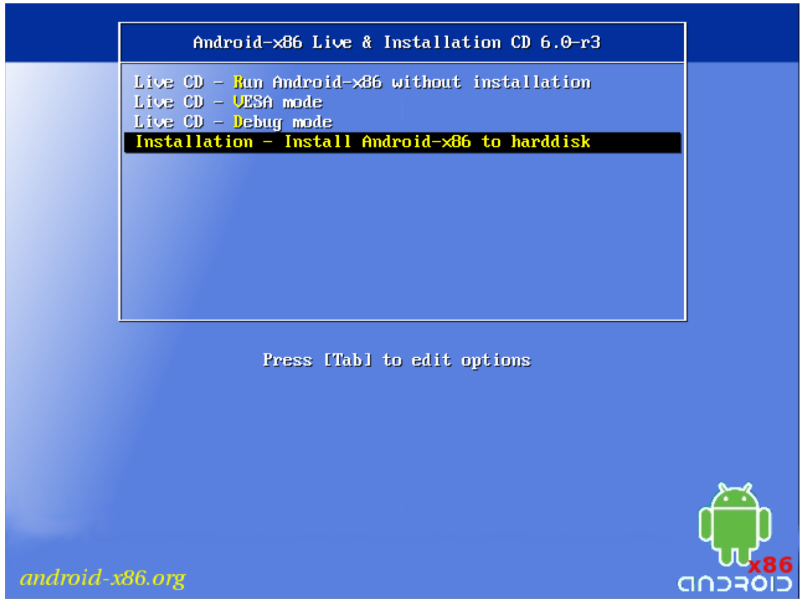


Android installation



Click on this icon, select the Android image you downloaded

Start VM!



Instructions on how to partition the disk and complete the installation:

https://www.howtogeek.com/164570/how-to-install-android-in-virtualbox/

Note: as part of the installation, you'll be requested to create/configure a Gmail account. I recommend not to use your personal account, and create a dedicated account instead.

TinyCore VM:

- You will download the VM image (no need to perform installation) from Canvas (”gateway_vm.ova” under Files/Project material).
  - You will need to import it in VirtualBox
- You will then need to perform various configuration steps.

TinyCore VM - 1

- You must configure your VM so that it is possible to SSH into it.
- **Step 1:** you must configure the Virtualbox NAT interface (Adapter 1) so that it port-forwards from port 12345 of the host to port 22 of the guest VM:
  - Open the VM network settings, select Adapter #1, click port-forwarding.
  - Add a port forwarding rule from port 12345 of the host to port 22 of the guest.

TinyCore VM - 2

- **Step 2:** install OpenSSH
  - Run run **`tce-load -wi openssh`** in the VM
- **Step 3:** create a configuration file for SSH
  - Go into **`/usr/local/etc/ssh`**, run the command **`sudo cp sshd_config.orig sshd_config`**
- **Step 4:** Create a password for the user tc.
  - Run **`sudo passwd tc`**. You will be asked to insert and confirm a password; use *cs4516*

TinyCore VM - 3

- **Step 5.** TinyCore uses a RAM-based filesystem, so you need to store the changes you just made (otherwise they will be lost on reboot):
  - Edit list of files to be backed up:
    `sudo vi /opt/.filetool.lst`
  - Append the following lines to the file:
    `/usr/local/etc/ssh`
    `/etc/shadow`
  - Backup changes: run **`filetool.sh -b`**
- **Step 6.** Ensure that SSH starts at boot.
  - Edit boot commands: **`sudo vi /opt/bootlocal.sh`**
  - Append the following line to the file:
    `/usr/local/etc/init.d/openssh start`

TinyCore VM - 4

- Now, reboot (**`sudo reboot`**)
- You should be able to SSH from the host into the VM, e.g.: **`ssh tc@localhost -p 12345`**

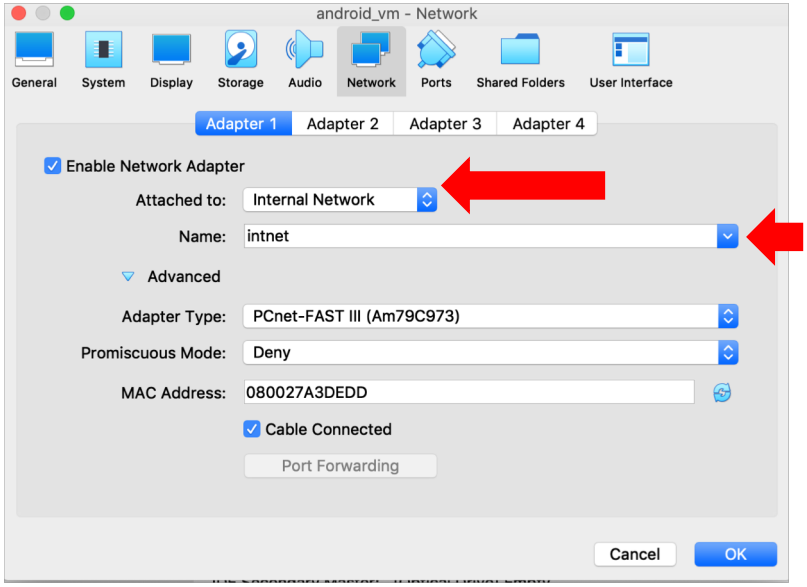
TinyCore VM -5

- Then, you will need to:
  - Configure the eth1 interface with static IP 192.168.12.1, netmask 255.255.255.0. You must persist this change, i.e., every time the machine boots this configuration must be deployed.
  - Configure the VM to behave as an IP gateway. Before you do anything, however, you will need to install the iptables package (packages on TinyCore Linux can be installed with **`tce-load -wi <package name>`**).
  - Configure the VM to act as DHCP server.

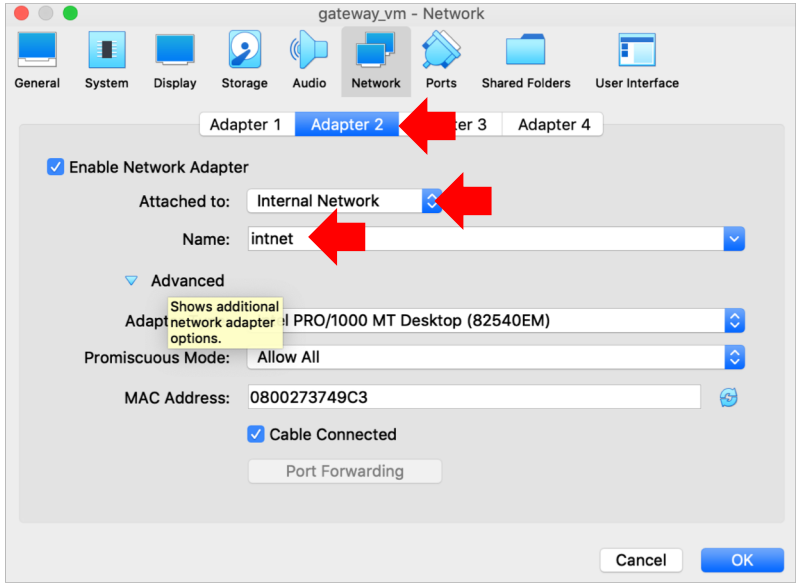
Putting it all together

- Finally, you must configure the Android VM to use the gateway VM to reach the Internet.

On the Android VM:



On the gateway VM:



Putting it all together - 2

- Now, to check if things work:
  - Boot the gateway VM first. When completed,
  - Boot the Android VM
- The Android VM should be able to access Internet. If not, there is a configuration error either in VirtualBox or in the gateway

Delivering your assignment

- Use Virtualbox's "Export appliance" function to export your gateway VM to an .ova file
- Rename your ova file to "cs4516d19_phase1_<team ID>.ova"
- Upload it to the "project phase #1" assignment on Canvas
- It is enough that one person per team performs the upload!