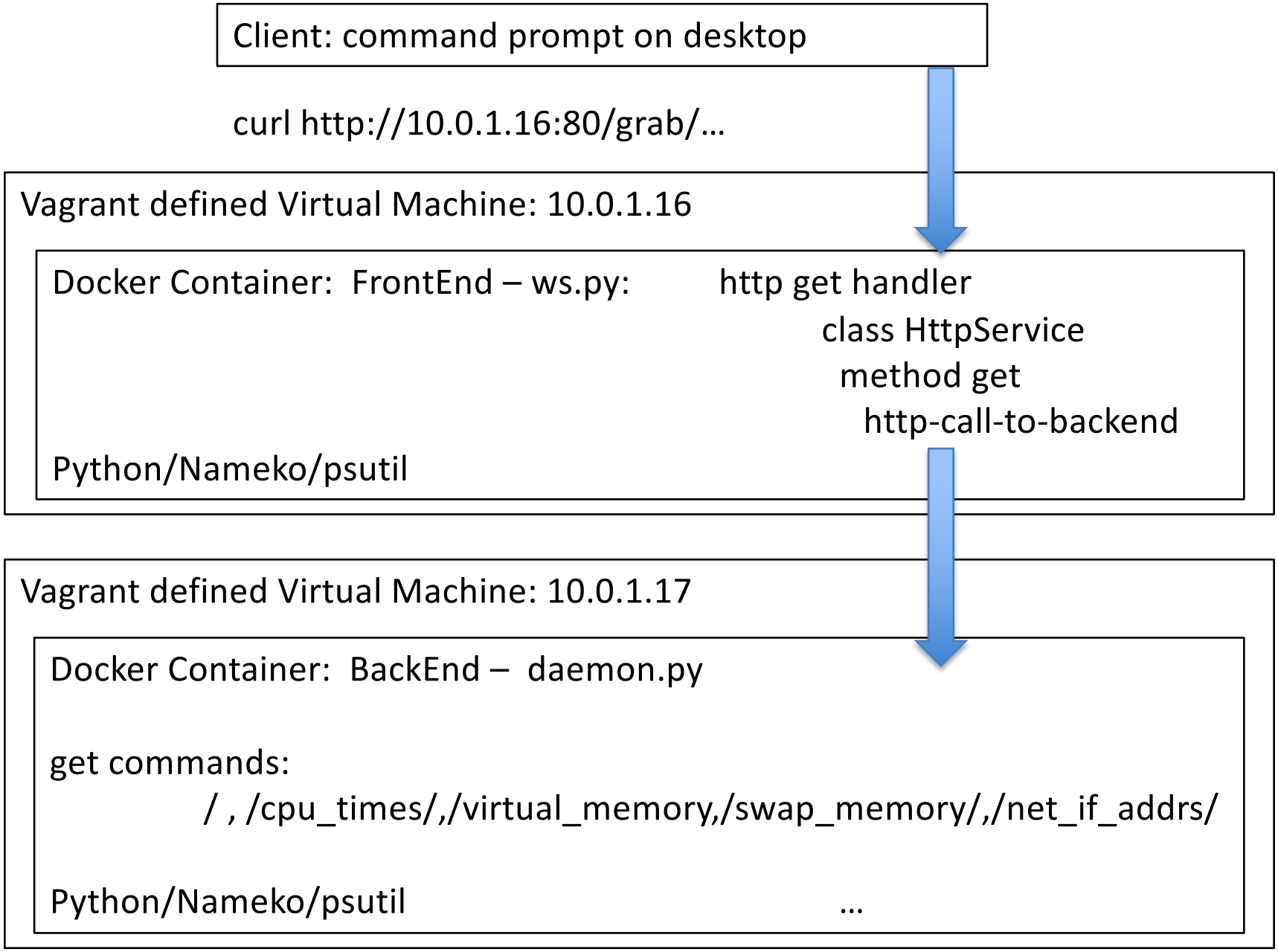
Mini-Project:
Vagrant, Microservices,Docker

Vagrant, Microservices,Docker

- In this section we will deploy a (very) mini 3-tier application.
- The focus will be on reading code – we have covered most of the topics related to Bash, Vagrant, and Docker we need to get started.
- The diagram on the following slide depicts what we are going to deploy

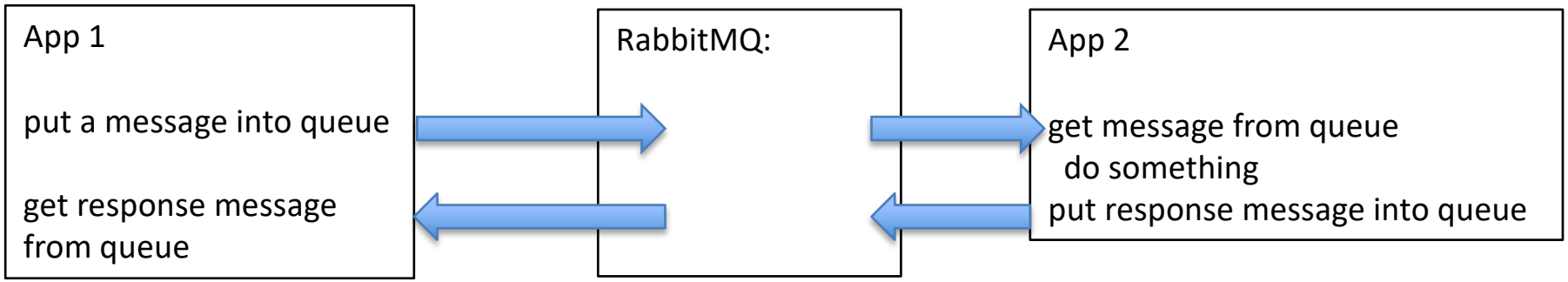

Client: command prompt on desktop
curl http://10.0.1.16:80/grab/…
Vagrant defined Virtual Machine: 10.0.1.16
Docker Container: FrontEnd – ws.py:
http get handler
class HttpService
method get
http-call-to-backend
Python/Nameko/psutil
Vagrant defined Virtual Machine: 10.0.1.17
Docker Container: BackEnd – daemon.py
get commands:
/ , /cpu_times/,/virtual_memory,/swap_memory/,/net_if_addrs/
Python/Nameko/psutil
…

What is RabbitMQ

Reference:

https://www.cloudamqp.com/blog/2015-05-18-part1-rabbitmq-for-beginners-what-is-rabbitmq.html

RabitMQ is message-queuing software. Applications can communicate with each other, over networks and the internet, by exchanging messages using RabbitMQ:



RabbitMQ

- For more information about message-queuing software see link on previous page and/or do a web search on message queues.

- For this presentation all we really need to deal with is:
  - getting RabbitMQ setup to enable communications between the FrontEnd microservice and the backend microservce

Example Usage

- The following slides show our mini-n-tier application running.
- From a command console on the host laptop/desktop

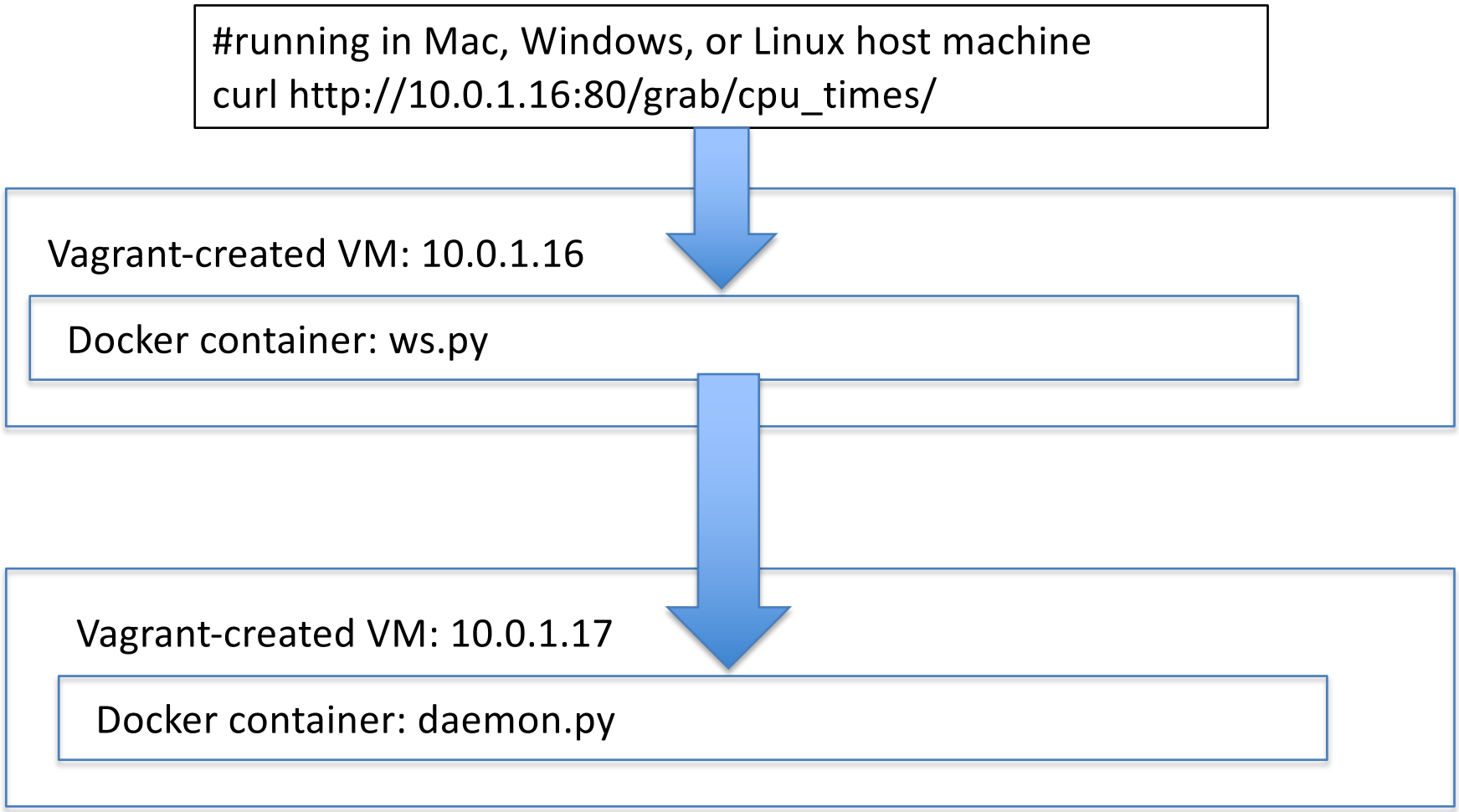
  curl is used to issue an http request to the FrontEnd

Example Usage

- When the FrontEnd gets the request, it calls the BackEnd

- Based on the uri (cpu_times, virtual_memory, etc.), the backend makes a call into a method in python library psutil

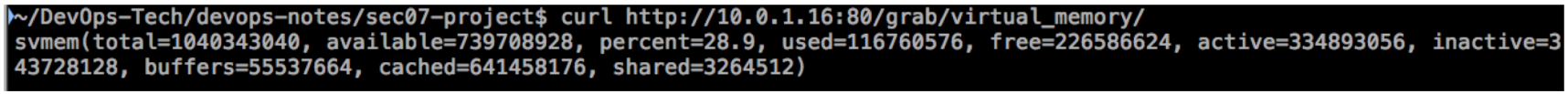
Example Usage



Example Usage

mac:
curl http://10.0.1.16:80/grab/virtual_memory/

svmem(total=1040343040, available=739733504, percent=28.9,
used=116736000, free=226586624, active=334917632, inactive=343728128,
buffers=55562240, cached=641458176, shared=3264512)



Example Usage

Windows cygwin: curl http://10.0.1.16:80/grab/virtual_memory/

```
u030077@USPLEC7JYH12 /cygdrive/c/jeff/DevOps-Tech/devops-notes/sec7-example
$ curl http://10.0.1.16:80/grab/virtual_memory/
svmem(total=1040322560, available=777289728, percent=25.3, used=117018624, free=689143808, active=185749504, inactive=110157824, bu
368, cached=219893760, shared=3629056)
```

Setup

- To setup the FrontEnd and BackEnd VMs
- Install Docker on both VMs
- Install Python, Nameko, and psutil in each Docker container

- You can run install.sh

Setup

- install.sh – which is very simple:

  cd backend

  vagrant up

  cd ../frontend

  vagrant up

Installation Notes

- **On a Mac**, when running install.sh, everything worked

- **For Windows**, Cygwin Specific setup to run install.sh – see supplemental notes:

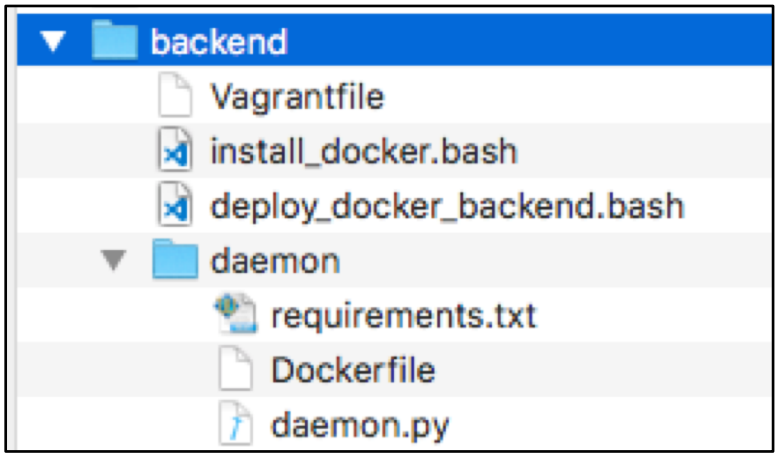
  ex07-cygwin-notes

  important:
  - you will need to alias sudo to nothing
  - get curl using the Cygwin installer

Installation Notes

- If you rerun install.sh
  - delete the .vagrant folders in the back and front end folders
  - if you make see "port in use" messages, trying using "vagrant reload" in place of "vagrant up"

Backend


backend
Vagrantfile
install_docker.bash
deploy_docker_backend.bash
daemon
requirements.txt
Dockerfile
daemon.py

Backend

- **Vagrantfile Summary**
  - config.vm.network "private_network", ip: "10.0.1.17 "
  - an inline Bash provisioner, "install_curl"
  - an inline Bash provisioner – "install_docker" – that invokes an external script **install_docker.bash**
  - an inline Bash provisioner , "install_backend", the builds and runs the daemon Docker image in the VM using an external script – **deploy_docker_backend.bash**

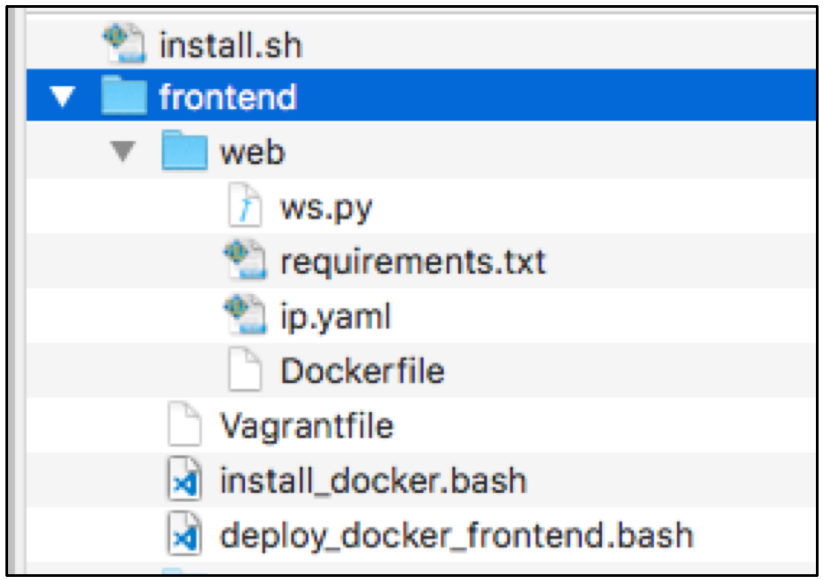
Backend

- **install_docker.bash**
  - using apt-get install docker in the backend VM
- **deploy_docker_backend.bash**
  - builds a Docker image from the Dockerfile in backend/daemon
  - starts a Docker container, using the Docker image built in the previous step

Backend

- **Dockerfile**
  - add the backend/daemon into the container
  - install psutil and Nameko into the container from requirements.txt

Frontend


install.sh
frontend
web
ws.py
requirements.txt
ip.yaml
Dockerfile
Vagrantfile
install_docker.bash
deploy_docker_frontend.bash

Frontend

- Vagrantfile
  - config.vm.network "private_network", ip: "10.0.1.16"
  - install_curl provisioner , same as backend
  - install_docker provisioner, same as backend
  - install_frontend provisioner, an inline Bash sccript that invokes external script deploy_docker_frontend.bash

Frontend

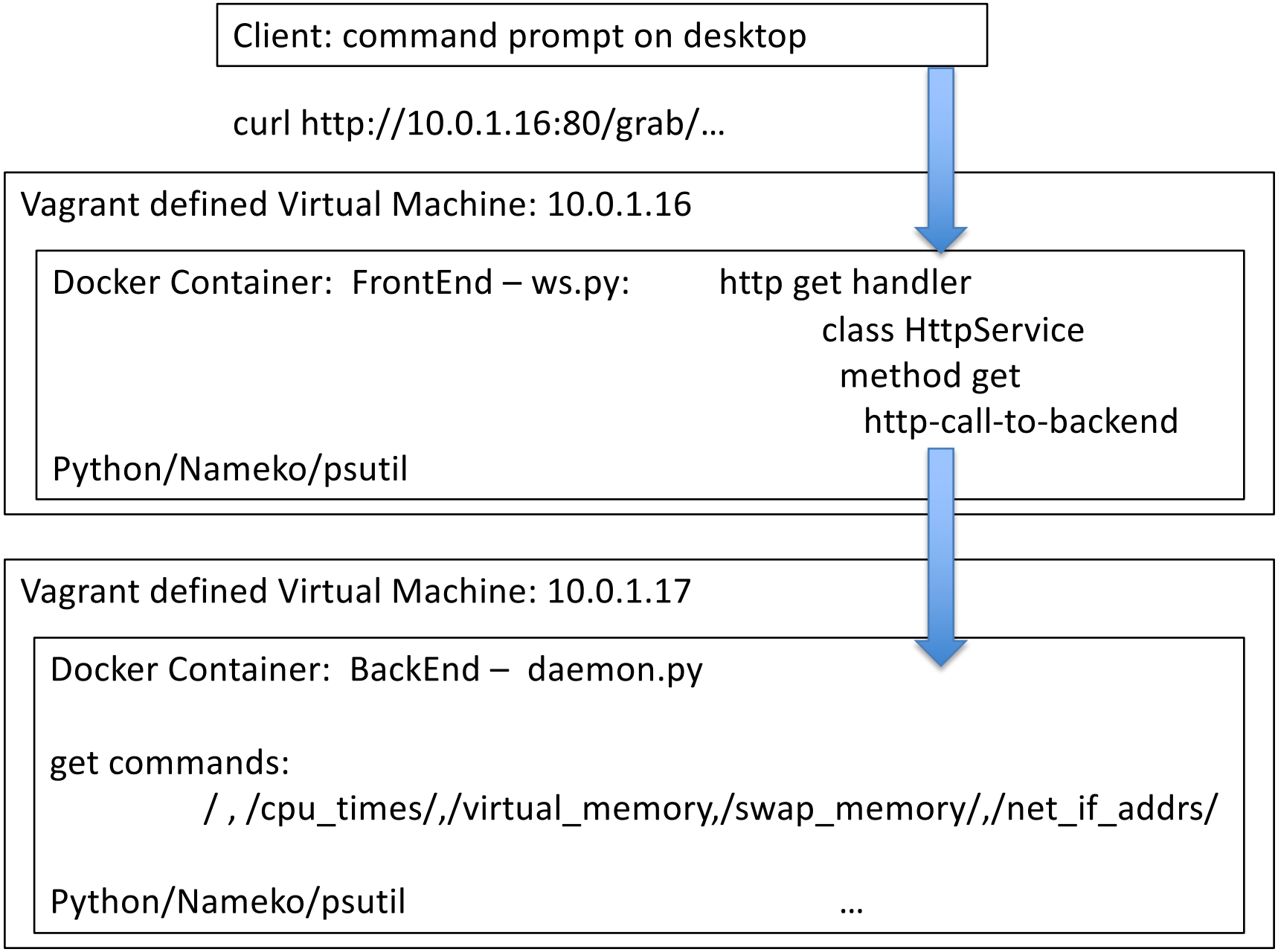
- **install_docker.bash**
  - using apt-get install docker in the backend VM
- **deploy_docker_frontend.bash**
  - builds a Docker image from the Dockerfile in frontend/web
  - starts a Docker container, using the Docker image built in the previous step

Frontend

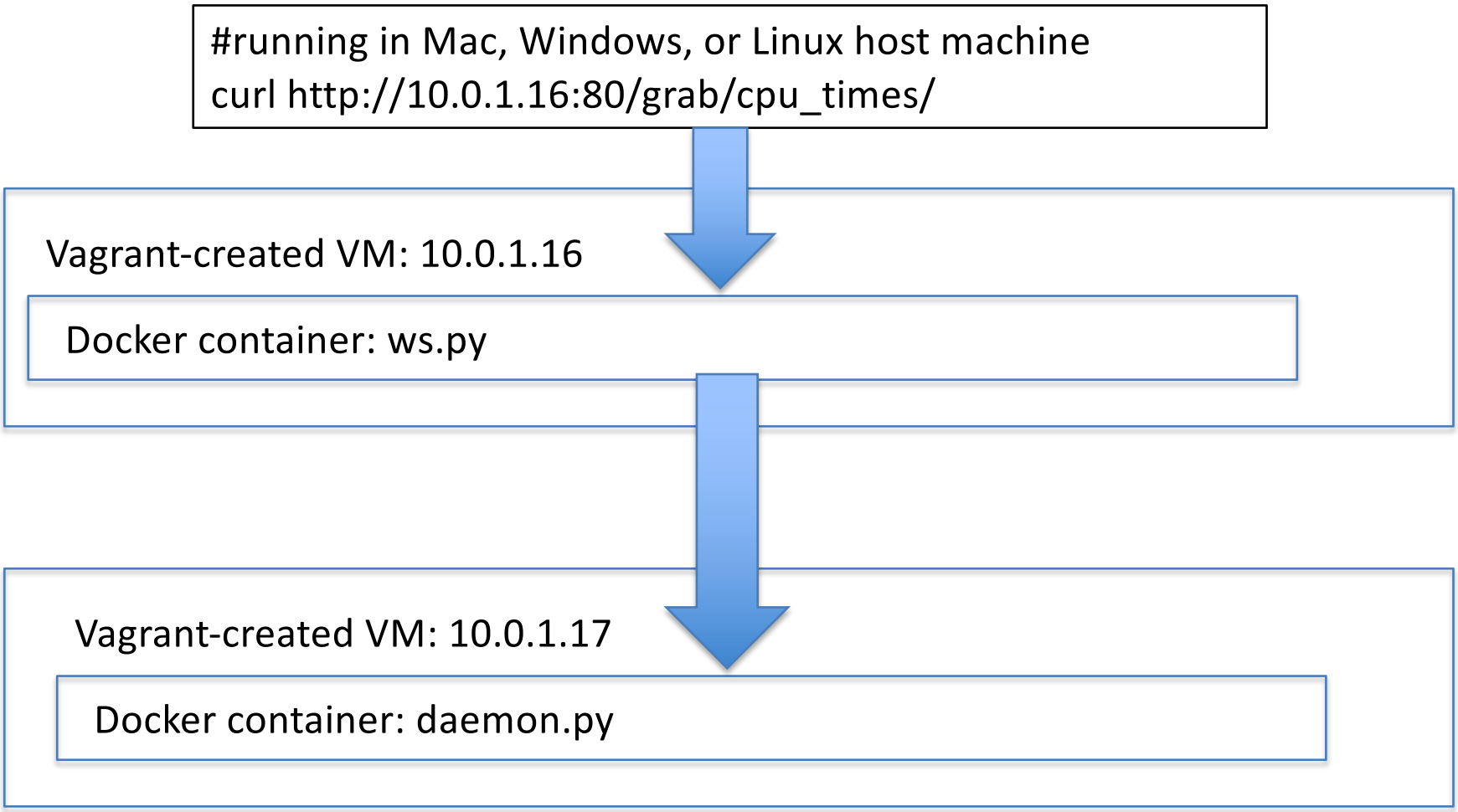
- **Dockerfile**
  - Add folder backend/web into container
  - Install Nameko and psutil (psutil may not be needed) as listed in requirements.txt
  - Run Nameko and our frontend

Client

- On a Mac, just run curl
- On Windows, in a Cygwin Bash shell, run curl


Client: command prompt on desktop
curl http://10.0.1.16:80/grab/…
Vagrant defined Virtual Machine: 10.0.1.16
Docker Container: FrontEnd – ws.py:
http get handler
class HttpService
method get
http-call-to-backend
Python/Nameko/psutil
Vagrant defined Virtual Machine: 10.0.1.17
Docker Container: BackEnd – daemon.py
get commands:
/ , /cpu_times/,/virtual_memory,/swap_memory/,/net_if_addrs/
Python/Nameko/psutil
…

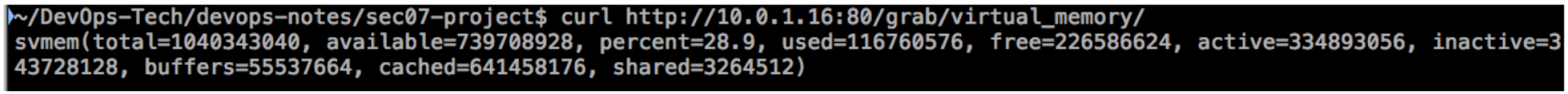
Example Usage



Example Usage

mac:
curl http://10.0.1.16:80/grab/virtual_memory/

svmem(total=1040343040, available=739733504, percent=28.9,
used=116736000, free=226586624, active=334917632, inactive=343728128,
buffers=55562240, cached=641458176, shared=3264512)



Example Usage

Windows cygwin: curl http://10.0.1.16:80/grab/virtual_memory/

```
u030077@USPLEC7JYH12 /cygdrive/c/jeff/DevOps-Tech/devops-notes/sec7-example
$ curl http://10.0.1.16:80/grab/virtual_memory/
svmem(total=1040322560, available=777289728, percent=25.3, used=117018624, free=689143808, active=185749504, inactive=110157824, bu
368, cached=219893760, shared=3629056)
```