Microservice Patterns

Definitions

- Instance : A microservice application.

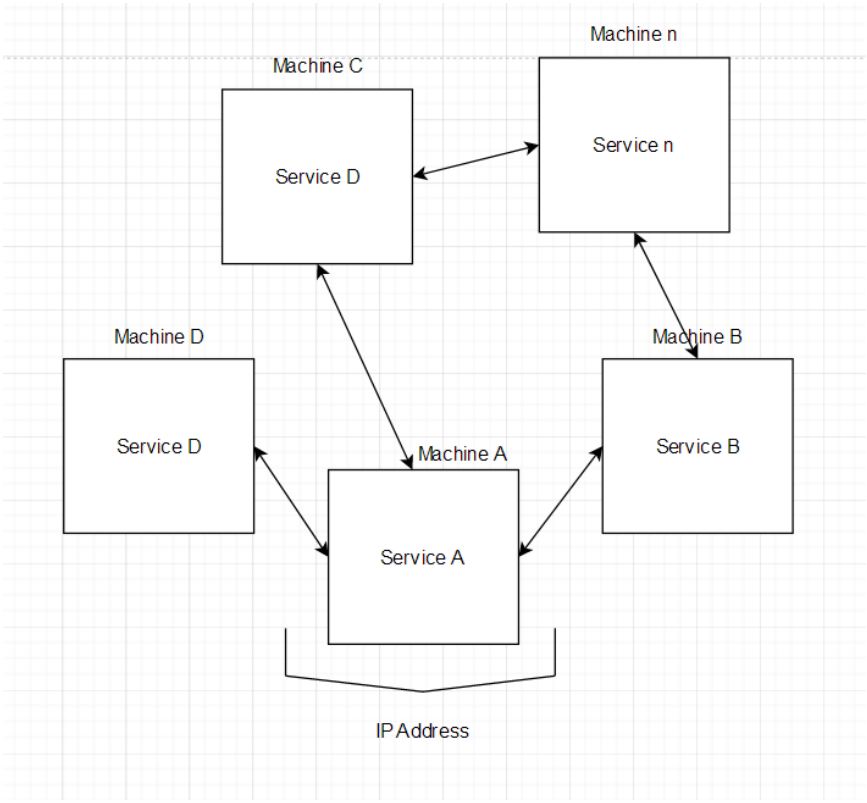
Relevant Links

- [The article for different patterns]
- [Different sizes and pricing]

---

1. **[Single Instance per host Pattern]**

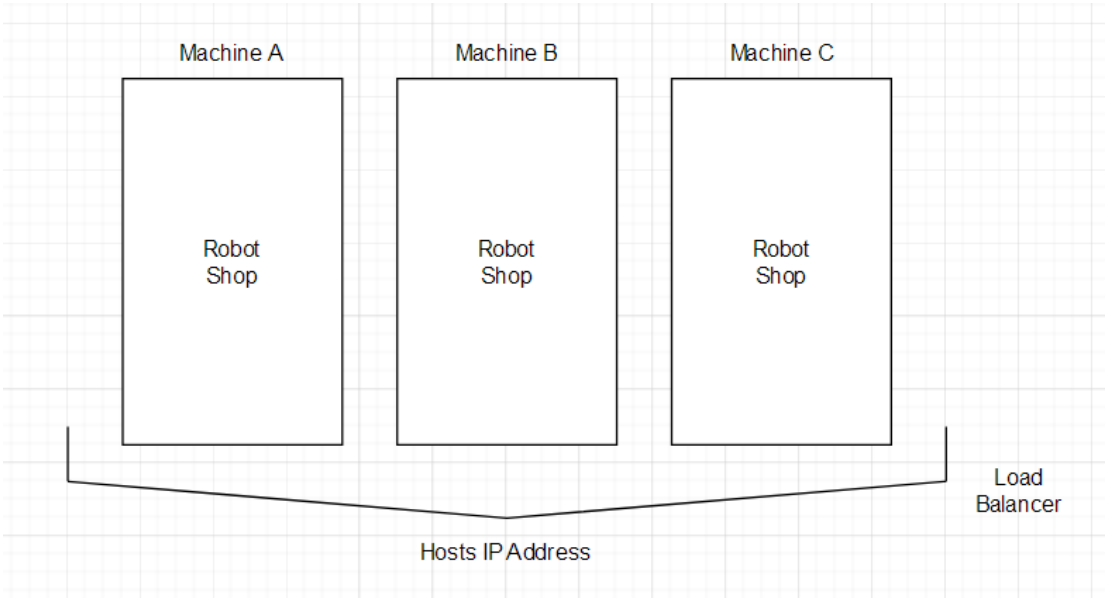
    1. Each service instance is deployed on its own host



Each instance is deployed differently, each instance can scale as it needs, i.e if service D is used less than service B, B could use more resources than D

2. Single / **Multiple services instance(s) per host pattern**

    1. Provision **one or more** virtual machines and run multiple instances on each one. (For our purposes we run a single application on a single machine)
        1. This method adopts running multiple machines (A, B, C) and attaching a load balancer onto it which servers it's own IP address for the load generator to use. Definition : "Load balancing provides a higher level of availability by spreading incoming requests across multiple virtual machines."

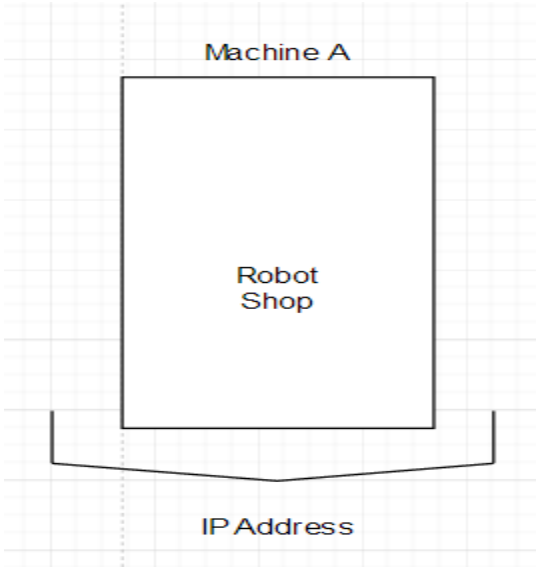


Relevant links - azurerm_lb , Tutorial: Load balance VMs for high availability

3. Server instance per VM
    1. Package the service as a virtual machine image and deploy each service instance as a separate VM
4. Service instance per container
    1. Package the service as a (Docker) container image and deploy each service instance as a container

Some edited versions of each. (Versions I'd potentially deploy for the project - in order)

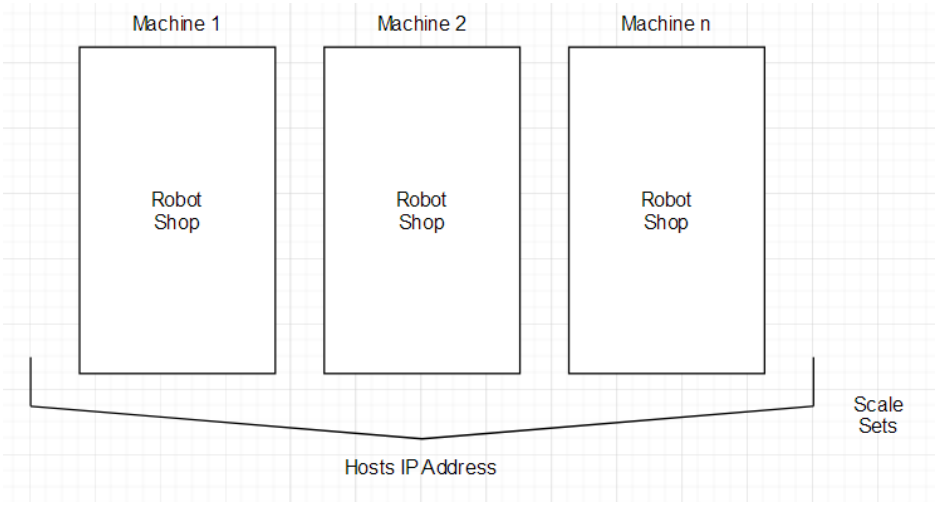
1. Single container instance per machine.



Current Pineapple application example.

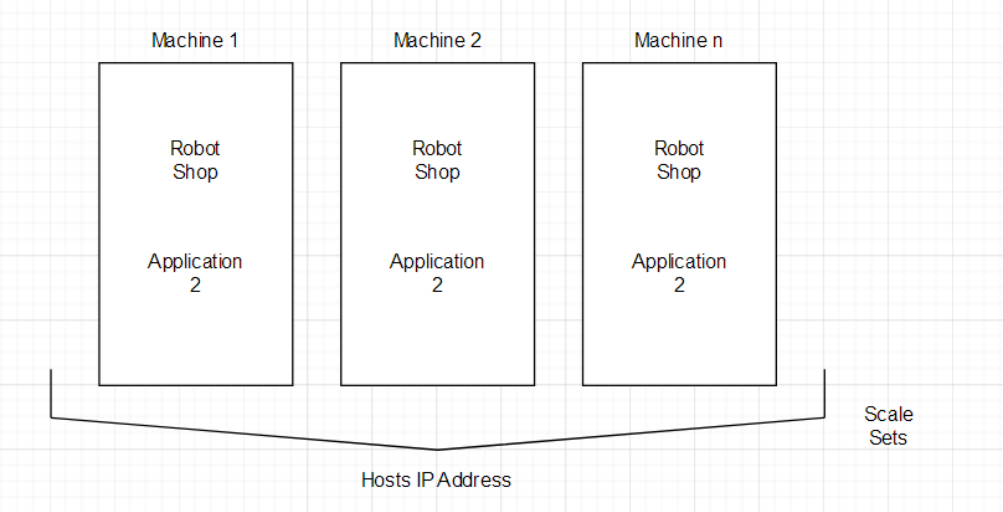
2. Single / **Multiple services instance(s) per host pattern**

    - This method adopts running a scale set. Definition : "Virtual machine scale sets with Flexible orchestration let you create and manage a group of load balanced VMs. The number of VM instances can automatically increase or decrease in response to demand or a defined schedule." (This might be the easier one to work with)



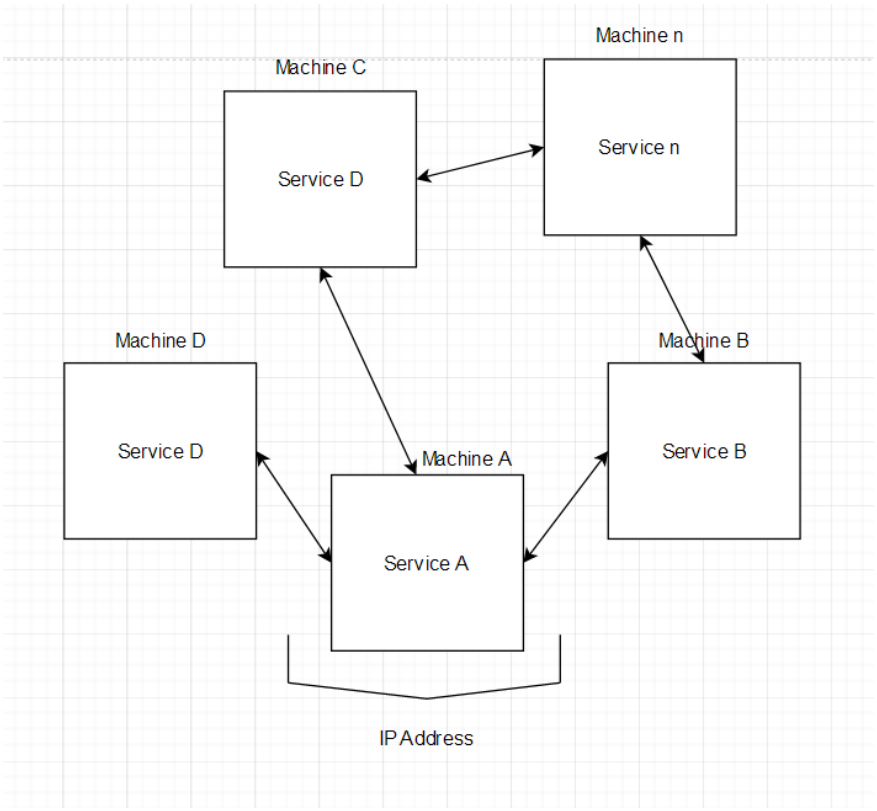
Relevant links - [azurerm_virtual_machine_scale_set](), [Create a virtual machine scale set]()

- Host multiple applications, on multiple machines. (Multiple Services Instance per Host).



3. **[Single Instance per host Pattern]() (This would be a nice pattern to deploy as it takes true advantages of the architecture, possibly the hardest to automate.)**

    - Each service instance is deployed on its own host



Each instance is deployed differently, each instance can scale as it needs, i.e if service D is used less than service B, B could use more resources than D

Objectives

1. Complete the 3 microservice patterns by start of February.
    1. These should be able to run simultaneously
        1. 100 or so metrics on various VM options running in one go, collecting data
    2. (Robot shop only) Use loctus.py to provide accurate API metrics - Not sure if this is impossible yet
    3. Set up a sustainable method to save logs.
        1. from API Metrics
        2. from Infrastructure azureCLI
    4. Complete **pattern 1** - Terraform script - by the 25th
    5. Complete **pattern 2** - m**id January**
    6. Complete **pattern 3** - **January end**
2. Set-up second application for Pattern 1, this should essentially not take a lot of time, only if the application works as intended out of the box. (Application set-up already, need to set up work-load.)
3. Start writing dissertation - now
    1. can possibly complete sections that talk about methodology, the different patterns we'd like to use and more, data analysis would be left out for later.
    2. How different patterns react with different type of applications (Resource intensive,
4. Data Analysis
    1. Currently I am unsure how to analyze data, or what most of it means in terms of metrics, I assume lower means better in some cases, but there should be more scientific terms which I'd need to learn.