# Cloud Deployment

## Containerisation

1. Purpose:
   1. Portability of SMPC
   2. Easy for testing and deployment in cloud environment in consistent way
2. Cons:
   1. Not necessarily needed, can be done without docker.
   2. Some effort needed
3. Docker based deployment and integration of docker is at feature/docker branch of abhi4578 fork of iudx-MOTION2NX

ref: <https://github.com/abhi4578/iudx-MOTION2NX/tree/feature/docker#docker-based-deployment>

1. Integrated remote\_server and remote\_server\_split
   1. Add config options to take from config file
   2. dns resolve if dns names given
   3. Separate Data from code and corresponding the paths in script
   4. Arrange scripts to different directories for ease of understanding
   5. docker-compose file defining two containers - smpc-server0 and smpc-server1
2. Some fixes to be done for local deployment to work in docker → need to be done
   1. Read from config file
   2. dns resolve
   3. change data paths in scripts
   4. merge scripts?
3. Helper node → need to be done
   1. Add configurable -ports, hosts, fractional bits, image id options to take from config file, separate config file for helper node
   2. dns resolve if dns names given
   3. Separate Data from code and corresponding paths in script
   4. Arrange scripts to different directories for ease of understanding
   5. docker-compose file defining three containers
   6. Test it locally
   7. and then can be tested in cloud
4. Docker related materials
   1. [Devops Training: Docker](https://docs.google.com/presentation/d/1mdPmhqLKgH7FVc-9mmcWtzHCoHl8xUJ10MDrB2M11V0/edit#slide=id.ge6fe7eb7ca_0_43) I
   2. [Devops Training: Docker II](https://docs.google.com/presentation/d/1mhLsiP4c2VTWi0LRfWs3JMK-AIGRDR0ITO541LVK21Q/edit#slide=id.p)

## Cloud Deployment Specification

1. **Cloud VM 2 (smpc-server1)**
   1. **public ip: 13.232.226.177**
   2. **private ip: 10.2.1.25**
   3. **username: ubuntu**
   4. **ssh ubuntu@13.232.226.177**
   5. **t2.micro**
      1. **1vcpu, 1GiB, 30 GiB SSD**
   6. **subnet iudx-test-1**
   7. **VPC iudx-test**
2. **Cloud VM3 (smpc-server-helper-node)**
   1. **public ip: 15.206.80.191**
   2. **private ip: 10.2.1.171**
   3. **username: ubuntu**
   4. **ssh ubuntu@15.206.80.191**
   5. **t2.nano**
      1. **1vcpu, 0.5GiB, 30 GiB SSD**
   6. **subnet iudx-test-1**
   7. **VPC iudx-test**
3. **Cloud VM 4 (azure) will act smpc-server0**
   1. **public ip : 20.197.27.117**
   2. **private ip: 10.139.0.4**
   3. **username: ubuntu**
   4. **Login: ssh ubuntu@20.197.27.117**
   5. **b1s** 
      1. **1vcpu, 1 GiB, 30 GiB SSD**
4. Generate ssh public private key
   1. ssh-keygen -b 4096 -f ~/.ssh/iudx\_cloud
   2. passphrase refers to encrypting a private key using a passphrase based key. For convenience, can be ignored and press enter, i.e. without passphrase and hence key would be unencrypted
   3. send the public key to add access to VMs, the file with ‘.pub’ extension, i.e the file at ~/.ssh/iudx\_cloud.pub
   4. **DO NOT SEND the private key**

## Deployment in Cloud

1. Do not build docker images in cloud vm because the memory cpu is small and for build > 2GB > 1 vcpu is required.
2. Build and push from local machine to docker registry and run the docker image from registry in cloud

### SMPC Remote Split

1. To run each server on different machine, git clone this repo on each of the machines and run following, :

| # after copy, replace with appropriate image tag (TBD only once per docker image)  > cp example-docker-compose.remote-registry.yaml docker-compose.remote-registry.yaml |
| --- |

| > # On SMPC server 0  docker compose -f docker-compose.remote\_split.yaml -f docker-compose.remote-registry.yaml up -d smpc-server0 |
| --- |

| > # On SMPC server 1  docker compose -f docker-compose.remote\_split.yaml -f docker-compose.remote-registry.yaml up -d smpc-server1 |
| --- |

#### Performance

Performance of each smpc server when run above specified cloud vms with 1 image inferencing

1. server 0
   1. Memory requirement: 0.451 GB
   2. Time taken by inferencing task: 176544 ms
2. server 1
   1. Memory requirement: 0.451 GB
   2. Time taken by inferencing task: 176398 ms

#### Issues

1. Unable to bind ip in inference only at server 0 when u bind docker port to machine. The error log is as follows

| ERROR OCCURRED: bind: Cannot assign requested address [system:99] Layer 1, split 1 - multiplication is done. terminate called after throwing an instance of 'std::invalid\_argument' what(): stoull ./remote\_server0\_split.sh: line 173: 62 Aborted (core dumped) $build\_path/bin/appendfile 0 ERROR OCCURRED: bind: Cannot assign requested address [system:99] Layer 1, split 2 - multiplication is done. terminate called after throwing an instance of 'std::invalid\_argument' what(): stoull ./remote\_server0\_split.sh: line 173: 66 Aborted (core dumped) $build\_path/bin/appendfile 0 |
| --- |

1. Argmax by default takes two threads and this won't work in one virtual/logical core cpu machine. Error logs as follows

| terminate called after throwing an instance of 'std::invalid\_argument' what(): FiberThreadPool needs at least two worker threads ./remote\_server0\_split.sh: line 216: 206 Aborted (core dumped) $build\_path/bin/argmax --my-id 0 --party 0,$cs0\_host,$cs0\_port\_inference --party 1,$cs1\_host,$cs1\_port\_inference --arithmetic-protocol beavy --boolean-protocol beavy --repetitions 1 --config-filename file\_config\_input0 --config-input $image\_share --current-path $build\_path > $debug\_0/argmax0\_layer2.txt |
| --- |

1. Docker Images built in [Srishti Mittal](mailto:srishti.mittal@datakaveri.org) laptop and when its executed in cloud vms , it gives following error

| ./remote\_server1\_split.sh: line 150: 44 Illegal instruction (core dumped) $build\_path/bin/tensor\_gt\_mul\_split --my-id 1 --party 0,$cs0\_host,$cs0\_port\_inference --party 1,$cs1\_host,$cs1\_port\_inference --arithmetic-protocol beavy --boolean-protocol yao --fractional-bits 13 --config-file-input $input\_config --config-file-model file\_config\_model1 --layer-id $layer\_id --row\_start $a --row\_end $b --split $splits --current-path $build\_path > $debug\_1/tensor\_gt\_mul1\_layer1\_split.txt |
| --- |

* Illegal instruction is due to mainly it was built with an instruction set unique to the laptop and not supported cloud vm chip. Ref: [(C++) Getting error "Illegal instruction (core dumped)" upon bitwise OR operation - Stack Overflow](https://stackoverflow.com/questions/47899971/c-getting-error-illegal-instruction-core-dumped-upon-bitwise-or-operatio)
  + The docker build ensures
    - same environment for build
      * os
      * libraries and version
      * packages and version
      * same instruction set for motion2nx build (AVX2)
      * gcc, g++ compiler
  + Might be following issue
    - either boost or eigen library is built with some special instruction i.e. not portable or cross platform. The gcc/g++ compiler taking certain hardware defaults while building
    - or somehow motion2nx is using some other hardware specific instruction other than avx2

### Server model

#### Experiment 1

* The compute servers on two cloud vms. The image provider and weights provider in local laptops.
* Because local laptops are behind NAT. The compute servers can’t reach the image provider to share final output.
  + Hence reverse tunneling is required through a separate cloud vm ( we assume that this VM is controlled by image provider)
    - Reverse tunneling materials:

1. ssh :<https://www.youtube.com/watch?v=ORcvSkgdA58>

2. ssh tunnel explained:<https://goteleport.com/blog/ssh-tunneling-explained/>

3. will use autossh to persist this ssh reverse tunnel : -<https://linux.die.net/man/1/autossh>, and cloud vm being analytics.iudx.org.in

4. understanding ssh connection, key based authentication and encryption process, <https://www.digitalocean.com/community/tutorials/understanding-the-ssh-encryption-and-connection-process>

5. What is Network address translation (NAT)? how it works and different types? All routers generally use the

Port address translation or NAT overload or one-to-many NAT. The NAT idea is to conserve public IPv4 address as its limited and enable devices with private IP connect to internet, . Ref:<https://www.geeksforgeeks.org/network-address-translation-nat/>

* How is the reverse tunneling setup?
  + created a user called “sshtunuser” with no shell login access, i.e. can only do reverse tunneling and configured sshd config as described in the article to allow reverse tunnel . see [setting up environment](https://cnly.github.io/2018/08/16/setting-up-autossh-to-maintain-a-reverse-tunnel-ssh-server-having-a-dynamic-ip-address.html)
* Reverse tunneling setup
  + First, check if possible to ssh using below command, u will be see some stats of server and will exit. this is expected as u dont have shell access for autossh machine

| > ssh -i ~/.ssh/iudx\_cloud sshtunuser@analytics.iudx.org.in |
| --- |

* + Open one terminal and reverse tunnel to receive image output shares from server 0

| > autossh -M 23323 -N -q -o "ExitOnForwardFailure yes" -o "ServerAliveInterval 10" -o "ServerAliveCountMax 3" -R 4007:localhost:4007 sshtunuser@analytics.iudx.org.in -i ~/.ssh/iudx\_cloud |
| --- |

* Open another terminal and reverse tunnel to receive image output shares from server 1

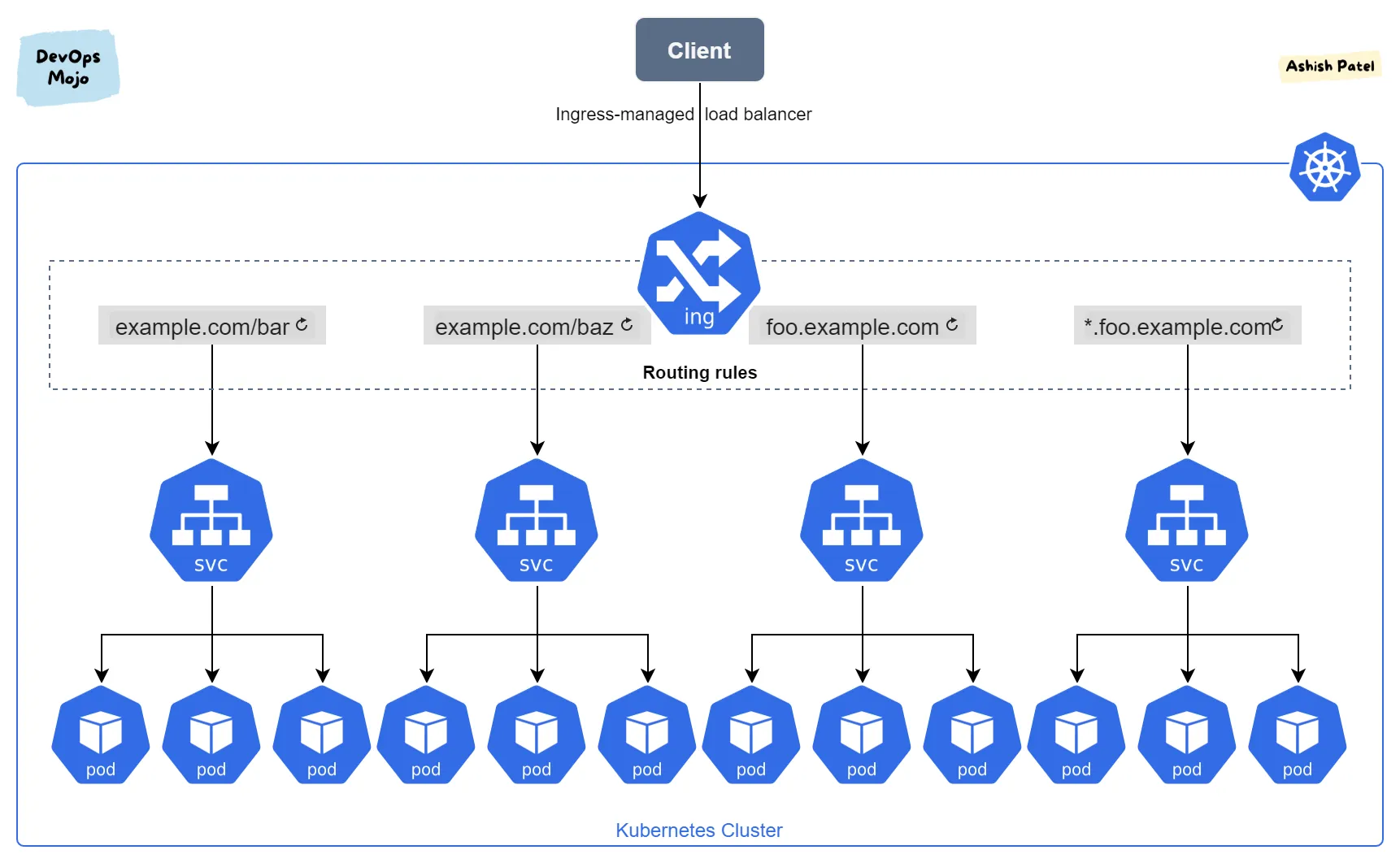
| > autossh -M 23325 -N -q -o "ExitOnForwardFailure yes" -o "ServerAliveInterval 10" -o "ServerAliveCountMax 3" -R 4008:localhost:4008 sshtunuser@analytics.iudx.org.in -i ~/.ssh/iudx\_cloud |
| --- |

* Run the compute servers on cloud, in the config for image provider ip , mention the public ip of the separate cloud vm
* Run the image provider in the laptop where reverse tunnel is established
* Run the weights provider in another laptop

### 

## Carbyne stack and K8s Cluster

### Ingress



<https://medium.com/devops-mojo/kubernetes-ingress-overview-what-is-kubernetes-ingress-introduction-to-k8s-ingress-b0f81525ffe2>

### CLI

what is it?

How it works?

### SPDZ <https://github.com/data61/MP-SPDZ>

* try out tutorials -
  + see if there is Neural network tutorial
  + ask on gitter
* ask on carbyne discord - to integrate with carbyne

### Terraform

* Infrastructure as a code
  + for cloud , bare metal server
    - VPC - private network
    - VMs
    - Security
    - Bring up cloud services
  + CDKTF
    - Bring K8s as code