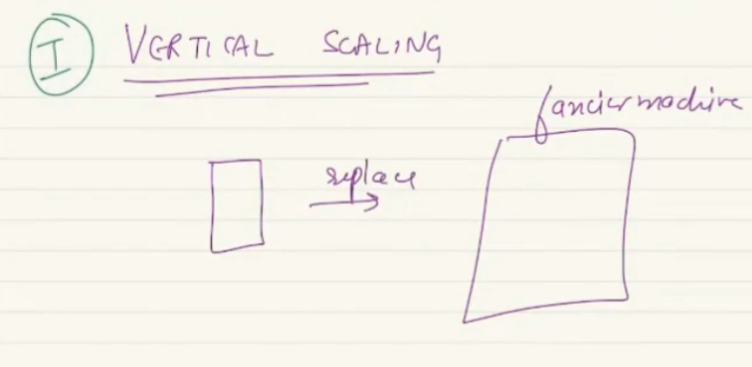


Vertical = replace current with powerfull single machine. Better ram,storage

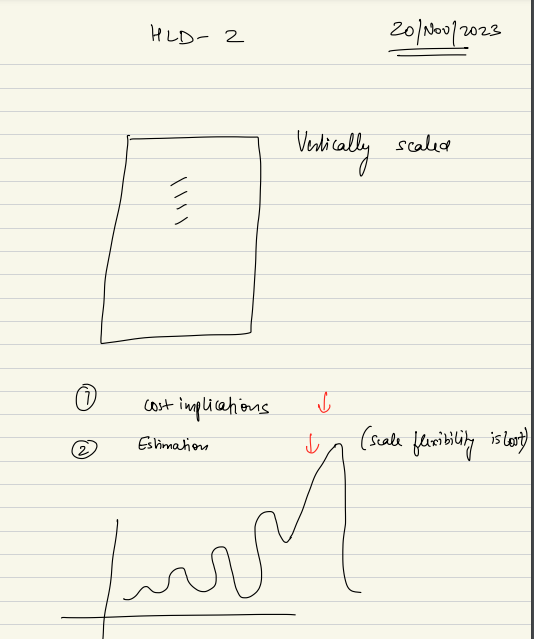


Replace with much more compitant machine. Great but still has limitation.

Rather than employing 1 machine, replace with cluster of machines. May be a bit less competent pawn machines. They can pull resource together.

When I have a vertically scalled machine, biggest adv: only have single machine, code will be deployed in just 1 machine. Problem in VS is ..

1. There are cost implication, buying a super computer is costlier than buying 4 sinple day to day computer. Spend a lot of money to get a vertically scalled machine.
2. A lot of products called FRAD product, Clubhouse appp: people can get in a voice room and talk to each other. Became success. Twitter has spaces.. so no one uses clubhouse. The rate they became quick success better rate they die. Facebook thread.. they are frad product. In real life no one can pedict how success or unsuccfull product will be.. a product will not have steady graph. Diff to predict how much resource I need 5 month down.. can be huge demand or a big flop.. so invest million of dollar in a machine is waste. We cant pre-comment on estimation on vertically scaling. A big machine – cost of running is also high. Commiting in a machine means scale flexibility is lost.
3. If I replace a smaller machine to a bigger machine- SPOF is not solved. If machine goes down nothing can be done.

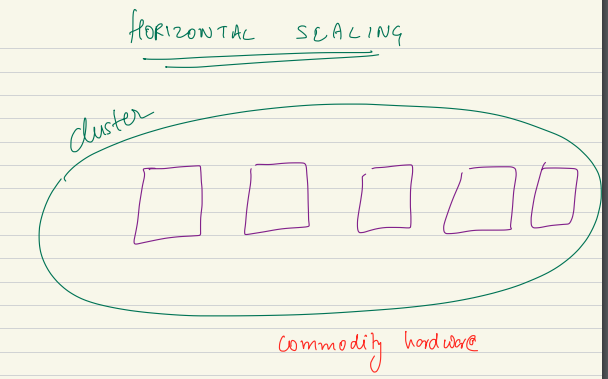


We will look at what are scaling challenge in cloud.

**Horizontal Scaling:**

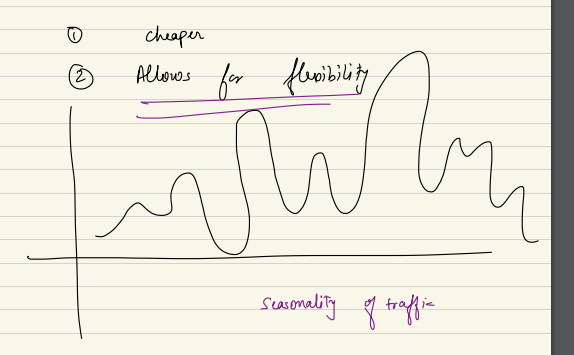
In case of HS we replace a single machine we replace with cluster of machines. To scale we will Buy a simple small cheap machines, and use them as cluster of machines, they pull the resources and will get the job done at higher scale. Cluster = bunch of machines, each can be simple cheaper machine.

These cheap machine is known as Commodity hardware – simple hardware with very fixed specs, cheaply available at market as created in high rate. Very common specs.



**Adv of HS:**

1. Get job done in cheaper price.
2. Allows for lot of flexibility. = if my req / activity reduce I can switch off some machine and save electricity bill. And we can add more machine when I need. Example: amazon is transformational business. Changed the world. Amazon has a business model if different. Their scale changes a lot throughout the year. During black Friday: everyone goes. People shopping rate is higher. The graph has high and low throughout the year. Seasonality of traffic. Traffic doesn’t stay constant. it changes with diff time on a day. Imagine Instagram has a time to go viral..



If I buy a big machine I can scale down…

If I have cluster of small machine based on traffic I can increase and decrease.

Amazon knew they had to buy a lot of machine as some day the traffic will grow. When more people login at same time, bnackend has to do a lot of computation, reads, writes. So load is propotional to user logged in.

Cbsc website me result k time site crash. Irctc used to crash a lot. Flipkart sales goes down.. if you don’t scale backend it will go down.

Amazon had to keep a lot of resource so when traffic goes up they can plug them. Otherwise they are idle. This led to creation of cloud computing. Amazon realize a lot of hardware keps idle. So they rent out hardware as aws. Idea is idle computer not used throughout the year. So rent them.. this is cloud computing.

In simply rather than using your own datacenter you pay someone to pay rent for your computation.

IRS interview of cloud computing. When it rains …

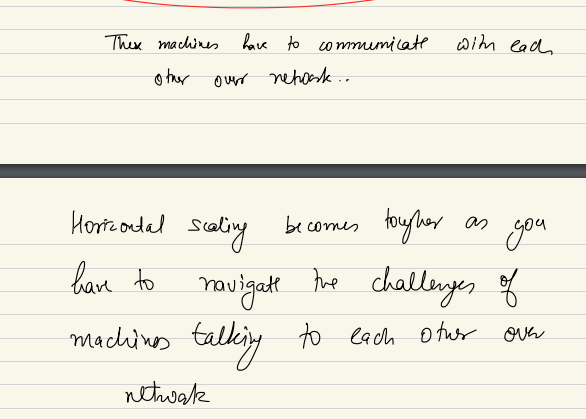
HLD contruct is fundamental to cloud computing to work. Here adv is no single point of failure.

When I have cluster of machines if one machine goes down everything is fine. Adv of cloud computing.

**Disadvantage of Horizontal :**

Your code run on multiple machines, these machines have to pull the resource and use each other’s data, disk, memory, resource. Also they need to talk to each other, communicate with each other. Talk means: make API calls, distribute data themselves, read-write themselves.. We need to deploy code in multiple machines. Also these machines have to **communicate with each other over network**. **Network is never realizable.**

Network is not reliable. 2 machine cant talk to each other means underlying network is down. Your interaction will not reliable. Lot of issues CAP. So biggest downside in machine need to talk to each other and they talk on network. And network is not reliable.



Need lot of engineering effort to put in to ensure the system is up and running... Spend lot of resource to make them talk to each other.

**Delicious problem**: with managing the scale.. they need to scale. Don’t have lot of money. They are growing twice month on month. They don’t know what to buy. They don’t have proper estimation. They have to do horizontal scaling rather than using single machine they bought many machines..

Aws is just implementation of these concepts.

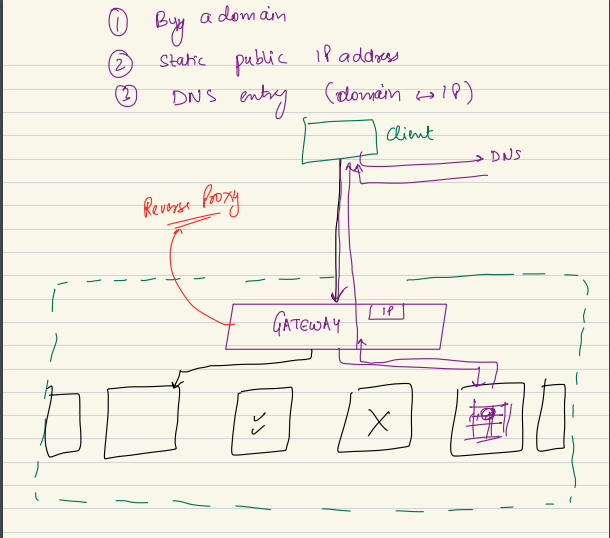
So they employed bunch of machines as horizontal scale. Earlier the bought a domain. And bought static public IP.

Made dns entry on domain and IP they bought.

Previously they had one machine. That time he had one IP. Now I have 4 machines. So I want to abstract the idea of I have 4 machine running. Also cant write machine entry in DNS. So I use gateway machine.

**Gateway:** a simple machine which will run on IP address I registered. It takes request and forward the request to other backend machines. They do computation. But the gwy is take a request and forward to the app server machine.

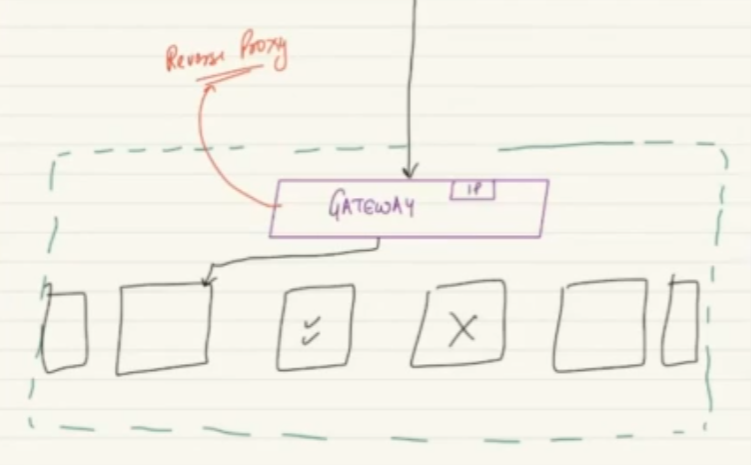
This gateway machine is known as Reverse proxy. I don’t expose the cluster of machines to outside world.



Gateway is first POC for outside world to my internal system. Why it is important: I don’t want my end machine to be exposed to outside world. I want them to be flexible, I can add or remove more machine. I don’t want to internal machine to world.

I don’t want world to talk to my machine directly. Don’t want to expose. I want them to be very flexible. Right now I have 4 mcine. In cluster. Tomroow I can add more or remove. So don’t want to expose internal machine to woutside world.

Also I want to make a secure ecosystem , don’t want outside to talk to my mahcines directly. If the internal box in intranet. I don’t give a public IP to them.. every communication happens through the gateway as it has static public IP address. It will only machine will take request from outside world.

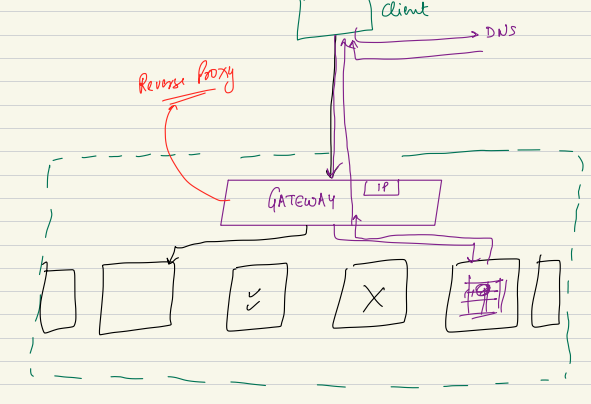


Other machine work on private internal only IP.. in DNS mapping I will wirte the Gateway machine..

Control Flow:

Client (browser, mobile app,) goes to DNS gets mapping , gets IP and send request to IP. That IP take to GWY machine.. gateway will not do any computation. Only task is to ensure the request coming to me is not a bad actor. Are yu coming froma blocked IP, bad actor. Yu sending me so many request that you are a bot. gwy does sanity checks… other than that the good request comes GWY acts aas postman and send to right machine. One of these machine to whom gwy forwad to… does computation, calculation, data fetching, does everything. It replies with an answer to GWY and GWY forward it to client. This is not one req at a time..

We will have multiple request… this is job of gwy machine. Acting as a postman.



If no of request is more than capacity of gwy it can handle???

We had removed the ASPOF from cluster but introduced in GWY… there are ways to handle it. GWY is dumb machine will not do much computation. I will have passive gateways which will take its position. It follows active passive gatewaying.. this is not master- slave.

Passive is like substitute..

IP address is not hardware specific. If replug to new machine. IP stays same.. the static ip can stay same, machine can change.. my lease line IP which machine is connected acts as Gateway. IP stays same. I can hve diff machine run on same IP.

Another way to handle gateway:

Sharded gatewaying, replicated gatewaying. gateway as a service. Multiple gwy machine each acts as gateway..

I call it gwy.. act as gwy of backend system..

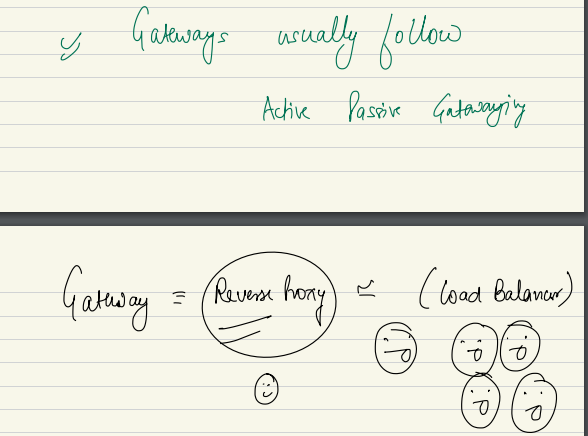
Some people call it reverse proxy, some people call it Load Balancer. Its slightly worng in reality. In relity a machine can act as gwy and LB. LB means I am gonan distribute load across many macine. Here gwy need not always do Load balancing.

The FPOC is gateway, will not call it LB. GWY is FPOC. The same machine can also distribute load. Can act as LB, but need not always do that. If people say it LB we wont argue.. gwy need not always be LB.

gateway maybe simply configured to forward the request without having to do any computation.

**Reverse Proxy: comes later.**

When you have gwy, that machine can also (usually) can perform as Load balancer.



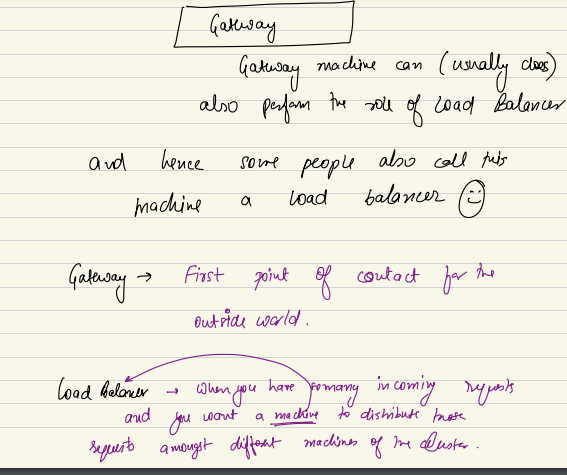
GWY: first POC for outside contact.

LB: when you have so many incoming request, and you want a machine to distribute those requests, amongst different machines of the cluster in that case this machine will cal LB.

Given both task can be done by same thing, people can call them 2 names interchangeably.

**But it can be 2 diff machine.**

LB can be done by hardware level or can be in software level to do load balancing.



When I have cluster of machine and gwy/LB at front…

We can do 2 kind of load balancing:

1. Stateless oad balancing
2. Statefull load balancing.

***STATELESS LOAD Balancing:***

There can be example when each machine are equality equipped to handle the incoming request, that’s stateless load balancing…

Example: a calculator app, code can run in all 4 amchines. I can do 5+2 in any machine. All run the same code. Any machine can do the calculation done.