* Web host – Someone hosting our data. It could be the same server which hosts other customer’s data as well.
* Virtual private server (having your own vm hosted by company)
  + This is like vm which is virtual slices in a single physical machine in which we still not secured as it can be accessed by web hosting company
* Scaling
  + Vertical – Adding more resources to the physical machine by adding more RAM or cores which has a limitation
  + Horizontal – Having more servers behind load balancer. DNS can be configured to return the ip address of each server in a round robin fashion so that the requests are sent to those servers. But there are chances some servers will be overloaded. Also cach’ing the dns name to ip at the OS level/browser level will end up sending requests to same server. To overcome this we can use load balancer take care of round robin distribution
  + RAID- hard drives RAID0,1,5,6,10
* Hard Drives
  + PATA – Parallel ATA
  + SATA – 7200 rpm
  + SAS (faster) – 15000 rpm
  + SSD (solid sstate drive) – no moving parts

1. What Type is the problem?

1. System Architecture or Object Oriented Model or Particular Algorithmic solution?

2. IAP - What is the input? What is the process? What is the output?

2. Use Cases (Ask lot of questions)

1. Shortening, Redirection, Custom, Automatic Expiry, Analytics,

2. UI and/API? Question to ask

3. Idempotency and Immutability

4. Finalize the Scope (Very important)

3. Constraints (Data storage, Traffic) (Ask lot of questions)

1. Maths

1. Bigdata statistics:

1. Facebook

2. Twitter

3. LinkedIn

4. Amazon

5. Dropbox

6. Google

2. Traffic:

1. Number of Requests per month

2. Number of Requests per sec for each type (possibly connect to use case/services)

3. Data storage

1. Number of bytes required for each types (possibly connect to use case/services)

2. Storage for 5 yrs (possibly connect to use case/services)

2. Confirm if the assumption is right

4. Abstract Design

1. List out an architecture

1. All components

1. UI

1. Small design about UI

2. Network layer

3. Concurrency (possible iOS components)

2. Services

1. Application Layer

2. Data storage Layer

3. Brief about some implementation or basic algorithm/logic

1. Hashing, Datastructure, Algorithm to use etc.

2. Confirm with Interviewer if I am in the right direction and he is happy (VERY IMPORTANT)

5. Bottlenecks

1. Check against the Maths (Constraints)

2. Traffic bottle necks

1. See if we should implement load balancer.

3. Data bottle necks

1. See if we should implement caching in-memory etc.

4. Availability bottle necks

1. Vertical Scaling or Horizontal Scaling (more in scaling)

5. CAP Theorem

6. Scalability

1. Knowledge - Tradeoff - talk!

1. Load balancer

2. DB scaling

3. Vertical vs Horizontal scaling

4. Datacenter to UI

5. DB sharding

6. Caching