**Duplication of Services and things that come along with it (Load Balancing)**

1. **Network Load Balancer**: It has one or more NICs mapped to one or more VIPs. A VIP is associated with a pool of servers downstream. Basic advantages: a) Clients don’t need to know the IP addresses of the individual servers. b)LB can take the faulty servers out of the pool. LB also supports service discovery and health checks. Service discovery makes sense in the context of API gateway or L7 LB. Both are reverse proxies.
   1. **How to load balance**: A) Basic round robin. B) Periodically monitor the health by hitting a service endpoint for CPU Usage for example. Continuous monitoring the health is expensive. Delayed monitoring, keep the last sampled info in cache. But this creates herd effect. If the server just joins, its load is zero. All the subsequent requests will go to that server only till the next load information is sampled. And at that time that server could be marked as busy and taken out of rotation. At the next sampling, requests could again switch to that server. Ping pong effect. Possible solution: Pick randomly any two healthy servers and send the request to one that reported less load last time.
   2. **Service Discovery:** Discover list of available servers. Static – Configuration files, Dynamic – DNS or using data store (zookeeper). Advantage – Servers can be added or removed anytime; autoscaling.
   3. **Health Check:** Do you need to take a server out of the pool. Passive – Forward requests to the server, if the request times out (503), take it out. Active – Monitor a health endpoint exposed by the server.
   4. **Load Balancer’s Implementation:**
      1. **DNS Load Balancing:** If you have a few servers for a particular service and they have a publicly reachable IP, add those IP to the A record for service. Client can pick one while resolving the DNS address. Disadvantage – What if a server becomes unavailable. You have to manually take the problematic IP out of the DNS record.

Diagram

Description automatically generated

* + 1. **Transport Layer (L4) Load Balancing:** Client creates a TCP connection with the VIP of load balancer, which then picks a server from the pool, performs NAT ( changes the source address of the packet to the VIP, and the destination address as that of server) and forwards it to the server. For the return path, it translates the source address to that of VIP and and destination address as that of client. Netfilter’s connection tracking mechanism is used to translate the reverse path. A conntrack entry is composed of IP addresses, ports, sequence numbers etc so a response can be mapped to the right conntrack entry and the load balancer can forward the request to the right client.

Usually consistent hashing is used to assign a connection to a server to minimize disruptions as it’s beneficial from the server’s cache perspective to have requests from certain clients hit the same server. If consistent hashing is not used, the entire cache built on the server can be rendered useless if a new server joins the pool or an existing server leaves.

Advantages of L4 – Fast, Passive Health checkup.

Disadvantages – Just a reshuffling of bytes back and forth, cannot do TLS termination or features that require higher level network protocols

Diagram

Description automatically generated

* + 1. **Application Layer (L7) Load Balancing:** Reverse Proxy. Inspects a packet and sends it to the backend server. Two TCP connections here. It can multiplex multiple HTTP requests to the same TCP connection (HTTP2 – multiple concurrent streams multiplexed over same TCP connection). TLS Termination, Rate limiting and routing HTTP requests belonging to the same logical session to the same server (using session ID and consistent hashing). L7 Load Balancer is typically used as the backend of L4 LB. L4 simply forwards it to one of the L7 LB which can then intelligently forward it to the right server based on the endpoint of the HTTP request. L4 LB cannot make endpoint based forwarding decisions so a L7 LB can cater to multiple services while L4 can’t. For example, server1, server2 serving endpoints for service A and server3, server4 serving endpoints for service B can all register with the L7 LB for a particular application.

**A picture containing diagram

Description automatically generated**

* + 1. **Geo Load Balancing:** Minimising latency between client and server. Minimising error rate which may occur if packets have to travel long distances. Make clients communicate with geographically closest data center’s L4 based on DNS. Based on client’s IP, DNS server returns the list of closest L4 LB VIPs.

Diagram

Description automatically generated

1. **Replication:** Replication of services or data. Static data or stateless services – just copy the data or service binary to multiple servers and add a load balancer in front of it and you are done. Challenge is with dynamic and changing data. Methods of Replication:
   1. **Single Leader Replication:** Writes sent to the leader which updates its local state and replicates the changes to followers.
      1. **Replication Methods**:
         1. **Asynchronous Replication:** Leader receives write request from the client, leader responds with success immediately while sending replication requests to the followers. Cons – Non fault tolerant. If the leader dies before sending the replication requests to the followers, data loss may occur. Non consistent.Only eventually consistent. Advantages – Highly available. Primary node can continue it’s work even if the secondary nodes are down – i.e. writes are accepted.
         2. **Synchronous Replication:** The leader waits for the write to be replicated to all the followers before returning a successful write to the client. Cons – Poor performance, slow. If a replica is down, writes become unavailable. A better strategy is to have a combination of aync and sync replication, raft replicates the writes to a majority before returning the response to the client.
      2. **Replication Semantics:**
         1. **Statement based replication:** Primary node saves the statements it executes and sends to the followers. Problems: any nondeterministic function (such as NOW()) might result in distinct writes on the follower and leader.
         2. **WAL shipping:** Problems: WAL defines data at a low level, and so is tightly coupled with the storage engine. Primary and secondary need to be running the same version. Used in Postgre and Oracle.
         3. **Logical (Row based) Replication:** Replication log is decoupled from storage engine iternals. Sequence of records describing writes to tables at the granularity of the row. Used in MySql.
   2. **Multi Leader Replication:** Needed for high throughput of writes which cannot be handled by a single leader or providing faster writes or when a leader needs to be available in the geographically closest data center. Replication happens asynchronously. Problems – conflicting writes. Conflict Resolution Strategies – Most recent write wins,
   3. **Leaderless Replication:** Any replica could accept writes. No leader. For making sure that every read will reflect the latest write, R + W > N needs to be satisfied. For a write request, atleast W replicas need to acknowledge before returning the response to client. Similarly for read. If high throughput of read is required, R needs to be small. If R is too small, W needs to be very large which makes it slower and less available in the worst case.
2. **Caching:** Reduces the load on the data center. In memory, so faster than the external data store.
   1. **Policies:** Cache Eviction – LRU. Expiration – TTL. Expiration doesn’t need to occur immediately. It can occur at the time of lookup. For an inline cache where user only interacts with the cache, if the data store is available, it’s better to return the expired record.
   2. **Scaling:**
      1. To scale the cache, a partitioning technique like consistent hashing could be used.
      2. If the cache is down, the external data store could get a surge in the requests causing it to become unavailable. Load Shedding should be used to prepare the data store in such a case.