Bangalore, India (Head Office) - #27, Ambalipura, Bellandur Road, Bangalore 560103, India.
Phone : +91 80 4131 7700

California, USA | Texas, USA | New York, USA | Doha, Qatar
www.instacarma.com

InstaCarma's high availability solution helps a gaming company handle its growth

The case study discussed here explains how we conceived and implemented a high availability clustering solution for a gaming company* whose sites receive massive traffic and extensive hits. The HA solution aims at providing high availability, redundancy, reliability of services and smooth load balancing for the websites.

HA cluster implementation using LAMP and Ultramonkey

We'll now go into the specifics of this HA cluster implementation in detail in the section below. The gaming website receives millions of hits from all over the world. The HA cluster implementation is aimed at balancing the http traffic over the clustered web-servers as well as providing high availability.

Load Balancing:

- The gaming website is hosted on a LAMP clustered setup, and the heavy HTTP traffic to the site is distributed over a six-node Apache cluster with the aid of a load balancer setup in front of the Apache clusters. To ensure that the load balancer does not become a "Single Point of Failure', the load balancer too is made up of two nodes forming an Active/Active configuration (LB-1 and LB-2), that monitored each other using a heartbeat, therefore ensuring that if one load balancer fails, the other would take over silently. Both these load balancer nodes have a public IP address and homogeneous configuration.
- DNS Round Robin is used to route the website to the public IP addresses of the two load balancers. The DNS is configured in such a way that the domain name has multiple A records pointing to the public IP addresses of the load balancers, and the RR data returned in a DNS query is rotated in cyclic manner to balance requests between servers. In this manner, the DNS requests are equally distributed to the two load balancers – LB-1 and LB-2.
- The load balancers are configured in such a way that it would send requests to the Apache node with less load, as well as take care of connections and sessions to ensure smooth delivery of the website. The HA and load balancing solution is actually achieved by Ultramonkey installed on the load balancers. We will now come to how Ultramonkey(UM) handles this effectively:

1. HTTP requests for the website reaches either of the load balancers based on the DNS round robin mechanism. Ultramonkey installed on the load balancer receives this request and sends it to one of the real http servers thereby making the clustered web server appear as a single web server to end-users.

2. UM does this load balancing seamlessly by making use of the Linux Virtual Server or LVS. LVS enables TCP/IP and UDP connections to be load balanced with the help of Layer 4 Switching mechanism (ie. with the help of IP address and port information) to make load balancing decisions. The host that LVS runs on is referred to as the Linux-Director which is our actual load balancer.
3. On the Linux Director, we have virtual services running, in our case a virtual HTTP service defined by an IP Address, port and protocol. The IP here is the global IP of the load balancer and port/protocol is specified based on the service we need (Port 80/tcp). By doing this, the linux director acts as an interface that receives the http requests from end users and passing it to the real servers.
4. When an end-user sends a packet to the above virtual service, linux-director with the help of a scheduling algorithm decides which real-http-server to send the packet to and forwards the packet. When the server replies, linux-director forwards this reply back to the end-user.
5. Since we have two load balancers to ensure high availability, UM uses heartbeat installed on linux-directors to ensure availability of both load balancers, and it does this by sending messages to the linux directors at regular intervals. If a response message is not received from a particular load balancer, then the machine is assumed to have failed and heartbeat on the active load balancer takes up the IP address of the failed load balancer. In such a scenario, requests to both public Ips setup using DNS round robin will be handled by the active load balancer node.
6. A service called Ldirectord running on the linux directors handles the monitoring of the real servers and their insertion and removal from the pool of servers available. It monitors the health of the real-servers by periodically making a request and checking for an expected response. For HTTP servers, this means requesting a known URL and checking that the response contains an expected string. This is called a “negotiate” check. If a real-server fails, then that server is made quiescent and is reinserted once it comes back on line.
7. Ganglia Cluster monitoring software is installed on one of the load balancers to monitor the availability of the cluster nodes and for maintaining a history of the node-availability statistics. And the ganglia monitoring daemon (gmond) is installed on all the six clustered webserver nodes. This daemon uses a simple listen/announce protocol via XDR to collect monitoring state and then shares this information via XML over TCP.

Data Management

We now come to how the data management is handled in our cluster configuration.
All the six real servers have a LAMP installation, and the web data is mainly replicated on all the six clustered servers via rsync to reduce I/O throughput, while variable data on the website is mounted via

NFS from a single machine. One of the six nodes are tagged as a master node for making changes to the website and all the other nodes rsync to this master node. This storage approach has the benefit of never requiring any software mirroring of data between cluster nodes, thus saving precious CPU, I/O and network resources. There is also a script setup for rsyncing to the other nodes, that is executed only when file changes are made to the website data on the master node. Running this script ensures that the website content is synchronized on all the nodes.

The mysql databases used by this website is maintained on a single high end dedicated dual quad core machine capable of handling high I/O, with all the clustered nodes connecting to this single machine for database access. This server is maintained solely for mysql access and it also had its mysql data folder synchronized with a backup folder on one of the clustered nodes in the event of a server failure.

The part of the website that hosts variable data such as user modifiable folders that permit write access to users is maintained only on the master node with its backup on one of the other nodes, and all the other nodes mounted this data via NFS. The NFS option did pose a performance bottleneck due to which other options for data integrity is being considered for future scalability.

** Name of the client cannot be revealed as per the NDA (Non-disclosure Agreement). References and testimonials are available.*