V2: Highly Available, Scalable WordPress Web site on Azure

# Previous Architecture

Previously, we had created a WordPress® blog site using the LAMP stack and got it up and running in minutes. All the software was deployed on one Linux machine and external connectivity was established using the Azure DNs Zone and a static public IP address component. It was and is a simplistic approach to get the WordPress® web site up and running. The architecture looked like following:



As good as the previous approach is, it suffers from some pitfalls;

* The architecture lacks the high-availability, scalability that is the hall mark of any cloud deployment. It has single point of failures and not much resiliency built in.
* The Azure Web plan – although provisioned, is not used – therefore we are unnecessarily paying for a service that we are not using.

# New Design

In this second iteration of this exercise, we are going to architect and deploy a highly scalable and robust solution which will leverage the cloud capabilities of scalability, resiliency and back-up and restore. We will also be removing the Azure Service Plan and App Services from our final deployment.

# Target Architecture

The resilient, scalable and available architecture that we are planning to build in this phase would look like following.



In the target architecture – I’ve separated the single instance architecture into front-end (Apache web server, WordPress installation) and back-end database server (MySQL database). The target architecture will increase the availability and reliability of the front-end Apache, WordPress service using the Azure Availability Sets Capability. Higher availability is also achieved by spreading the incoming request across multiple virtual machines and will utilize the Azure Load balancer in front of the network interfaces of each VMs. The deployment will be utilising subnets and network security groups (NSGs) to provide secure access. Subnets in Azure are used to control network flow and acts as a security boundary. Back-end database will accept dataflow only from the front-end sub-net.

Database layer – using MySQL – is still a single instance in our architecture and this can still be a single point of failure. Regular back-up and restore capability of Azure will be utilised to provide high availability solution – but there is still a possibility that the solution is unavailable for a certain time when the database is being restored. Given the expected load and cost involved, this risk is deemed to be justified.

Let’s start the engines and create the new target architecture. Initially, I’ve removed all the existing components except the following

1. DNS Zone
2. Public IP address

If you recall, I bought the domain, *ninadkanthi.co.uk* from GoDaddy.com® domain registrar and later changed the default NS records specified there to the four records of the Azure DNS name servers. I also added a record-set into the Azure DNS to connect the A level record ‘blogs.ninadkanthi.co.uk’ with the public IP address component. You can review both these steps [here](https://blogs.ninadkanthi.com/index.php/getting-wordpress-site-up-and-running-in-azure/) under the section ‘Static IP Address, DNS Name Label’

The components that I am starting from are:

