

**Creating a high-availability,
scalable web-server system
within the Digiweb cloud.**

Table of Contents

Abstract.....	Page 3
Introduction to Cloud computing	Page 3
Business benefits of Cloud computing	Page 4
A highly scalable environment	Page 5
Implementing shared storage	Page 6
Installing & configuring the database server.....	Page 7
Installing & configuring the web server	Page 7
Load balancing the environment	Page 9
Planning for scalability	Page 9
Digiweb & Cloud computing	Page 11
Glossary	Page 12

Abstract

Cloud computing is all the rage. "It's become the phrase du jour," says Gartner senior analyst Ben Pring, echoing many of his peers. The problem is that (as with Web 2.0) everyone seems to have a different definition. The key thing to note however is that when used effectively Cloud computing can provide an entirely new way of using IT resources in the most efficient and practical way possible. Cloud computing really comes into focus when you think about what IT always needs: a way to increase capacity or add capabilities as and when required without investing in new infrastructure, training new staff, or licensing new software. Cloud computing encompasses any subscription-based or pay-per-use service that, in real time over the Internet, extends IT's existing capabilities. Cloud computing, therefore, brings a more cost-effective, faster and less risky alternative to traditional on-premises development, changing forever the economics of business computing.

In this white paper we will look at the practical configuration of a highly available and scalable Wordpress-based web application. A web application generally has a number of discrete tasks and activities. Most common amongst these are storage (of code, databases & static content), serving of databases, generation and serving dynamic content and serving static content. Whilst all of these activities can be carried out on the same server, specialising these functions onto several dedicated machines can provide far greater scalability and in many cases, a highly available environment.

Please note the technical glossary at the end of this document is used to define many common terms for the avoidance of ambiguity.

Introduction to cloud computing

According to leading research company Gartner, 'Cloud computing heralds an evolution of business that is no less influential than e-business.' Gartner maintains that the very confusion and contradiction that surrounds the term "cloud computing" signifies its potential to change the status quo in the IT market. As its name suggests, this kind of platform lets developers write applications that run in the cloud, or use services provided from the cloud, or both.

Put simply, Cloud computing is a computing paradigm in which tasks are assigned to a combination of connections, software and services accessed over a network. This network of servers and connections is collectively known as 'the cloud.' Users can access resources as they need them. For this reason, cloud computing has also been described as on-demand computing. Cloud computing, therefore is a highly scalable and flexible service that can be delivered and consumed over the internet through an as-needed, pay-per-use business model.

Traditionally, if you wanted to run IT applications you needed to buy servers and software, place them in your office and hire IT staff to manage them. Cloud applications are a new way of using applications with none of the overheads of traditional applications.

Business Benefits of Cloud Computing

Cloud computing offers many business benefits; the two major benefits being cost and flexibility. A Cloud-based Infrastructure can be up and running within 10 minutes, which is unheard of with traditional physical servers. They cost less, because you don't need to need to pay the people, products, and facilities to run them. Additionally they are more scalable, more secure, and more reliable than most applications. Plus, upgrades are taken care of for you, so your applications get security and performance enhancements and new features—automatically.

Cloud computing therefore is an attractive option for business offering:

- Immediate access to computing hardware, with no up-front fee
- Lower IT costs by only investing in the applications and services needed
- Develop, deploy and run applications that can easily grow capacity
- Flexibility in working

A Highly Scalable Environment

A web application generally has a number of discrete tasks and activities. Most common among these are the following activities:

- Storage of (code, databases & static content)
- Serving of databases
- Generation and serving dynamic content
- Serving static content

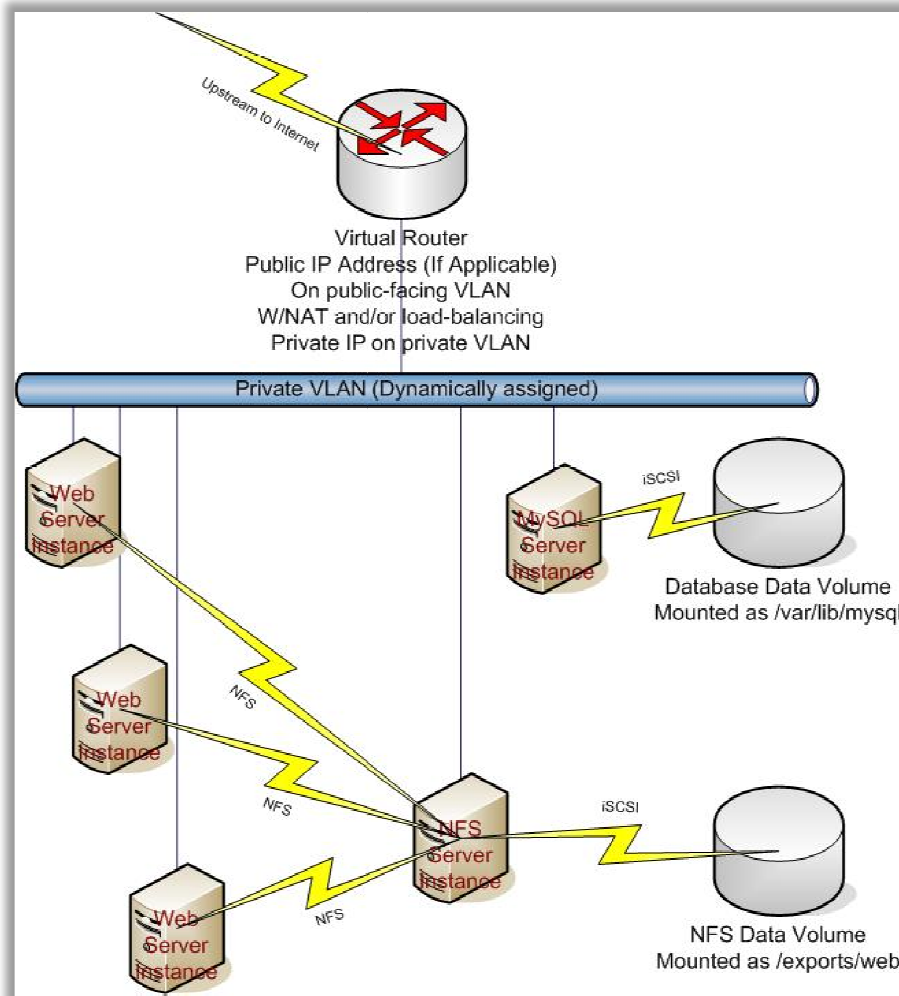
While all of these activities can be carried out on the same server, specialising these functions onto several dedicated machines can provide far greater scalability and in many cases, a high availability environment.

For the purpose of this paper we will focus on the generation of dynamic content and serving of static content, done on the same layer. This is generally going to be the case with a LAMP stack¹, but in other cases, using a highly efficient web server such as thttpd may be required.

Database and storage each have their own layer. Our objective here is to scale the servicing of end-user in addition to being able to have the minimal resources available for off-peak times.

¹ If the generated content is coming from something like Jboss or Tomcat, splitting those up would again allow for more scalability for those functions, and isolation of scalability to individual functions.

A Sample Highly Scalable Environment



'Sample environment with multiple load-balanced webservers, with discrete database and file servers'

Implementing Shared Storage

To implement storage, NFS was chosen. Due to the built-in High Availability (HA) features of our cloud environment, it was not necessary to create an HA-enabled NFS cluster with DRBD, but this can be added as an additional level of availability if required.

Initially, a suitably sized server instance was instantiated, with a separate large storage volume. One of the pre-existing CentOS Linux templates was used. With this cloud environment, new templates can be created by booting a server from a supplied ISO file, configured and then saved.

Once the server was built and running, NFS services were installed. Following this the SELINUX settings were adjusted and local (iptables) firewalls amended accordingly to serve the content.

LVM2 was used to create a storage group called “exports” from the storage volume:

- `pvcreate /dev/xvdb`
- `vgcreate exports /dev/xvdb`

In this storage group, 50% of the space was allocated to a logical volume called “web”²

- `lvcreate --name web -L 10GB exports`

and an ext3 file system was created on that volume.

- `mkfs -t ext3 /dev/storage/web`

An appropriate entry was created in /etc/fstab, creating the mount point. The new partition was then mounted.

- `echo '/dev/exports/web /exports/web ext3 defaults 0 0' >> /etc/fstab`
- `mkdir -p /exports/web`
- `mount /exports/web`

An entry was placed in the /etc/exports file which allows all servers within the local subnet to mount the shared web file space. The NFS service was then started and the directories exported:

- `echo '/exports/web 172.23.1.0/24(rw,no_root_squash,sync)' >> /etc/exports && service nfs start && exportfs -r`

The NFS service is now ready to begin serving files to any local web servers. Please note the IP address assigned to this VM for future use when mounting the NFS shares. The private IP can be seen through the control panel, or by using interface configuration from a shell on the instance.

² Using LVM and not allocating all space allows for scaling up of the in-use storage areas, and creating new ones later on if they become necessary or desirable.

Installing & configuring the Database server

For the database server, the same initial template as the NFS server was used, with a smaller data volume size. Similarly, the built-in HA feature was used but a MySQL replicating pair could also be used.

Once the initial instance was running, a volume group called “storage” was created following the same procedure used for the “exports” volume group on the NFS server. A similarly-sized logical volume called “mysqldata” was also created.

- `pvccreate /dev/xvdb`
- `vgcreate storage /dev/xvdb`
- `lvcreate -name mysqldata -L 5GB storage`
- `mkfs -t ext3 /dev/storage/mysqldata`

Once this step was done, the mount point is created. Knowing that the default install of mysql for this Linux distribution is `/var/lib/mysql`, a directory is created, `fstab` updated, and mounted.

- `mkdir -p /var/lib/mysql`
- `echo '/dev/storage/mysqldata /var/lib/mysql ext3 defaults 0 0' >> /etc/fstab`
- `mount /var/lib/mysql`

At this point, `mysql-server` was installed.

- `yum install mysql-server`

The steps above in creating the `/var/lib/mysql` mount caused the data that contains the actual mysql databases to be put on the mounted file system (which is resizable on the fly) rather than on the root file system.

Installing & configuring the Web server

The web server is now installed and configured. The shared NFS file store does not just contain the web code and static content - it also contains the Apache server configuration files. Similarly, the shared storage can contain configuration for items such as PHP if necessary, but in this example, it is not implemented.

The web-server instances do not need a separate storage disk, as almost everything they do comes from either the shared file system, or the database server. Therefore, the shared file system must be mounted, with the webroot and Apache configuration files symbolically linked to the correct directories.

Step 1: Create a new instance, again using the same CentOS template:

- `Install apache and php (if making a LAMP setup.)`
- `yum install apache php`

Once Apache is installed, it must be immediately shut it down.

- `service httpd stop`

Now that Apache is no longer running, the shared storage must be mounted

- `mkdir /mnt/web`
- `echo '172.23.1.x:/exports/web /mnt/web nfs tcp,wsiz=1024,rsiz=4096,rw 0 0' >> /etc/fstab`
- `mount /mnt/web`

This makes the shared filesystem available, and cause it to re-mount when the server restarted. Check with

- `df -h`

Step 2: create the directories for the webroot and the web configuration:

- `cd /mnt`
- `mkdir webroot`
- `mkdir webconf`

Step 3: put the webroot onto the shared storage:

- `cd /var/www`
- `tar -c ./ * | tar -xv -C /mnt/web/webroot/`

Step 4: check to make sure that the files copied:

- `ls -lhr /mnt/web/webroot/`
- `ls -lhr /var/www/`

You should see nearly identical directory listings (the `/mnt/web/webroot/` listing should contain a "lost+found" directory that should not be in the `/var/www/` listing).

Step 5: remove the old webroot, and symlink the new one into place:

- `cd /var`
- `rm -fr ./www`
- `ln -sf /mnt/www/webroot ./www`

Step 6: repeat the above process for the apache configuration directory:

- `cd /etc/apache2`
- `tar -c ./ * | tar -xv -C /mnt/web/webconf/`
- `cd /etc`
- `rm -fr ./apache2`
- `ln -sf /mnt/web/webconf ./apache2`

In Centos by default SELINUX does not allow Apache to serve content off of symbolic links. Therefore the SELINUX policies need to be adjusted. Check your distribution for more details.

Now that both the webroot and the Apache configuration directories exist on the shared storage (with symbolic links, so that they can be “seen” in their default locations.) Apache can be restarted.

```
- service httpd start
```

Load Balancing the Environment

There is now a running web-server, but it is not visible to the outside world. To enable this function follow the steps below:

Go to the “Network” section of the cloud control panel. Obtain a public IP address, and create a load balancing policy that forwards port 80 to port 80, and add the web-server to the load balancing pool. Most applications really need to have IP-based load balancing turned on, to ensure that a user hits the same back-end server - this helps maintain session information.

You should now be able to access content being served by Apache via the public IP address you associated with the load-balancing policy.

Now you can install whatever applications you want to install under Apache. In this example, Wordpress was subsequently installed.

Once the application is configured and tested to your satisfaction, the web server can be stopped and a template from the root disk created. Be sure that your testing includes some reboot tests where you shut down, then restart the instance and ensure that all services necessary to serve content come back up automatically. It is recommended that the NFS and MYSQL instances are tested in a similar way.

Now that a template has been created from the web-server, you can start the web-server back up. Once this is fully started you should once again be able to see web content via the public IP associated with the load-balancing policy.

Planning for Scalability

As traffic grows, users may find that the site has become slow, or see that the load is too high on the web-server instance.

As the web server has been “templated”, and the system allocates IP's and such dynamically as you create new instances, you can simply create a new instance (or however many are necessary) to serve your traffic from that template. Each new instance will get a new IP address in the private network.

Once those instances are running, you can add those instances to the load balancing pool, and they will all serve content to end users, thus sharing the load for serving the content between all web servers. This also increases overall capacity for the site.

This allows a system to scale up and down in response to load. For example, during peak times of 12-3pm, a site may need five web server instances while after 11pm in the evening, it may only need one. With this cloud platform, it is possible to run servers only when needed – and pay only for the hours used.

NB: When you light up your 2nd, 3rd or 10th web-server instance, you didn't have to do ANY configuration for them to start serving content, other than to add their newly-assigned IP's to the load balancing pool.

Being able to template things, and create & turn them on when you need them, is part of the “magic” that cloud computing can provide.

NB: After initially creating the first instance in the system, and setting it's root password to something other than the default, I created and applied a port-forwarding service, and used SSH to get to the server instead of using the remote-console through the system. Access is much faster, and things like cut & paste work the same as you would be used to.. and you can SSH from this instance to the others in the system, so that you only need the one port-forwarding service to be able to administer all the servers you create within this system.

Digiweb & Cloud Computing

Within today's economic environment, companies must use IT resources in a manner that provides optimum return on investment (ROI). Digiweb's Cloud computing solutions offer business' an entirely new way of using IT resources in the most efficient and practical way possible. Our service delivery model, uses various available cloud computing models, to deliver your IT services in the most cost-effective way that also meets your specific needs for security, performance, regulatory compliance, and strategic objectives.

We feel that a true cloud computing platform must have the following elements:

- Hardware management is abstracted from the user.
- No upfront costs, all charges are for usage only.
- Service must be scalable up and down, without contractual obligation – use only what you need.
- Must be instantly self-provisioning, via web interface or API calls.

Utility Cloud computing

The future of computing is cloud – power your IT needs with Digiweb's utility computing cloud. This approach gives company's immediate access to computing hardware, with no up-front fee whilst also allowing companies to only pay for the solutions they need, therefore significantly lowering IT costs. This solution also allows organisations to easily scale their IT in line with company growth with no penalty

How is this achieved?

- Pick your server specification (RAM, CPU)
- Pick the size of high-performance SAN storage
- Pick the operating system and applications
- Provision – and your server is ready instantly

Business & IT benefits

- Provision new servers, destroy old servers in seconds
- Scale CPU, RAM up or down in seconds
- Schedule snapshots or restore in seconds
- All servers enjoy high availability, failing over instantly and automatically to other servers in the event of server failure
- With built in load balancing, use cloud servers to scale our horizontally automatically

Private Cloud Computing

If you need the power of a hosted cloud environment, but do not wish to use our shared utility cloud environment, you may want a wholly private cloud. With a Digiweb's private cloud, you gain all the benefits offered by Infrastructure-as-a-Service cloud computing without working in a multi-client environment. Digiweb manage the entire physical environment, servers and storage – all the client needs to do is deploy their virtual server assets as needed. These environments are also 100% customisable so if a client needs a particular environment deployed we can provide it.

With the Digiweb Cloud, our two priorities are security and availability. For cloud computing to be truly usable, there cannot be any doubt about the confidentiality, availability and integrity of the service. Our data centre facilities are secured and regularly audited. All data centre's are protected with uniformed security and manned 24/7, with round-the-clock monitoring and patrolling by more security agents.

Technical Glossary

System: A group of instances working together to perform an overall function. (In this context systems are used to serve web content to end-users.)

Instance: An instance is for all intents and purposes, a server. In this case, it is a virtual server with fixed resource allocations.

Volume: This is a virtualised storage repository that can be created separately and associated with a particular instance. Each instance has at least one “root” volume associated with it to contain the operating system. Generally an instance will ALSO have a storage volume associated with it, although you can attach and detach volumes at will to a running instance. (although.. depending on how you have your VM organised for an instance.. detaching a storage volume might cause problems.) The storage volumes used in the cloud are served via iSCSI, but this is entirely abstracted from the end-users, and the virtual machine instances, so that you will see the iSCSI-based storage as if it were local drives on the server instance.

Public IP: A public IP is an IP address that can route traffic to and from the public Internet. A public IP is not necessary for most functions within the cloud. Generally for a given system of instances, single public IP address is sufficient, as inter-server traffic happens in a private network that ties the instances together.

Private IP: An IP address that resides within the private network that ties the instances of a system together.

Port Forwarding Service: A port-forwarding service allows you to associate specific ports on a public IP in a 1:1 relationship with specific ports on an instance's private IP. Generally only management of servers should need be done this way, as services for end-users are best served via load-balancer rules. (see below.) Several port associations can be added to a port-forwarding service. When this multi-port service is associated to a particular instance, all the ports are forwarded to the same one. The port associated on the public IP does not have to match the port used on the private IP, which can be used to add non-standard ports such as a non-privileged port mapping through to port 22 for SSH or a random port mapping to port 3389 for RDP if this is desired, without having to modify the service configuration to change the listening port.

Load Balancer Rule: A load-balancer rule lets you establish a 1:n relationship of inbound ports on a public IP address with ports on the private IP addresses of content servers, such as web-servers. Load-balancing can be done either “round robin” where it cycles requests through each server, or “IP based” (often known as “sticky”) where an individual user should see the same content server each time they access a site/service. Again, the port on the public IP address does not have to match the port used on the content server, which can be useful for serving content via port 80 from tomcat servers and other applications which default to using port 8080, for example. (This keeps end-users from having to add :8080 to the URL's they are typing into their browsers.)

HA: In this instance, HA (High Availability) refers to the cloud system's ability to tell if a node goes down, and instantly start new copies of any instances running on that node on another node. This minimises downtime.. as if the NFS or MYSQL server in this example fail, the system will turn on another copy of the instance which will start serving content once it is back up.

Template: A template is a pre-saved image of a server. Typically contains operating system and applications.

For more information, contact Digiweb

College Business & Technology Park,
Blanchardstown, Dublin 15, Ireland

Tel: +353 (0)1 256 9200

Fax: +353 (0)1 8242586

Web: www.digiweb.ie

Email: solutions@digiweb.ie