Load Balancing/Clustering

Case Study: DrukNet - Bhutan

An ISP in Bhutan is looking towards the future and a possible mail cluster to better support expected increase in demand. The following is a proposal discussing how they might go about this.

In addition to this proposal here are some comments:

A proxy-based cluster may be the best way to go. This can be cheaper albeit harder to manage. Cambridge University has well-documented how they built their mail clustering solution:

http://www.cus.cam.ac.uk/~fanf2/hermes/doc/talks/2004-02-ukuug/
http://www.cus.cam.ac.uk/~fanf2/hermes/doc/talks/2005-02-eximconf/

As of June 2005 the Cambridge's patches to Cyrus for replication have not yet made it into a public release.

Concerning back-end storage, using NFS with something like a Netapp is the prefered route. As of this writing Netapp has a lower-end product, the FAS270C, which is a single chassis containing a bunch of disks and two Netapp head-ends in a cluster configuration. More details here:

http://www.netapp.com/products/filer/fas200_ds.html

DRUKNET MAIL SERVER CLUSTER PROPOSAL
----

1. Introduction

This brief outlines a design for a scalable E-mail cluster which should serve Druknet's current needs (around 3,000 mailboxes) whilst allowing for substantial growth (100,000+ mailboxes). The design uses free, open-source software, aiming to allow maintenance and growth to be managed locally by Druknet staff, minimising reliance on external support and maintenance contracts. The architecture provides for high service availability, performance, and security. It has been proven at large ISPs in other countries.

2. Terminology

2a. SAN versus NFS

The most important aspect of a mail server cluster is the storage array which holds the customers' mailboxes, and at this point it is worth distinguishing between two different technologies which might be applied.

* a SAN (Storage Area Network) is a disk drive array, which may be partitioned up so that portions of it can be assigned to separate servers. The most common attachment methods are FibreChannel and iSCSI (SCSI over IP). In both cases, the storage portion appears like a locally-attached drive to the server to which it has been assigned, and the server creates a filesystem on it just as it would with a local drive.

The advantage of a SAN is that it centralises resources, possibly saving space over having a separate RAID array on each server, adding the flexibility to assign more or less storage to each server as assigned, and allowing for fast centralised data backup.

However, each storage partition can be accessed by only ONE server at a time. This is a fundamental limitation for a mailserver cluster, where for both resilience and scalability reasons you would like multiple machines accessing the mailboxes for delivering incoming mail, and for users to collect their stored mail via POP3, IMAP or Webmail.

* an NFS (Network File System) server also is a disk drive array, and can divide it into partitions. The difference is that the filesystem(s) are created on the disk drives by the server itself. The individual clients access files using the NFS protocol over IP, and this allows multiple clients to have access to the *same* files and directories.

The main advantages of an NFS server for a mail cluster are:

- multiple machines can have access to the same mailboxes at once. Load can therefore be distributed between multiple front-end servers, and should any one fail or be taken out of service for maintenance, the workload can be redistributed between the remaining front-end servers without any noticeable effect to the end users.

- commercial NFS servers can have their own custom filesystems and hardware to provide extremely high performance and data safety when reading and writing files. For example, the Network Appliance devices (www.netapp.com) implement their WAFL filesystem (Write Anywhere File Layout) which optimises writes across the whole RAID array, and has on-board non-volatile RAM so that write caching can take place without compromising data integrity in the event that power is lost.

- commercial NFS servers are designed to be straightforward to manage (for example, when adding extra drives or replacing failed drives)

The main disadvantages of using NFS for mailbox access are that there is historically poor support for file locking. However, by using Maildir format for storing messages (where each message is stored in a separate file), locking is eliminated. Maildir is safe for use over NFS.

System administrators should be aware that NFS is not a secure protocol, and therefore NFS servers should be kept on a separate private network which is not directly connected to the Internet.

2b. Load Balancer

A Load Balancer (or Local Director or Redirector) is a device which provides a "virtual IP address" for customers to connect to when accessing a service, and in turn directs each connection to a real server behind which provides the service. It performs periodic service testing (e.g. every 2 seconds), so that if a server fails, that server is taken out of the pool of available servers and new connections are directed to the working servers only. It also allows a server to be manually taken out of the pool, allowing existing connections to "drain away" naturally, so that the machine can be taken out of service for maintenance with zero customer impact.

Load balancers can either operate in NAT mode, so that the destination IP address of each packet is rewritten, or in "transparent" mode. In the latter case, the virtual IP address needs to exist on each of the servers as a loopback interface, and the packets are directed to the correct one by MAC address. The advantage of transparent mode is that the load balancer has to do less work to process incoming packets, and zero work on outgoing packets, reducing the work its CPU has to perform.

3. Outline design

```
                        uplinks to rest
                           of network                    ^
                         /            \                  |
                 +----------+      +----------+          |
                 |   load   |------|   load   |      public IPs
                 | balancer |------| balancer |          |
                 +----------+      +----------+          |
                  |   |   |         |   |   |            v
       Front-end +-+ +-+ +-+       +-+ +-+ +-+
          PCs    | | | | | |       | | | | | |
                 +-+ +-+ +-+       +-+ +-+ +-+
                  |   |   |         |   |   |            ^
                 +----------+      +----------+          |
              ___|  switch  |------|  switch  |      private IPs
            /    +----------+      +----------+          |
          /       |       |          |       |           v
   logging    database  +---+      +---+  database
 & management    PC     |NFS|__   _|NFS|     PC
                        +---+  \ / +---+
                          |    / \   |
                       storage    storage
                        shelf      shelf
```

* LOAD BALANCERS

Provide uplinks to the rest of the network (100M or 1Gbps), and allow the front-end PCs to be connected (100M fast ethernet). There are a pair, with a network cross-connect and a heartbeat failover cable. Both devices act as switches, and therefore the front-ends can be distributed across them. At any one time, one device provides the virtual IPs and the other is just a switch; if the heartbeat cable detects a failure, the other switch can take over and provide the virtual IP services.

Proposed hardware: Foundry ServerIron (1U form factor; available with 8, 16 or 24 ethernet ports, and with optional gigabit ethernet fibre GBIC ports)

These devices come with their own operating software and store configuration data in flash memory. They are configured via telnet or serial console, using a Cisco-like command-line interface.

Using two load-balancers means that should one fail completely, at least half of the servers will still have Internet connectivity. By distributing the services appropriately, so for example one POP3 server is connected to the first load-balancer and one to the other, the service should continue to run albeit with reduced capacity.

* FRONT END PCs

These are the machines which will provide the actual network services. These PCs need to be rack-mountable and have two ethernet interfaces. Otherwise, you have a wide choice of hardware, so can choose machines which meet your requirements for ease of purchase, maintenance, etc.

Using two SCSI disks in each front-end PC (as a mirrored pair) will assist in reliability, since a failed disk will not cause a total failure of the machine. You should therefore choose hardware where the disks can be replaced easily via the front panel. These disks may also be used for data storage for some applications (e.g. RADIUS logs on RADIUS server; mail queue on outgoing mail relay). This minimises the load on the shared storage array.

Suggested hardware: Dell Poweredge 2450 (2U) or equivalent. Choose the "sweet spot" for cost versus CPU power; that is, the CPU speed just *below* the sharp price rise for the fastest available processors. Not only will you save money, you will probably have a more reliable server as you are using components which have been more thoroughly tested in production. Depending on the time of purchase, this might be 2.8GHz Pentium 4 for example (assuming the price is substantially higher for 3GHz or 3.2GHz parts). Buy boxes with a single CPU for maximum software reliability, although having the option to plug a second CPU into the motherboard may be useful at a later date.

You should also consider adding remote power bars (to allow the machines to be power-cycled remotely) and a serial console server (allowing console access for recovery from serious problems remotely). However if the cluster is to be located in a site easily accessible, e.g. Thimpu, then this may not be necessary.

RAM of 512MB or 1GB is recommended. Any extra RAM will be used as disk cache to improve performance.

SOFTWARE FOR FRONT-END PCs:

All the software listed here is FREE OF CHARGE, available for download from the Internet with full source code and has no licence fees to use. It is widely deployed at other ISPs. Should software support be required (other than that available from the Internet for free - e.g. FAQs, mailing lists) then there are independent consultants who can assist. However, the more you invest in learning these packages yourself, the less reliant you will become on external consultancy.

- Operating system: FreeBSD 5.3 or later. This is an extremely reliable platform which has been proven in ISP environments across the world under very heavy load, and has a very good security track record. Enable SSH for secure remote management from either Windows or Unix workstations ("putty" is an example of a free SSH client for Windows).

- Application software:
  + Incoming mail (MX receiver): Exim. Very flexible, supports Maildir delivery and database lookups
  + POP3 and IMAP: Courier-Imap. High performance, good feature set, supports Maildir, supports SSL encryption of connections.
  + Webmail: Courier Sqwebmail plus Apache. High performance, allows some customisation of templates and style of display. Not the prettiest webmail interface, but perfectly functional. Alternative webmail interfaces can be provided on top of the IMAP interface (e.g. Squirrelmail), but may entail further configuration work or be more demanding of CPU resources.

You may consider adding further services into the cluster:

  + Outgoing SMTP mail relay: Exim.
  + RADIUS (dial-up authentication): OpenRADIUS or FreeRADIUS. Both can

authenticate against a MySQL database.
+ DNS cache: BIND 9 (provided in FreeBSD base system)
+ Authoritative DNS service: BIND 9 (ditto)
+ Signup / account management (e.g. user password changing): web application written in Perl, PHP, Ruby etc running under Apache
+ "Homepages" web space: Apache plus pureftpd

These services scale very easily because they do not need to use any space on the shared NFS storage array (except for homepages), and therefore do not apply any extra workload to this central resource.

The number of front-end boxes will depend on how you wish to partition your services. Clearly each service needs to be provided on at least two boxes for resilience, one connected into each load-balancer switch. Separating the services onto separate boxes makes it easier to determine which services are using more CPU than others, and therefore makes it easier to scale your service. However, while the traffic load at Druknet is small, you could combine services onto the same box to save on hardware.

A small initial deployment might look like:

```
    PC1:    Incoming MX
            RADIUS
            DNS cache

    PC2:    Outgoing SMTP
            POP3, IMAP, Webmail

    PC3:    same as PC1
    PC4:    same as PC2
```

(where PC1 and PC2 are uplinked to the first load-balancer switch, and PC3 and PC4 are on the other).

A larger deployment might have two PCs for each service, potentially meaning up to 18 machines. I do not think the size of Druknet's current customer base warrants this immediately, and if you can defer the purchase of additional machines for say two years, then you will get much better specification machines at a lower price.

* BACK-END SWITCH FABRIC

The back-end switching fabric does not need any intelligence like load-balancing or virtual IPs, or other features like layer 3 switching. You can therefore use whichever make or model of switch you are most comfortable with buying and managing and which has a sufficient number of ports. You will need a 100Mbps connection into each of the front-end servers and database servers, and preferably gigabit into the NFS servers. For example, Cisco 2xxx or 3xxx series switches will be just fine.

The backend network should run on private IP address space (e.g. 192.168.x.x) so that it is reachable only from the front-end servers. In order to manage the devices connected to the back-end network, a good solution is to connect the back-end network to a DMZ port on your firewall (such that connections can originate from your office network to the DMZ, but not vice versa). Or, as a simple alternative, just ssh into one of the front-end boxes and then ssh/telnet from there.

A concern is what happens if one of the back-end switches suffers a total failure. It may be possible to uplink both the NFS servers to both switches in such a way that at least a partial service will function in this scenario. It would however be wise to choose switches with dual power supplies, or indeed to replace the pair of switches with a single chassis-based switch designed with internal redundancy.

4. NFS STORAGE ARRAY

This will be by far the largest investment required to build this cluster, and is critical to its performance and reliability.

I recommend Network Appliance fileservers, as I have used them in exactly the cluster configuration shown above, and they have proved themselves to be extremely fast and reliable. The cost will depend very much on the configuration you choose.

In particular, you may choose to install either a single file server (in that case, make sure you also buy a spares kit so that you can replace a failed component locally should any hardware failure occur), or a pair of file servers in a resilient cluster configuration.

The cluster licence costs extra money, in addition to the cost of the two fileservers. However, the cluster configuration is extremely reliable. You can give a separate IP address to each NetApp and share the mailboxes between them (e.g. mount one as /mail1 and the other as /mail2). However, in the event that one suffers a hardware failure, the other device will take over the IP address of the failed device *and* take over its disk shelves. The service will therefore continue to run unaffected, albeit with one NetApp having to perform twice the workload it had before.

NetApps have a number of features which make them extremely suitable for use in an ISP environment, as well as performance and reliability:
- simple "appliance"-type configuration (they are fileservers and nothing else)
- the WAFL filesystem and non-volatile write cache, described above
- the ability to grow a filesystem just by adding disks and assigning them to a volume (no data transfer or shuffling is required)
- "snapshots", which allow you to freeze the data multiple times and go back to an image of the data exactly as it was at that time
- "snap mirroring", which uses the snapshot feature to replicate a filesystem very quickly to another NetApp; very useful when migrating from an older NetApp device to a newer model, for example
- the ability to back up snapshots to an attached tape drive

You will need to decide exactly how much performance, resilience and storage space you need at the outset, to determine the price of your configuration.

In the event that the NetApp solution is deemed too expensive for a small initial deployment, there are other cheaper solutions which can be considered: for example, Linux-based fileserver appliances from Convolo. This would give you a low initial capital outlay to deploy and test the above design; you could migrate to another storage solution at a later date, and then redeploy the initial fileserver elsewhere in Druknet, or for a different application (e.g. for commercial web hosting)

5. DATABASES

These are where user configuration data is stored - e.g. usernames and passwords for mailboxes and dial-up accounts.

MySQL provides a high-performance database solution in a free, open-source product (additional, paid-for support is also available from the company which provides MySQL). MySQL provides replication, so that a master database can be replicated to one or more slave databases.

A typical configuration, therefore, will have all account *updates* being directed at the master server (e.g. new account creation, password changes), whilst the read-only access required by the mail server front-end boxes can be directed across the slave(s) or the slave(s) together with the master.

This means that in the event of the master database being corrupted, existing services continue as normal; only account creation and updates cease to work. A slave server can then be manually reconfigured as a master server to allow the service to continue fully as normal.

You may wish to choose the *same* hardware here as you use for the front-end machines, to minimise the requirements for spares.

If you wish to store the MySQL data on the NFS server, then you will need to do some checking to ensure that MySQL is NFS-safe, and also measure whether this creates an unwanted extra load on the NFS server. However, even if this configuration is not recommended, you can always just use two mirrored SCSI disks in each MySQL box for data storage. I would not expect user configuration data to exceed more than a few GB.

You may wish to consider storing service data in an LDAP database (e.g. OpenLDAP), which also can be replicated between servers. However, if LDAP access is important to you, first consider that you may be losing out on the data integrity and querying features that an SQL database provides. You could consider a hybrid solution using OpenLDAP + backsql, which provides LDAP access to an SQL backend. You can then use direct SQL queries to update and query your database.

6. LOGGING AND MANAGEMENT

I highly recommend adding one or two PCs in the cluster for logging and management. These would include:

* A syslog server on the private network. All front-end machines can have their syslog daemons configured to send log messages to this machine. As well as reducing the workload on the front-end machines' hard drives, this

improves security auditing and makes log entries easier to find because they are all in one place.

Periodic log processing, e.g. analysis of POP3/IMAP/Webmail logins and RADIUS logins, can be used to determine which accounts are "active" and therefore to purge inactive accounts. It can also provide information on the number of mail messages traversing the system, the amount of mail which is backlogged on the system, and so on.

* An admin server (can be the same as the syslog server). The purposes of this machine are:
- to mount the NFS servers for uploading data, performing mailbox cleaning etc.
- to act as a secure gateway for administrative access to the private network. You can configure backend machines to permit telnet/ssh connections *only* from the admin server's IP address; this increases security, because in the event that someone breaks into one of the front-end machines, they would have gained only minimal access to the private network
- a machine for performing data imports, ad-hoc database queries etc
- if required, a private web interface for account management, not available via the Internet

* A monitoring machine connected to both front and back networks. This can be used to send periodic probes to each of the machines on the network and create alerts for failed devices or services. Suitable software to run on this machine would be "Big Brother" or "Big Sister", "Nagios", "NOCOL" etc.

This monitoring machine could also collect stats using SNMP for providing graphs of traffic on each of the switch ports (software: "MRTG" or "Cricket")

7. DATA MIGRATION

Having built a solution such as that shown above, migration of mailboxes onto the new service will need to be performed. There are several tools in the above suite which can make this easy:

- courier-imap has an account initialisation function called 'loginexec'. By putting a loginexec script into each mailbox, containing code to pull mailbox contents from the old server using POP3, each mailbox will be migrated automatically as soon as each user makes their first connection to the new mailserver. The loginexec file is deleted once an account has transferred its contents.

- courier-imap also has a proxy function, which can redirect inbound POP or IMAP connections to another mail server. You could therefore choose to point all users at the new mailserver, but redirect their access to the old server. Individual users can have their mailboxes manually moved, and their database entry updated so that they hit the new server instead of the old. This permits you to perform a 'staged' migration of a few accounts at a time. However the proxy function is newly added to courier-imap, and you should therefore test it carefully before deploying it.

- some services, such as outbound SMTP relay, are trivial to migrate. You simply point all your existing customers at the new relay service, and leave the old relay running until it has finally flushed all the messages from its queue, at which point it can be turned off.

Separate migration plans should be written for each service, but each is achievable.

8. DISCLAIMER

A service using a design very similar to the one shown above has been successfully implemented at a large ISP. However, you should be aware that building a service from components like this leaves you with responsibility for successfully integrating the parts. You may wish to set aside a consultancy budget in case you need assistance. The vendors of the individual components are unlikely to provide you with any application-layer integration help.

The design above has several components which you will need to design yourself to meet your specific needs, most importantly:
- the database table structures
- any web interfaces for signup, account management, and account self-care
- any scripts for log analysis and reporting
- configuration management and backup (i.e. having a procedure in place so that a failed machine can be rebuilt quickly, complete with its applications and configuration files)

The effort required to develop these components should be factored into your rollout plans, along with the effort required to plan and perform service migrations.

As with any software solution, you will need to monitor the mailing lists for the operating system and applications for upgrades (in particular security fixes) and roll them out when necessary.

Last modified: Tue Jun 14 23:45:40 CLT 2005