Load Balancing for TCP/IP in TPF

by David Morley

Any chain is only as strong as its weakest link, and no matter how good your TPF system, it is useless if there is no communications to connect you with your customer. Since the introduction of TCP/IP, TPF is able to beg, borrow and steal from the midrange environment to deliver the maximum network availabilty.

Load Balancing
Load balancing is about spreading your traffic across all processors in a complex (or front-ends), it offers maximum return on your hardware investment by utilising all available resources avoiding the need for idle (and expensive) hot standbys and in the process delivering the best possible response times to those important end users. It reduces the risk of demand spikes overloading any single element of your complex causing an outage. Plus with good design load balancing can achieve performance greater than the sum of the parts and delay the cost of upgrading to bigger processors as your volumes increase.

Quick and Dirty
The most simple example of load balancing is putting several IP addresses into your DNS. Each IP address represents an FEP (e.g. Cisco CIP or 3172 interface) connected to independent processors. Each processor being a miniature clone of the whole (so 8 Mini-Me's = 1 Dr Evil). Every client wishing to connect to the complex goes to the DNS with the host name (DrEvil.EvilEmpire.com) and is returned the IP address of a Mini-Me clone. The next client gets the address of a different clone and the DNS cycles through all the available clones on a round robin basis providing crude load balancing across the complex.

This works and has the advantage of cost since no new equipment is needed, but it suffers from a number of limitations based on invalid assumptions,

1) Not all processors are created equal
2) Not all clients are the same
3) The DNS is not aware of the state of the processor.

Therefore, some processors end up over taxed while others remain largely idle, clients may also remain unserviced because the processor they were directed to is unavailable.

For many this technique is often the first example of IP load balancing they will deploy and the the limitations are not an issue until they begin to scale up.

Time to Network
To overcome this various hardware manufacturers (e.g. Cisco or IBM) are keen to sell their solutions, and these split into two camps, either they act as a more sophisticated DNS replacement or they provide a VIPA (Virtual IP Address) solution.

DNS Solutions
The next logical step from using a traditional DNS is to link the DNS and the server complex so that the load balancing can be made more intelligent. The DNS replacement may run several different distribution algorithms such as weighted average, the IP address returned is then based on a predetermined notion of relative ability. DNS replacement is also frequently used where processors are geographically dispersed, the returned address could then be defined by its proximity to the client.

The latest announcements from IBM (PUT 13) drop into replacement DNS category. By providing a DNS in the complex that can monitor the available interfaces and system loads and return the most appropriate IP address it eliminates the biggest drawbacks of the DNS based solution namely that it will not return the IP address of a failed processor, while still being a cost effective strategy.

Deploying such a solution may require some reworking of your DNS policy and your naming conventions since in order to be effective the DNS must be authoritative for the TPF complex subdomain. Those of you who made TPF just another host in the datacentre/corporate domain may now want to revisit that strategy as you will now have a migration exercise.

As a solution it still has some limitations, since it is based on the DNS architecture it assumes that your client can use a DNS and it is well recognised that DNS support is not always perfect with many platforms caching the DNS results which undermines the effectiveness of the solution.

VIPA Solutions
VIPA on the other hand always returns the same address. VIPA usually comes in 2 forms internally managed and externally managed. Internally managed VIPA like that currently found on OS/390 and soon to be on TPF (PUT 13) present a single IP address to the entire network and organise all the load balancing completely invisibly and they should work with all applications. External solutions are typically 3rd party suppliers with black boxes that sit in the network between the client and the host and

may perform NAT (Network Address Translation) to map the VIPA address to one of the real addresses according to various algorithms., as such they are not always as transparent but usually offer a much more sophisticated solution as a result of higher levels of investment.

So far so good, either VIPA or DNS solutions begin to address the problem of different processor sizes, but it is only balancing new connections and over time the relative loads on each processor may drift away from the desired norm. Some clients may be long term host to host links while others are quick one shot messages, in order to address these other limitations some sort of feedback is required. The VIPA solution typically offers a least connections option by tracking the number of active links this allows for a certain degree of feedback without modifying the servers. Given sufficient granularity in the number of connections relative to their demand on resources and a reasonable rate of attrition amongst the connections a least connections option is frequently sufficient for many purposes (Web Serving typically fits this profile). The VIPA solution is also aware when hosts go offline and will balance connections away from the unavailable resource. The downside of NAT is that all traffic must go through it creating a potential bottleneck particularly on the return path where responses are usually several times larger than the inbound data. Many applications (such as IPSEC) check the IP addresses and would perceive NAT as a security violation and fail. There is also an inherrent risk of the NAT equipment becoming a single point of failure even with hot standby capability.

Moving into the enterprise space (and where else would TPF be?) more sophisticated solutions require more awareness of the server capacity. The major players here include IBM's Enterprise Network Dispatcher (eND) and Cisco's Multi Node Load Balancing (MNLB). Through detailed tracking of the server status via agents running on the servers themselves these solutions avoid the usual problems of VIPA solutions because traffic is tagged and balanced rather than NAT'd, this means that only the initial connection need go through the balancing process, responses and subsequent queries are able to go direct to the host processor reducing the balancing overhead. The equipment is also easily duplicated to scale up to eliminate any bottleneck and provide transparent resilience. Obviously this comes at a cost.

Show Me The Money

At the start of this article I said that good load balancing can actually save money, but so far I have only succeeded in spending money on new equipment, so where are the savings?
Up until now we have used the 'Mini-Me' model to provide a number of similar hosts each offering the same services, although each clone may vary in size they are essentially the same. But as anyone who runs a loosely coupled complex knows twice the processors does not give twice the performance. This is because there is an unavoidable overhead associated with managing multiple i-streams spent on conflict arbitration and avoidance techniques. As the number of Mini-Mes increase so we experience diminishing returns from our extra investment. To access this 'lost' capacity we can delegate a particular activity to a single processor. E.G. Extortion and Blackmail applications (a.k.a. Marketing) can be always delegated to a single processor, so a client wanting access to a particular application hosted on DrEvil.EvilEmpire.Com will always be mapped to the particular IP address and Port Number of the "Number One" processor that uniquely carries out these activities on behalf of the whole complex. That application is then much less likely to waste clock cycles waiting to access scarce resources such as spinning on a FIWHC or synchronising across processors and hence it runs faster for any given amount of processing power, these saved cycles can then be utilised elsewhere. Major applications like world domination will continue to be balanced across the whole complex but as a result of the sophisticated feedback will use proportionately less of the "Number One" processor.
An example of such activity is a feed of PNR data to populate a UNIX database for further processing. This can easily operate as a background task on single back-end processor within a multi-processor environment and will also avoid a common deficiency of UNIX databases which do not traditionally scale well across server farms unless designed to from the ground up. You may also be able to port an old but still useful uni-processor application into a multi-processor environment without undertaking an expensive rewrite!!

Finally, load balancing uses all your available resources improving your ROI figures, and provides more flexibility in your upgrade strategy and maintenance cycle which can offer significant savings over time. The individual amounts may be small but a couple of percent trimmed here and there all adds to the bottom line. However the big savings are derived from the improved resilience strategy, eliminating hot standby equipment offers a quickhit benefit, reductions in the likelihood and impact of any outages either planned or unplanned may offer similarly large gains. Sooner or later you will experience either planned or unplanned downtime, resilience is about dealing with it so that hopefully the end user will not notice and you can meet those all important service level agreements. While the load balancing techniques discussed frequently complement resilience strategies, an optimum load balanced solution may run counter to the needs of a resilience strategy, in the second article I will discuss how to use the equipment we have to provide the best possible resilience strategy and how to balance under utilising resources against the need to cope with downtime.