



Loadbalancer.org Appliance Setup v5.1



This document covers the basic steps required to setup the Loadbalancer.org appliances.
Please pay careful attention to the section on the ARP problem for your real server OS.

Table of Contents

Loadbalancer.org Appliance Setup v5.1.....	1
Planing.....	3
Refining the planing.....	4
Finalizing the network diagram.....	5
Example Network Diagram: Direct Routing Mode.....	6
Example Network Diagram: Network Address Translation Mode.....	7
A bit more detail on the NAT style set up.....	8
A Firewall is simple in theory but can be complex in practice:.....	10
Explaining the RIP & VIP.....	11
Configuring your Loadbalancer.org appliance.....	12
Physical network configuration.....	12
Modify Logical Virtual Server Configuration.....	13
Modify Logical Real Server Configuration.....	14
Solving the ARP problem.....	15
The procedure for Windows 2000/2003 web servers is as follows :.....	16
The procedure for Linux	19
Transparent Proxy (Horns method) - RECOMMENDED.....	19
The hidden interface approach	19
The procedure for Solaris.....	20
The procedure for Mac OS X & BSD.....	20
Testing.....	21

Planing

Setting up a load balancer from Loadbalancer.org is easy, but a little planning never hurt anyone. Deciding on your objectives is always the first step, are you looking for increased performance, reliability, ease of maintenance or all three?

Performance	A load balancer can increase performance by allowing you to utilize several cheap servers to do the work of one web site.
Reliability	Running a web site on one server gives you a single point of failure, utilizing a load balancer moves the point of failure to the load balancer. At Loadbalancer.org we advise that you only deploy load balancers as clustered pairs for this very reason.
Maintenance	Using the load balancer you can easily bring servers on and off line to perform maintenance on individual web servers.

For this example we will assume that we would like to balance the load over three Windows 2000 web servers, all three web servers are running the same web site which uses both HTTP & HTTPS. All of the web servers will talk to a shared Oracle database to handle persistence. The session table is held on the database so it doesn't matter which web server answers the client request. HTTPS is often said to require a persistent connection to the client, it doesn't, but it does help performance not to have to re-negotiate the key for every connection. When a client requests HTTPS we will configure the load balancer to keep the connection persistent (Always use the same server) for 5mins.

www.example.com currently points at a single web server with its own valid fixed IP address, the firewall then NATs this valid IP address to the web servers RIP address 192.168.1.10.

In order to provide a smooth transition, the plan is to set up the clustered load balancer on the same subnet as two new servers. Test it carefully to make sure it detects when the servers are up or down and balances the load evenly. Then when confident everything is OK change the firewall so that it NATs the external IP of www.example.com to the external floating VIP of the load balancer.

After an extended live testing period, the original web server can be added to the cluster. This method ensures that if at any stage you run into trouble you can quickly change the firewall back to the original configuration.

Refining the planing

Now step back and review the plan. Load balancers only work effectively if the web servers are completely stateless, do your web servers store persistent information on local drives?

Images (jpeg,png,gif etc.)

Files (html,php,asp etc.)

Session data (Standard ASP and PHP session data is stored locally by default!)

- 1) Your session data MUST be stored on a shared database Postgres, Oracle, MSSQL etc.
- 2) Your content either needs to be on shared storage, or replicated between each web server.

On UNIX you can use the RSYNC command to replicate files, on Win2K you need ROBOCOPY from the NT resource kit. Usually you will FTP your content to one master server and then replicate it from there to the others.

Now is also a great time to document the disaster recovery process for your web site, after all you do need to build two new web servers.

What do you do if your application is not stateless?

Re-write it! Seriously that's the best option, but if not don't worry just use persistence, you'll loose your fail over but you'll still get increased capacity. This problem occurs with all load balancers what ever the technology used to get around it.

Finalizing the network diagram

DNS name: www.example.com

Public IP address: 213.213.213.213

Current internal web server address: 192.168.1.10

DNS server: 192.168.1.2

Default Gateway (firewall/router) 192.168.1.1

Loadbalancer.org VIP (Virtual IP) 192.168.1.20

#only active on one load balancer

Loadbalancer.org RIP1 (Real IP 1) 192.168.1.21

Master

Masters Default Gateway 192.168.1.1

Loadbalancer.org RIP2 (Real IP 2) 192.168.1.22

Slave

Slaves Default Gateway 192.168.1.1

Web Server 1 RIP 192.168.1.50

Web Server 1 VIP 192.168.1.20

#So it can answer requests directly

Web 1 Default Gateway 192.168.1.1

Web Server 2 RIP 192.168.1.60

Web Server 2 VIP 192.168.1.20

#So it can answer requests directly

Web 2 Default Gateway 192.168.1.1

And to be added later.. Or roll back to in the unlikely event of a problem.

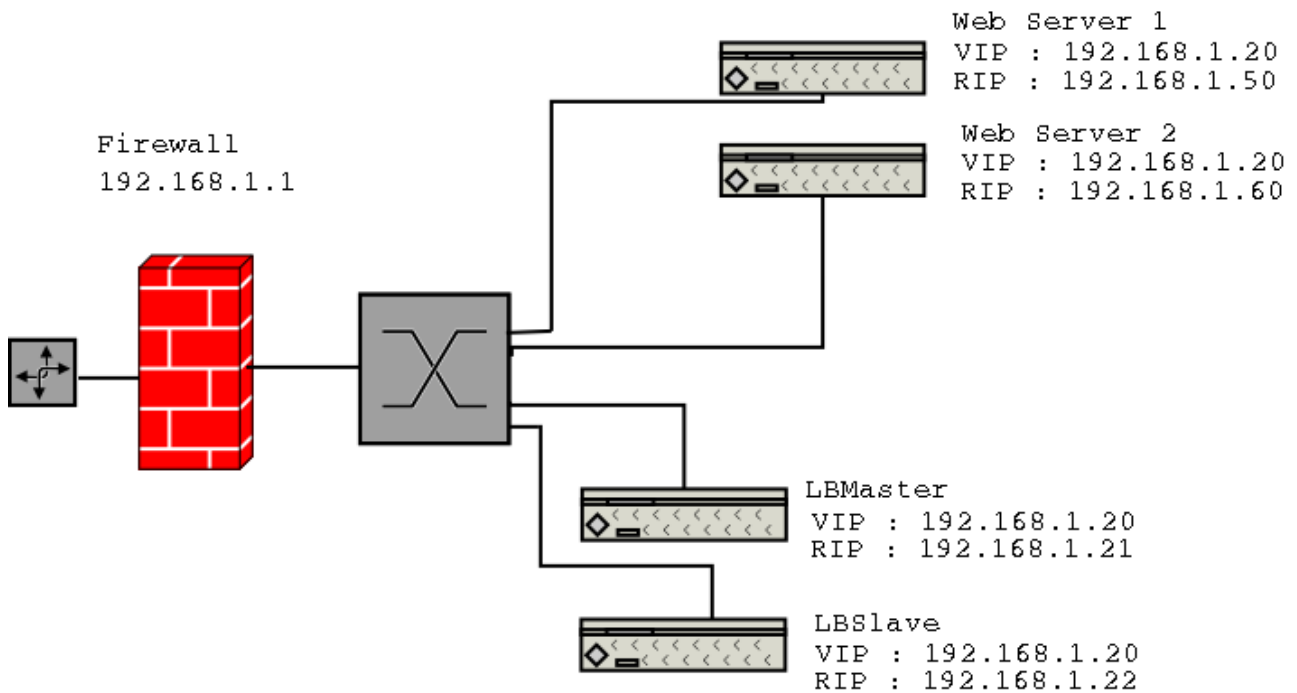
Web Server 3 RIP 192.168.1.10

Web Server 3 VIP 192.168.1.20

#So it can answer requests directly

Web 3 Default Gateway 192.168.1.1

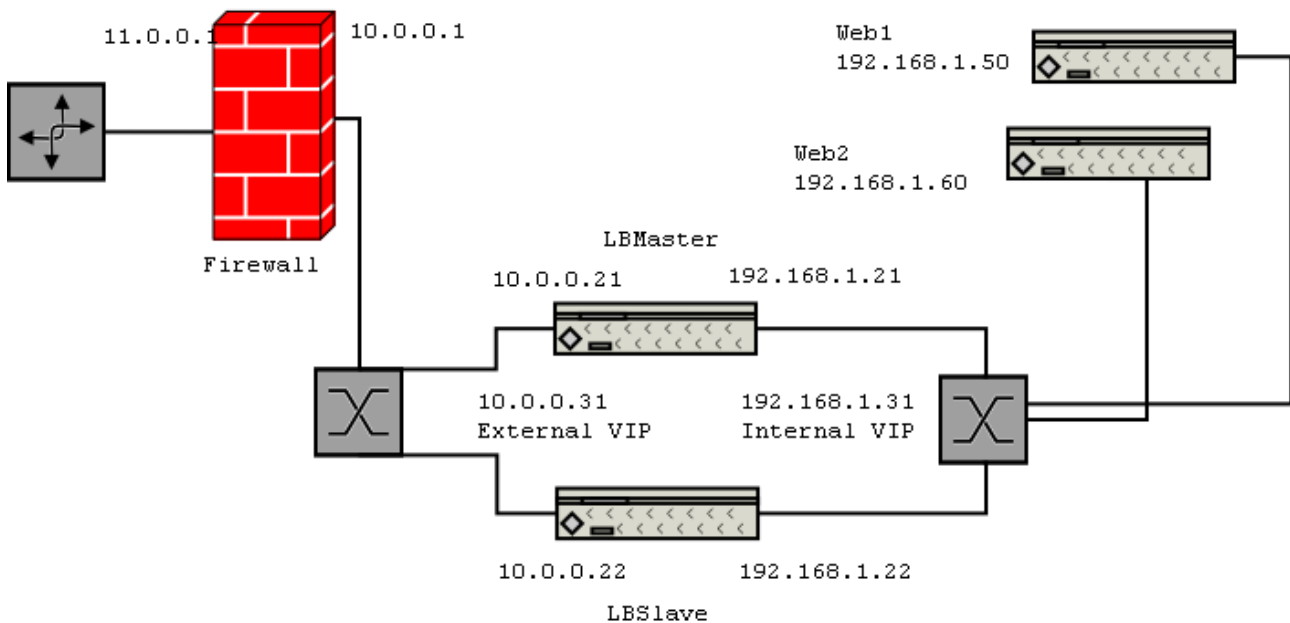
Example Network Diagram: Direct Routing Mode



Notes :

- Direct routing works by changing the destination MAC address of the incoming packet on the fly which is VERY fast.
- BUT it means that when the packet reaches the real server it expects it to own the VIP, this means you need to make sure the real server responds to the VIP, BUT DOES NOT RESPOND TO ARP REQUESTS.
- On average DR mode is 8 times quicker than NAT for HTTP, 50 times quicker for Terminal Services and much, much faster for streaming media or FTP.

Example Network Diagram: Network Address Translation Mode



Notes :

- The Loadbalancer.org appliance can work in Dual NIC NAT mode by default. Just set up your VIP with a default mode of MASQ instead of GATE.
- All the internal NAT will be handled automatically for balanced services on the VIP.
- BUT you will need to set up two Physical Virtual IP addresses (one for the internal floating IP, and one for the external floating VIP.)
- Your real servers will need their default gateway changed to the internal floating IP address.
- If you want the real servers to be able to access the Internet on their own i.e. Browse the web you will need to set up a MASQ rule in the firewall script.
- If you want real servers to be accessible on their own IP address for non-load balanced services i.e. SMTP you will need to setup individual SNAT and DNAT firewall script rules for each real server.
- You can also do Single NIC NAT in exactly the same way, but instead of using a second NIC just set up an alias for the second network in your firewall script. *NB. When doing single NAT make sure you disable redirects in the rc.firewall script an example is given in the default script.*
- Because of these issues (which are common to all load balancers) we recommend that you use the default Direct Routing Setup when it is possible.

A bit more detail on the NAT style set up

The NAT style of load balancing does have the advantage that the only change to the real servers is to modify the default gateway, IP address and subnet. You can also utilise the added security of having your real servers hidden in a subnet behind the load balancer. However in our honest opinion we think its a bit daft to use your load balancer as a firewall, it adds complexity and while the Loadbalancer.org appliance can be configured to be rock solid secure *you should at least be fully aware of what you are doing if it is going to be your bastion host.*

There is no harm in putting a pair of Loadbalancer.org appliances in NAT mode behind your own firewall solution as shown in the diagram (a so called double NAT).

In order to use NAT mode on the load balancers you'll need a couple of things :

1. You need an external and internal floating VIP (physical virtual IP address)
2. The external one is the one the clients connect to
3. The internal one is the default gateway for the real servers
4. Set your virtual server to use the MASQ method and hey presto you are done.

BUT :

1. Your real servers won't be able to access the Internet through the new default gateway (except when replying to requests made through the external VIP)
2. External (non-load balanced) services such as FTP or SMTP will not be accessible because you haven't exposed any public IPs.

To solve this:

- 1) You need to add a line to the *rc.firewall* script on the load balancer to allow all outgoing traffic from the internal network to be MASQUERADED.

i.e.

```
$INT_SUBNET="192.168.1.0/255.255.255.0"
iptables -t nat -A POSTROUTING -s INT_SUBNET -j MASQUERADE
# i.e. Everything coming from the internal subnet should be
automatically NATed to the external subnet
# If you don't do this you will have no Internet access from your
real servers (which may not be required)
```

2) If you want any specific services to be exposed for your real servers you have two choices :

a) Set up a specific virtual server with a single real server for the service i.e. Just one real server in the FTP group.

Or

b) Set up individual public IPs for the services required with individual SNATs and DNATs for each service required i.e.

```
# SNAT & DNAT all traffic from EXT_MAIL to INT_MAIL
# NB. You will need a floating VIP set up for the external ip if
you haven't got one already
$INT_MAIL="192.168.1.13"
$EXT_MAIL="234.23.45.236"
# MAIL
iptables -t nat -A POSTROUTING -o $EXT_IFACE -p tcp -s $INT_MAIL -
j SNAT -to-source $EXT_MAIL
iptables -t nat -A PREROUTING -i $EXT_IFACE -p tcp -d $EXT_MAIL -j
DNAT -to-destination $INT_MAIL
#NB. Obviously this should now be locked down with ACCEPT & DENY
rules on FORWARD chain
```

A Firewall is simple in theory but can be complex in practice:

Understand what you are trying to achieve and how to go about it in the *rc.firewall* script may look a bit scary but it uses Linux netfilter which is an excellent transferable skill to learn.

If you want to setup a complex NAT solution, or use the Loadbalancer.org appliances as bastion hosts then here are a couple of pointers:

1. All virtual server connections are dealt with on the INPUT chain NOT the FORWARD chain.
2. The SNAT & DNAT is handled automatically for all the Virtual/Real load balanced services.
3. HTTP, HTTPS & SSH are by default OPEN on the INPUT chain i.e. If you have a public ip for your VIP someone can use HTTP to get to the local apache installation on the load balancer, unless you :
 - a) Set up a real server group for HTTP (and HTTPS & SSH).
 - b) Firewall the appliance! (*either using your firewall or the rc.firewall script or both*)
4. You can use the standard Linux filters against spoofing attacks and syn floods etc.
5. LVS has built in DOS attack filters that can be implemented
6. If in doubt take a look at the excellent documentation on the www.linuxvirtualserver.org site.

Explaining the RIP & VIP

RIP is the Real IP address of a server and VIP is the Virtual IP address of the cluster. You can have as many VIPs as you like but for this example we are only using one.

Any web request to the VIP (from inside or outside your network) will have the MAC address of the packet changed so that it goes directly to the web server with the least connections. When the packet arrives at the web server it will declare its destination as the VIP (10.0.0.20), The web server would normally say 'No sorry thats not me' and drop the packet (not what we want.) So we need to fool each web server into believing that it IS 10.0.0.20, BUT NOT to tell the rest of the network. Otherwise we'd have lots of server advertising the same IP address (This is called the ARP problem)

Configuring your Loadbalancer.org appliance

This section deals with the process of configuring the load balancers.

Physical network configuration

- Power up the slave load balancer first
- Configure the physical IP address (the internal address that you will use for administration) it is 10.0.0.21/255.255.0.0 by default. Just log into <http://10.0.0.21/lbadmin> as the user **loadbalancer** with the password of **loadbalancer**.
- Then use Edit configuration / 'Modify the physical network configuration of this load balancer'
- Set the required IP address, subnet mask & default gateway for your slave load balancer
- Click 'Update IP Configuration'
- Go back to 'Modify the physical network configuration of this load balancer'
- Make sure you set the hostname to lbslave and enter at least one DNS server address.
- Click 'Update Network Configuration'
- Make sure that the serial (Null modem cable) is attached between the master & slave load balancer for the heartbeat signal and also make sure they are both plugged into the same network switch before turning on the master load balancer.
- After the master has booted, just log into <http://10.0.0.21/lbadmin/> as the user loadbalancer with the password of loadbalancer.
- Then use Edit configuration / 'Modify the physical network configuration of this load balancer'
- Set the required IP address, subnet mask & default gateway for your master load balancer.
- Click 'Update IP Configuration'
- Then go back to 'Modify the physical network configuration of this load balancer'
- Make sure you set the hostname to lbmaster and enter at least one DNS server address.
- AND you must tell the master load balancer the IP address of the slave load balancer. (Once this step is complete all changes on the master will be replicated via the network to the slave using an encrypted connection.)
- Click 'Update Network Configuration'

Modify Logical Virtual Server Configuration

- You need to tell the master load balancer which service you want to load balance. Go to Edit Configuration / Modify logical Virtual Servers and remove the default virtual servers.
- Add a new virtual server. The virtual server is added in the following format ipaddress:portno. It basically means that any packet arriving at the load balancer with that IP address and that port number will be handled by the logical real servers associated with this logical virtual server.

Virtual Server Label	VIP Name
Virtual Server (ipaddress:port)	10.0.0.20:80
Do you want Sticky Connections ? (recommended for HTTPS)	no ▼
Automatically add the related physical VIP ?	<input checked="" type="checkbox"/>

Add A New Virtual Server

- You don't need to change any of the default Virtual Server settings but by all means take a look.
- Once you have setup your logical VIPs e.g. 192.168.1.20:80 and 192.168.1.20:443 you need to add some logical real servers (web servers) to the cluster.
- You may need to restart the heartbeat on both the master and the slave or re-boot both boxes after you have set up new physical floating VIPs. This is to ensure that the heartbeat doesn't get out of sync. *NB. With a correctly configured clustered pair this is however an automatic process. Just check 'View Configuration | Current network configuration' to ensure that the IP address has been activated correctly.*

Modify Logical Real Server Configuration

- Go to Edit Configuration / Modify logical Real Servers and you should see your logical virtual servers listed, select the one you want and click on **add real server**.
- You just need to give the **ipaddress:portno** of your web server and specify a relative weight. A weight of 0 is the default (which you'll obviously need to change.)

Real Server Label :	RIP Name
Real Server (ipaddress:port) :	IPAddress:80
Real Server Weight :	0
Forwarding method	gate ▼

Add New Real Server to Virtual Server 1

- Add as many real servers as required. (*Set the weight to 1 or more if you want them to be active immediately*)
- You have now finished configuration of the load balancer.
- **BUT YOU MUST configure the VIP on each of your web servers**, and ensure that the web servers are responding to the VIP address as well as the RIP address (health checks are on the real address.)

Solving the ARP problem

Each web server needs a loopback IP address to be configured as the VIP 192.168.1.20. This address needs to be stopped from responding to ARP requests and the web server needs to be configured to respond to this IP address.

Why?

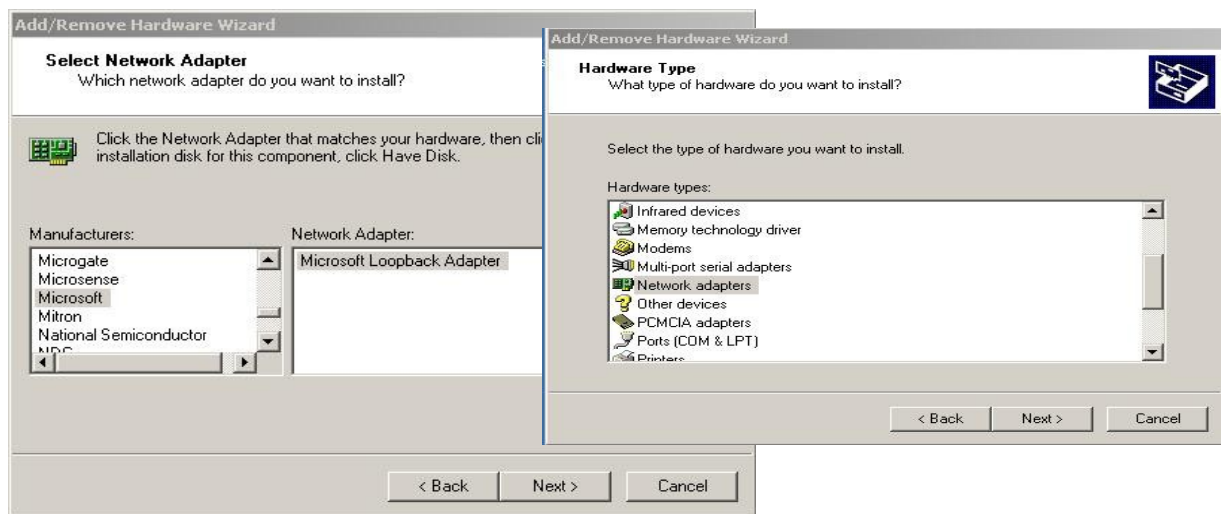
It is important that your web servers do not fight with the load balancer for control of the shared VIP. If they do then request will be sent directly to the web servers rather than hitting the load balancer VIP as intended.

NB. You only need to resolve the ARP issue on the real servers when you are using the default DR (Direct Routing) load balancing method or IPIP (TUN or IP encapsulation). If you are using NAT mode you don't need to make any changes to the real servers except to make sure the load balancers internal floating IP address needs to be set as the default gateway.

The procedure for Windows 2000/2003 web servers is as follows :

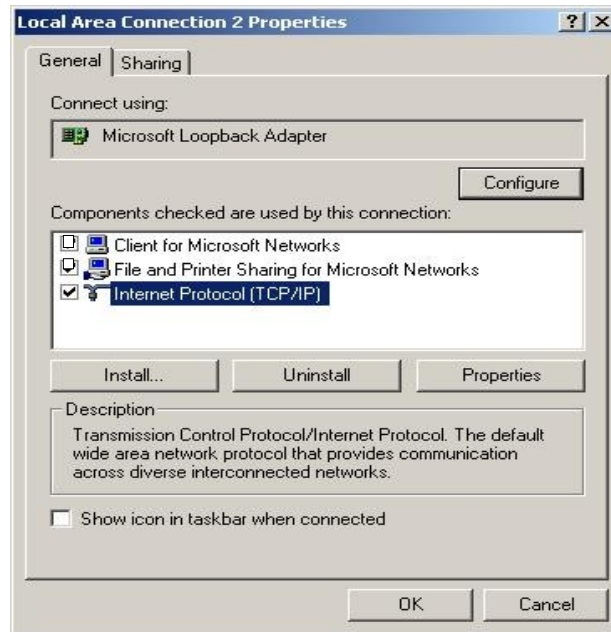
First you need to install the MS Loopback Adapter :

1. Click Start, point to Settings, click Control Panel, and then double-click Add/Remove Hardware.
2. Click Add/Troubleshoot a device, and then click Next.
3. Click Add a new device, and then click Next.
4. Click No, I want to select the hardware from a list, and then click Next.
5. Click Network adapters, and then click Next.
6. In the Manufacturers box, click Microsoft.
7. In the Network Adapter box, click Microsoft Loopback Adapter, and then click Next.
8. Click Finish.

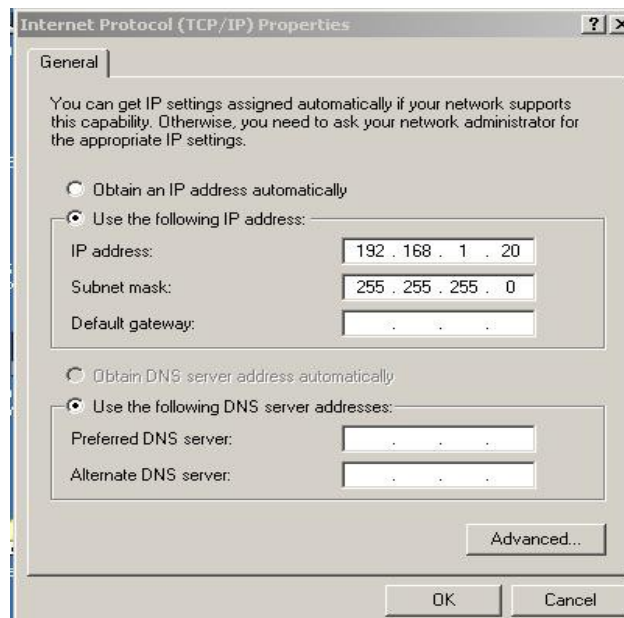


Once the Loopback Adapter is installed you need to configure it with an extra IP Address

1. Click Start, point to Settings, click Control Panel, and then double-click Network and Dial up Connections.
2. Right click the new local adapter and select properties
3. Remove the tick from Client for Microsoft Networks
4. Remove the tick from File and Printer Sharing for Microsoft Networks

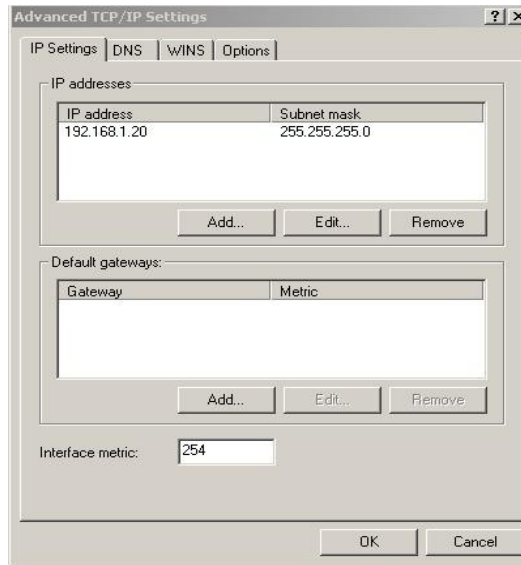


3. Select TCP/IP properties



4. Fill in the VIP 192.168.1.20 and the subnet mask.
5. DO NOT ENTER A DEFAULT GATEWAY!
6. Click on the **Advanced...** button

7. Change the Interface Metric to 254 (This stops the adapter responding to ARP requests)
8. Click OK and save all changes



And repeat the above process for all of the web servers.

The procedure for Linux

Ironically Linux is a pain because the lo interface responds to ARP requests by default!
Use the following to get around this :

Transparent Proxy (Horns method) - RECOMMENDED

You can side step the issue for both by using iptables on the real server to re-direct incoming packets destined for the VIP. Many people find this a lot less scary than a Kernel upgrade on a live web server... (Kernel upgrades are easy honest but always practice on a test box.)

This is a simple case of adding the following command to your rc.firewall script :

```
iptables -t nat -A PREROUTING -p tcp -d VIP -j REDIRECT
```

i.e. Redirect any incoming packets destined for 10.0.0.20 (the VIP) to my local address.

You can also do the same with 2.2 Kernels and ipchains (but why not just upgrade your kernel?)

```
ipchains -A input -s 0/0 -d VIP -p tcp -y -j REDIRECT
```

NB. You don't need to configure a loopback adapter when using this method.

OR . . .

The hidden interface approach

If you have the hidden interface patch installed (Mandrake/SuSe) you can use this method or you can patch your kernel, Red Hat patches are available at www.ultramonkey.org.

The configuration instructions to hide interface from ARP for LVS are as follows:

```
# Start the hiding interface functionality
echo 1 > /proc/sys/net/ipv4/conf/all/hidden
# Hide all addresses for this interface
echo 1 > /proc/sys/net/ipv4/conf/lo/hidden
# Now configure the VIP
ifconfig lo:1 VIP netmask 255.255.255.255
```

Note that once an interface is set hidden, all the addresses of the interface is hidden from ARP broadcast and being included in the ARP response of other addresses. So, it is not good to configure VIP on the aliases of real Ethernet interfaces and make it hidden, unless you have a unused Ethernet interface.

For LVS/DR clusters, it is good to configure VIPs on the aliases of dummy or loopback device and hide the corresponding device. Then, you can configure as many VIPs as you want.

The procedure for Solaris....

Solaris is nice and easy, the loopback interface does not respond to ARP requests so you just add your VIPs to it.

```
ifconfig lo0:1 plumb  
ifconfig lo0:1 VIP netmask 255.255.255.255 up
```

You will need add this to your start up scripts for your server.

The procedure for Mac OS X & BSD....

OS X is BSDish, so you need to use BSDish syntax:

```
ifconfig lo0 alias VIP netmask 255.255.255.255 -arp up
```

You will need add this to your start up scripts for your server.

Testing

For testing add a page to each real web servers root directory i.e. Test.html and put the server name on this page.

Now you need a couple of clients to do the testing, Open up a web browser on two different clients and enter the URL for the VIP i.e. 192.168.1.20 .

Each client should see a different server name because of the load balancing algorithm in use.

Why two test clients? If you use a single client it will most likely keep on hitting the same server for multiple requests this is to do with the way the TCP protocol works.

Pull the Network cable out of one of the web servers , wait 6 seconds (for the load balancer to detect the change) and then refresh the browsers on both clients, they should both now switch to the same server (as one has been removed from the load balancing list).

Put the Network cable back in to the web server, wait 6 seconds and then refresh the browsers again, they should now show different web servers again.

NB. If you want to test the fail over of the actual load balancers make sure you power one of the boxes down **DONT JUST PULL THE SERIAL CABLE OUT!** This will cause big problems...

Also be careful of your ARP cache (arp -a) make sure it is empty on the client before testing fail over.

If you are having any kind of difficulty at all 'arp -a' is your friend.