Creating Web Farms with Linux
(Linux High Availability and Scalability)

Horms (Simon Horman)
horms@verge.net.au

December 2001

For Presentation in Tokyo, Japan

http://verge.net.au/linux/has/
http://ultramonkey.org/

Introduction

This presentation will focus on:

What technologies are available for High Availability and Scalability.

Building a Highly Available, Load Balanced Cluster
using Ultra Monkey.

What is High Availability?

In the context of this paper:

The ability to provide some level of service during a situation where one or more components of a system has failed.

The failure may be scheduled or unscheduled.

What is High Availability?

The key to achieving high availability is to eliminate single points of failure.

A single point of failure occurs when a resource has only one source:

- A web presence is hosted on a single Linux box running Apache.
- A site that has only one link to the internet.

What is High Availability?

Elimination of single points of failure inevitably requires provisioning additional resources.

It is the role of high availability solutions to architect and manage these resources.

The aim is that when a failure occurs users are still able to access the service.

This may be a full or degraded service.

What is Scalability?

Scalability refers to the ability to grow services in a manner that is transparent to end users.

Typically this involves growing services beyond a single chassis.

DNS and layer 4 switching solutions are the most common methods of achieving this.

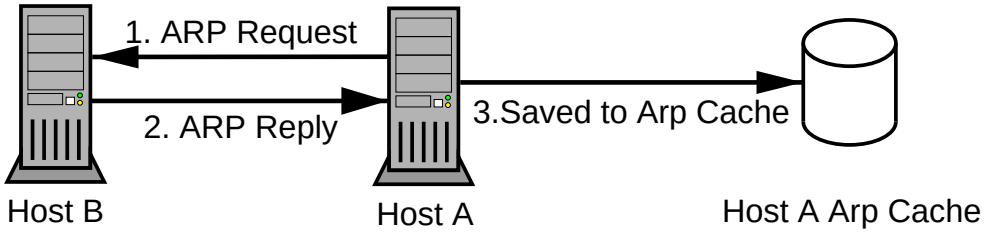
Data replication across machines can be problematic.

Technologies: IP Address Takeover

If a machine, or service running on a machine, becomes unavailable, it is often useful to substitute another machine.

IP Address Takeover allows a hot stand-by to assume the IP address of a master host.

IP Address Takeover: ARP



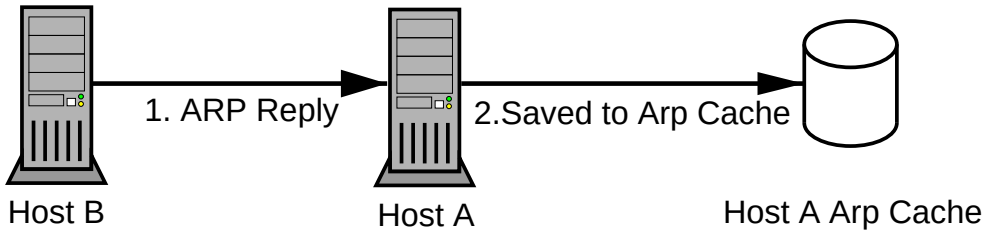
1. Host A sends out an ARP request for the hardware address of an IP address on host B.

2. Host B sees the request and sends an ARP reply containing the hardware address for the interface with the IP address in question.

3. Host A then records the hardware address in its ARP cache.

Entries in an ARP cache typically expire after about two minutes.

IP Address Takeover: Gratuitous ARP



A gratuitous ARP is an ARP reply when there was no ARP request.

If addressed to the broadcast hardware address, all hosts on the LAN will receive the ARP reply and refresh their ARP cache.

IP Address Takeover: Gratuitous ARP

If gratuitous ARPs are sent often enough:

- No host's ARP entry for the IP address in question should expire.
- No ARP requests will be sent out.
- No opportunity for a rogue ARP reply from the failed master.

Technologies: Layer 4 Switching

Ability to load balance connections received from end-users to real servers.

This can be implemented in an ethernet switch such as the Alteon Networks ACESwitch.

It can also be done in a host such as a Linux box with the Linux Virtual Server (LVS).

Layer 4 Switching: Virtual Server

A Virtual Server is the point of contact for by end-users and is typically advertised through DNS.

A virtual server is defined by:

- An IP address, port and protocol (UDP/IP or TCP/IP)
- Matches made by a packet filter

Layer 4 Switching: Scheduling Algorithm

The virtual service is assigned a scheduling algorithm.

The scheduling algorithm determines which back-end server an incoming TCP/IP connection or UDP/IP datagram will be sent to.

Packets for the life of a TCP/IP connection will be forwarded to the same back-end server.

Optionally, subsequent TCP/IP connections or UDP/IP datagrams from a host or network to be forwarded to the same back-end server.

This is useful for applications such as HTTPS where the encryption used relies on the integrity of a handshake made between the client and a server.

Layer 4 Switching: Forwarding

When a packet is to be forwarded to a back-end server, several mechanisms are commonly employed:

- Direct Routing
- IP-IP Encapsulation (Tunnelling)
- Network Address Translation (NAT)

Technologies: Heartbeat

A heartbeat is a message sent between machines at a regular interval of the order of seconds.

If a heartbeat isn't received for a time, the machine that should have sent the heartbeat is assumed to have failed.

Used to negotiate and monitor the availability of a resource, such as a floating IP address.

Technologies: Heartbeat

Typically when a heartbeat starts on a machine it will perform an election process with other machines.

On heartbeat networks of more than two machines it is important to take into account network partitioning.

When partitioning occurs it is important that the resource is only owned by one machine, not one machine in each partition.

Heartbeat: Transport

It is important that the heartbeat protocol and the transport that it runs on is as reliable as possible.

Effecting a fail-over because of a false alarm may be highly undesirable.

It is also important to react quickly to an actual failure.

It is often desirable to have heartbeat running over more than one transport.

Putting It Together: Ultra Monkey

Suite of tools to enable a load balanced, highly available farm of servers on a LAN.

LVS for load balancing.

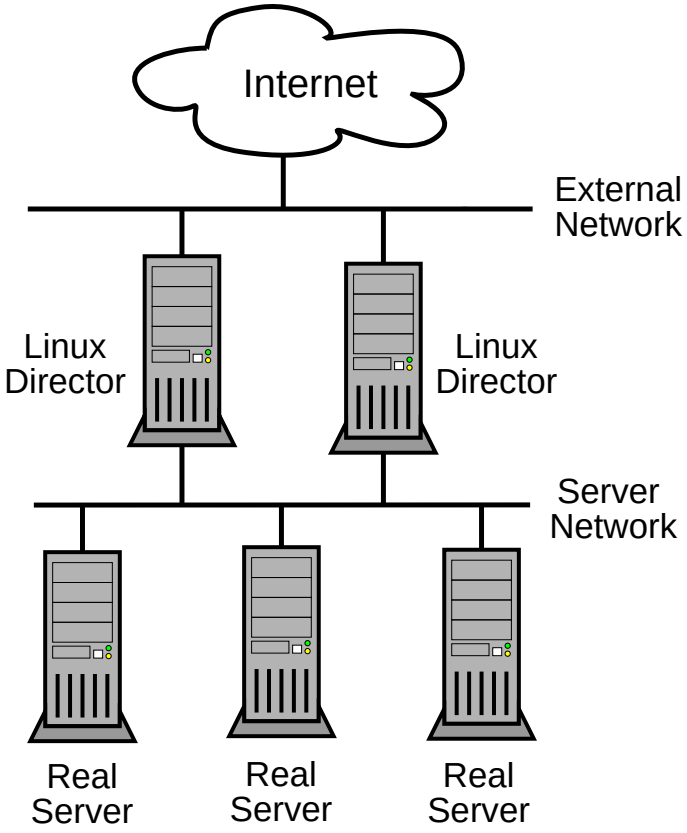
Heartbeat for High Availability between load balancers

Ldirectord monitors the health of real servers

Sample topologies for Highly Available and/or Load Balanced networks are provided.

Single Point of Contact for an Open Source solution.

Sample Configuration



Ultra Monkey: Linux Directors

Top layer of servers handles load balancing clients to real servers.

Incoming traffic travels through the active Linux Director.

The other Linux Director is a hot stand-by.

Ultra Monkey: Linux Directors

Heartbeat is run between the two Linux Directors.

IP address Takeover is effected if the master Linux Director fails.

Other resources may be defined

Ultra Monkey: Linux Directors

Ldirectord is used to monitor the real servers

A page is requested periodically and the output parsed

Adds and removes servers from the load balancing pool as necessary

Sample Web Farm: Real Servers

Any processing of client requests is done by these servers.

Processing power can easily be increased by adding more servers.

Where possible state should is stored on clients by either using cookies or encoding the state into the URL.

- Client doesn't need to repeatedly connect to the same server.
- More flexible load-balancing.
- Session can continue even if a server fails.

Conclusion

High Availability and Scalability are critical issues for network based services.

Using a variety of well known techniques this can be realised.

Ultra Monkey provides packages and documentation for deploying this under Linux.

Creating Web Farms with Linux
(Linux High Availability and Scalability)

Horms (Simon Horman)
horms@verge.net.au

December 2001

For Presentation in Tokyo, Japan

http://verge.net.au/linux/has/
http://ultramonkey.org/