ISP Failover and Load Balancing

Final Report

Supervised by

Dr Timothy Anthony Gonsalves

Submitted by

Umang Jain, Vikhyat Korrapati, J.P Jangid

Table of Contents

1. Introduction

1.1 Intended Purpose

Growing organizations are typically served by multiple ISPs in order to provide greater cumulative bandwidth by load-balancing and availability via automatic failover. There is a definite need for a generic tool to facilitate this which can run on a generic server because dedicated hardware will become obsolete as the organization grows.

1.2 Intended Audience

This document is intended for the users, developers and administrators of this system. It provides details of the kind of interfaces that end-users and administrators can expect, and also provides valuable information for the implementers of the system.

1.3 References

[1] http://redis.io/documentation
[2] http://www.squid-cache.org/
[3] http://www.cyberciti.biz/faq/what-is-a-routing-table/
[4] http://www.thegeekstuff.com/2012/04/ip-routing-intro/
[5] https://redis-py.readthedocs.org/en/latest/
[6] http://en.wikipedia.org/wiki/Iptables
[7] Software Requirements Specification Doc , submitted by us

2. Detailed Design

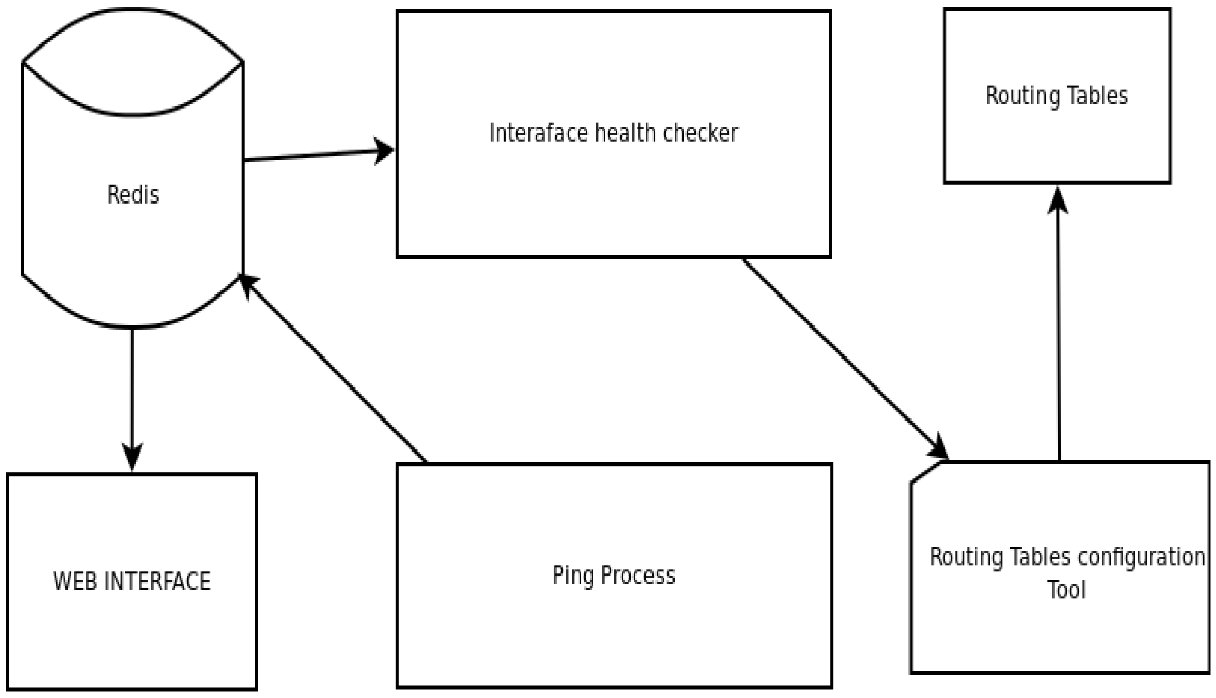
2.1 Design Overview

Failover: The system sends ICMP echo requests over all available interfaces in order to determine the status of each interface. It updates the routing table to make sure that all traffic is routed to an active interface in the order of precedence specified by the network administrator.

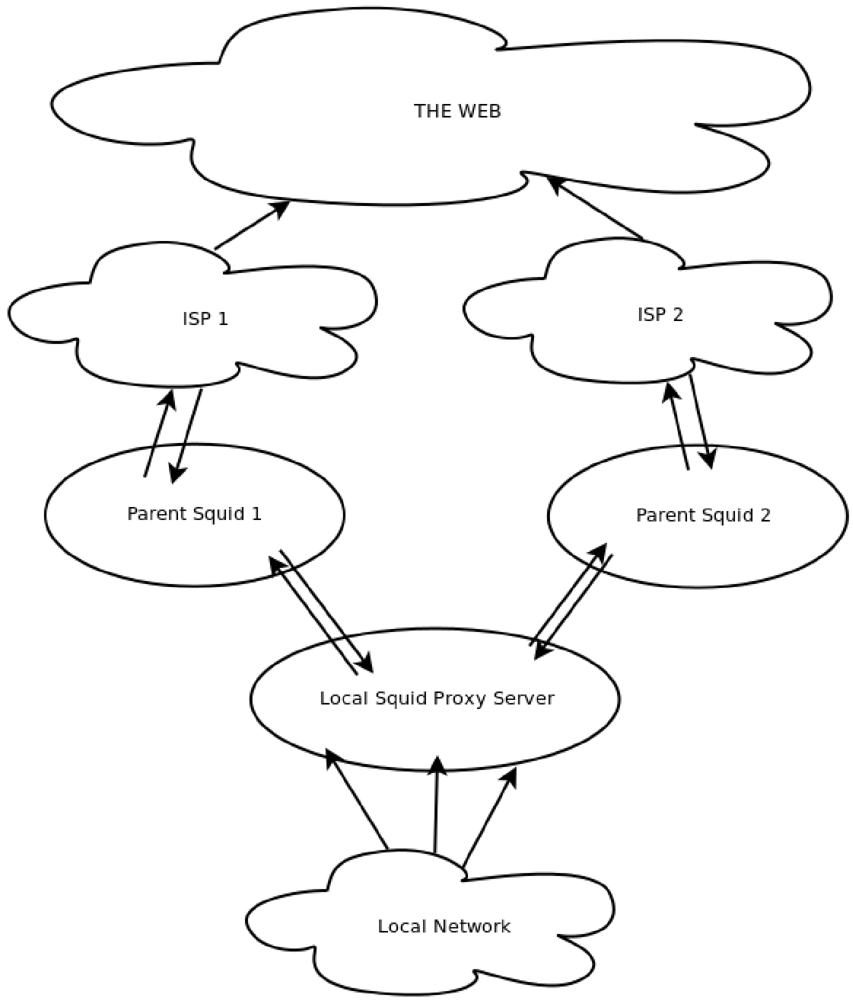
Load-balancing: Squid is a software package that implements a caching proxy server for the web. It reduces bandwidth and improves response time by caching and reusing frequently-requested webpages. Squid can also be used for load-balancing by arranging multiple Squid processes running in virtual or physical machines in a hierarchy with one child which the end-users connect to.

2.2 Architecture

Failover



Load Balancing



2.1.1 Components

1. Ping Process:
   This process will (at an interval of 1 second) send ICMP requests via different uplinks and add to the database a binary value indicating whether or not the last packet transmission was successful.
2. Health Checker:
   This process checks the Redis database periodically and calculates the packet loss at each interface over a fixed period of time. It then decides according to packet loss statistics of each interface if it should be used or not and updates a TRUE/FALSE value on the corresponding *interface:status* variable.

2.1.2 Interfaces

1. Redis Database Interface:

   Ping interface script writes the ping statistics to the Redis Database. The Link Status Monitor periodically reads the content from the Redis Database and publishes the status of the interface. The IPTABLES Manager process is subscribed to the Redis database receives a message whenever an i*nterface:status* variable is changed.

2. Web Interface:

   The ping statistics and link statuses shall be exposed by a web interface for access by System Administrators.

2.3 Algorithms and Data Structures

1. Squid will be configured to use the weighted-round robin algorithm to load balance between the two ISP-specific parent caches. This algorithm takes into account both the latency of the parent cache and a user-specified weight.

2. Redis Sorted Sets : Redis Sorted Sets are non repeating collection of strings. Every member of a sorted set in associated with a score that is used to take the sorted set in ordered from the smallest to the greatest score. With Sorted Sets we could implement add, remove operations in time proportional to the logarithm of the number of elements.

2.3 External Data

2.3.1 Databases

There is no external database used by the hierarchy of squid proxies that we use load-balancing. The ping process stores the result of each ICMP echo request in the Redis database. These values are read by the health monitor process, which determines on the basis of a threshold which interfaces are active and which are not. The web interface also displays the same statistics from Redis.

2.3.2 Files

The difference between the squid instances running in different virtual machines is the squid configuration file. No other data is stored in files.

1. For the failover the network administrator needs to specify the network interface precedence order in a configuration file.

2. There is a file(“ping.py”) which creates and manages the process that pings the given interfaces and updates their values in the corresponding Redis Database Tables.
3. The file, “health_monitor.py” creates the process that makes the decision whether an interface is up or down

2.4 Performance

The statistics, logging and load balancing should not consume an excessive amount of server resources, and must not compromise the connection speed users get.
During our tests , We got a maximum failover time of less that 3 Minutes.

2.5 Test Scripts

We test the product using different combinations of a wireless network, ethernet connection and mobile tethering. We simulate interfaces going up and down using IPTables.

3. Implementation Procedure

3.1 Failover

1. Install Redis Database package on the system with the command

   “sudo apt-get install redis”

2. Install the python libraries for using Redis using the command
   “sudo apt-get install pip”
   “sudo pip install redis”

3. Enter the interfaces to be used in the interfaces file
4. Run the “ping.py” script with root privelages
   “sudo python ping.py”
5. Run the “health_monitor.py” with root privelages
   “sudo python health_monitor.py”

3.2 Load Balancing

1. Set up three virtual machines with the names Parent 1, Parent 2 and Local Squid
2. In the Virtual box settings configure the Parent 1 and Parent 2 to use the two interfaces between which we need to balance the traffic
3. Install Squid Proxy Software on all the three virtual machines using the command
   ```
   “sudo apt-get install squid3”
   ```
4. Configure the Squid in Local Squid to connect to the two parents Parent 1 and Parent 2 using the commands
   ```
   “cd /etc/squid3/”
   ```
   Edit the squid.conf file in this directory by adding the following lines in it
   ```
   cache_peer parentcache1.foo.com parent 3128 0 no-query round-robin
   cache_peer parentcache2.foo.com parent 3128 0 no-query round-robin
   never_direct allow all
   ```
5. Ask the clients to connect to the Local Squid Proxy server that we set up