CERN DNS
Load Balancing

Vladimír Bahyl  IT-FIO
Nicholas Garfield  IT-CS
Outline

- Problem description
- Possible solutions
- DNS – an ideal medium
- Round robin vs. Load Balancing
- Evolution of DNS setup at CERN
- Application Load Balancing system
  - Server process
  - Client configuration
- Operational examples
- Statistics
- Conclusion
Problem description

- User expectations of IT services:
  - 100% availability
  - Response time converging to zero

- Several approaches:
  - Bigger and better hardware (= increasing MTBF)
  - Redundant architecture
  - Load balancing + Failover

- Situation at CERN:
  - Has to provide uninterrupted services
  - Transparently migrate nodes in and out of production
    - Caused either by scheduled intervention or a high load
  - Very large and complex network infrastructure
Possible solutions

- Network Load Balancing
  - A device/driver monitors network traffic flow and makes packet forwarding decisions
  - Example: Microsoft Windows 2003 Server NLB
  - Disadvantages:
    - Not applications aware
    - Simple network topology only
    - Proprietary

- OSI Layer 4 (the Transport Layer – TCP/UDP) switching
  - Cluster is hidden by a switch behind a single virtual IP address
  - Switch role also includes:
    - Monitoring of all nodes in the cluster
    - Keep track of the network flow
    - Forwarding of packets according to policies
  - Example: Linux Virtual Server, Cisco Catalyst switches
  - Disadvantages:
    - Simplistic tests; All cluster nodes should be on the same subnet
    - Expensive for large subnets with many services
    - Switch becomes single point of failure
Domain Name System – ideal medium

😊 Ubiquitous, standardized and globally accessible database
😊 Connections to any service have to contact DNS first
😊 Provides a way for rapid updates
😊 Offers round robin load distribution (see later)

😊 Unaware of the applications
  - Need for an arbitration process to select best nodes
    - Decision process is not going to be affected by the load on the service

✔ Application load balancing and failover
DNS Round Robin

- Allows basic load distribution

```bash
lxplus001 ~ > host lxplus
lxplus.cern.ch has address 137.138.4.171     (1)
lxplus.cern.ch has address 137.138.4.177     (2)
lxplus.cern.ch has address 137.138.4.178     (3)
lxplus.cern.ch has address 137.138.5.72      (4)
lxplus.cern.ch has address 137.138.4.169     (5)

lxplus001 ~ > host lxplus
lxplus.cern.ch has address 137.138.4.177     (2)
lxplus.cern.ch has address 137.138.4.178     (3)
lxplus.cern.ch has address 137.138.5.72      (4)
lxplus.cern.ch has address 137.138.4.169     (5)
lxplus.cern.ch has address 137.138.4.171     (1)
```

😢 No withdrawal of overloaded or failed nodes
DNS Load Balancing and Failover

- Requires an additional server = arbiter
  - Monitors the cluster members
  - Adds and withdraw nodes as required
  - Updates are transactional
    - Client never sees an empty list

```
 lxplus001 ~> host lxplus
 lxplus us. cern. ch has address 137.138.4.168
 lxplus us. cern. ch has address 137.138.4.171
 lxplus us. cern. ch has address 137.138.4.174
 lxplus us. cern. ch has address 137.138.4.177
 lxplus us. cern. ch has address 137.138.4.166
 lxplus us. cern. ch has address 137.138.4.169
 lxplus us. cern. ch has address 137.138.4.170
```

DNS evolution at CERN – static model

Not scalable.
No high-availability.
Obsolete.

AXFR = full zone transfer
IXFR = incremental zone transfer
zone = [view, domain] pair (example: [internal, cern.ch])
DNS evolution at CERN – scalable model

No support for frequent 3rd party updates. Only full zone transfers.

AXFR = full zone transfer
IXFR = incremental zone transfer
zone = [view, domain] pair (example: [internal, cern.ch])
DNS evolution at CERN – 3rd party updates

Multiple points of failure.
Management nightmare.

AXFR = full zone transfer
IXFR = incremental zone transfer
zone = [view, domain] pair (example: [internal, cern.ch])
DNS evolution at CERN – Dynamic DNS

AXFR = full zone transfer
IXFR = incremental zone transfer
zone = [view, domain] pair (example: [internal, cern.ch])
Application Load Balancing System

Application Cluster

node1: metric=24
node2: metric=48
node3: metric=35
node4: metric=27

Load Balancing Arbiter

2 best nodes for application.cern.ch:
node1
node4

DNS Server

A: application.cern.ch resolves to:
node4.cern.ch
node1.cern.ch

Q: What is the IP address of application.cern.ch?
A: application.cern.ch resolves to:
node4.cern.ch
node1.cern.ch

Connecting to node4.cern.ch

Connecting to node4.cern.ch
Load Balancing Arbiter

- Collects metric values
  - Polls the data over SNMP
  - Sequentially scans all cluster members
- Selects the best candidates
  - Lowest positive value = best value
  - Other options possible as well
    - Round robin of alive nodes
- Updates the master DNS
  - Uses Dynamic DNS
  - With transactional signature keys (TSIG) authentication
- At most once per minute per cluster
- Active and Standby setup
  - Simple failover mechanism
  - Heartbeat file periodically fetched over HTTP
- Daemon is:
  - Written in Perl
  - Packaged in RPM
  - Configured by a Quattor NCM component
Application Cluster nodes

- SNMP daemon
  - Expects to receive a specific MIB OID
  - Passes control to an external program

- Load Balancing Metric
  - `/usr/local/bin/lbclient`
  - Examines the conditions of the running system
  - Computes a metric value

- Written in C
- Available as RPM
- Configured by a Quattor NCM component
Load Balancing Metric

- System checks – return Boolean value
  - Are daemons running (FTP, HTTP, SSH) ?
  - Is the node opened for users ?
  - Is there some space left on / tmp ?

- System state indicators
  - Return a (positive) number
  - Compose the metric formula
    - System CPU load
    - Number of unique users logged in
    - Swapping activity
    - Number of running X sessions

- Integration with monitoring
  - Decouple checking and reporting
  - Replace internal formula by a monitoring metric
  - Disadvantage – introduction of a delay

- Easily replaceable by another site specific binary
Operational examples

- LXPLUS – Interactive Login cluster
  - SSH protocol
  - Users log on to a server and interact with it

- CASTORGRID – GridFTP cluster
  - Specific application on a specific port
  - Experimented with live evaluation of the network traffic by the metric binary

- WWW servers

- Could be any application – client metric concept is sufficiently universal!
State Less vs. State Aware

- System is not aware of the state of connections
- **State Less Application**
  - For any connection, any server will do
  - Our system only keeps the list of available hosts up-to-date
  - Example: WWW server serving static content
- **State Aware Application**
  - Initial connection to a server; subsequent connection to the same server
  - Our load balancing system can not help here
  - Solution: after the initial connection *the application* must indicate to the client where to connect
    - Effective bypass of the load balancing
    - Example: `ServerName` directive in Apache daemon
LXPLUS statistics

Selection process - 2 weeks totals comparison

Number of times the node was part of the DNS alias (selected as one of the best candidates)

13 February
CHEP 2006, Mumbai, India
LXPLUS statistics

Metric value of 2 nodes - 2 weeks comparison

- LXPLUS071
- LXPLUS073
Conclusion

- Dynamic DNS switching offers possibility to implement automated and intelligent load-balancing and failover system
- Scalable
  - From two node cluster to complex application clusters
  - Decoupled from complexity of the network topology
- Need for an Arbiter
  - Monitor the cluster members
  - Select the best candidates
  - Update the published DNS records
- Built around OpenSource tools
- Easy to adopt anywhere
Thank you.

- [http://cern.ch/dns](http://cern.ch/dns)
  (accessible from inside CERN network only)

Vladimír Bahyl
[http://cern.ch/vlado](http://cern.ch/vlado)