HITACHI
Inspire the Next

Networking Best Practices

Use of a Load Balancer With Hitachi Content Platform and Hitachi Content Platform Anywhere

By Hitachi Data Systems

August 2015

Contents

Executive Summary

In the world of cloud and data mobility, there is an expectation that information can be accessed anytime, anywhere and from any device. While this expectancy was originally driven in the form of public cloud services, enterprise users quickly took advantage of these services as corporate IT struggled to keep up.

As many services are provided by many clustered entities (servers), each offering the same service in conjunction with its partners, random but reliable session establishment between client devices and the service providing entities is key.

Randomness is required to equally spread the load across all entities, distributing the load evenly for the best possible resource usage; reliability means that a single failing entity must not degrade or even circumvent service availability.

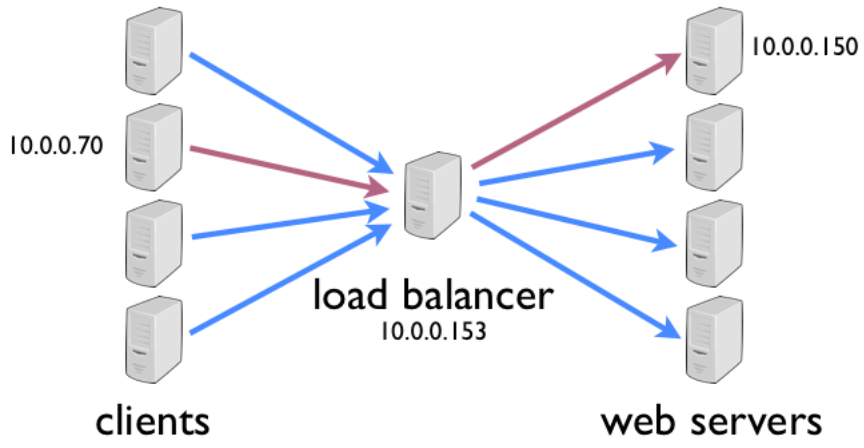
Up-to-date network design makes use of an active network device, a load balancer, to provide the claimed characteristics.

This white paper discusses the use of load balancers for Hitachi Content Platform (HCP) and Hitachi Content Platform Anywhere (HCP Anywhere).

Introduction

In computing, load balancing distributes workloads across multiple computing resources, such as computers, a computer cluster, network links, central processing units or disk drives. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource (see Figure 1). Using multiple components with load balancing instead of a single component may increase reliability through redundancy. Load balancing usually involves dedicated software or hardware, such as a multilayer switch or a domain name system (DNS) server process.[1]

Figure 1. Generic Scenario



HCP and/or HCP Anywhere greatly benefit from being used through a load balancer. This white paper discusses these benefits and the concepts to achieve them.

Within this white paper, all examples state to use port 443. This is true for data access to HCP through one of its http(s)/REST-based data access (native, HS3, HSwift) interfaces. For Management Console, Search Console or Management API (MAPI) access, replace 443 by 8000, 8888 or 9090.

[1] (wikipedia.org, 2015)

Load Balancer Concepts

A load balancer, often called local traffic manager (LTM):

- Is a device that (from a client's perspective) receives requests through a network interface (customer network interface), which it then forwards to one out of a selection of back-end servers of equal functionality. This process implies that it doesn't matter to which server the load balancer forwards a request, in terms of functionality.
- Is a device in the data path. Each request must to be processed by the load balancer, making it a single point of failure, if not clustered.
- Has a set of rules it applies to decide to which back-end server to forward a request.
- Maintains a dictionary of requests in transaction and a dictionary of known clients, likely with information to which back-end server it was forwarded recently.
- Maintains a list of usable back-end servers by constantly monitoring their availability.

Closely related to a load balancer is the concept of a global traffic manager (GTM), which:

- Is a kind of intelligent DNS server.
- Is not in the data path.
- Has a set of rules it applies to decide which back-end servers IP address to promote when answering a query.
- Maintains a list of usable back-end servers by constantly monitoring their availability.
- Answers clients DNS queries with IP addresses according to the implemented rules and back-end server availability.
- Is often used in multisite environments in addition to local load balancers to help keeping traffic local as long as possible, but allowing for automated evasion to other sites in case on site fails.

Hitachi Content Platform Anywhere

HCP Anywhere is a cluster consisting of two servers (nodes) with equal functionality.

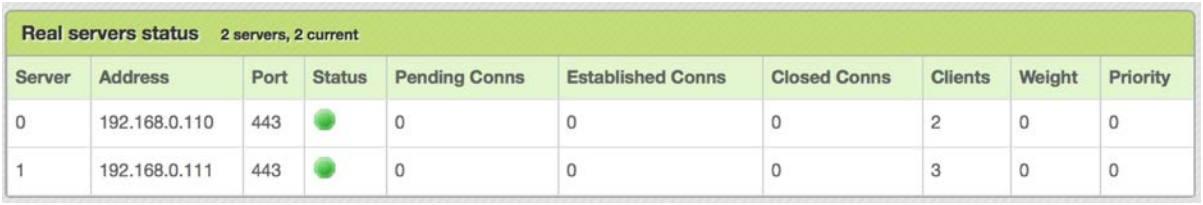
Clients use https/REST-based API calls to communicate with HCP Anywhere nodes. As each request to the API is atomic and all relevant information is carried within each single request, it doesn't matter to which of the two nodes a client's request is directed.

Standard Load Balancer Setup

- Create a server pool (See Figure 2.).
  - Setup a Transmission Control Protocol (TCP) based configuration [don't need the overhead of an http(s)].
  - Use both HCP Anywhere nodes as back-end servers.
  - Create a virtual network interface for the server pool and assign a static network address to it.
  - Configure the server pool rules to:
    - Be responsible for traffic received at port 443 (https).
    - Distribute traffic "round robin." (See Figure 3.)
    - Give both nodes an equal weight and priority.

- Make sure that client IP addresses are not made persistent to a specific back-end node.
- Check for accessibility using TCP at port 443 (typically built-in).
- Periodically check for service availability every few seconds.
  Do an HEAD request to https://hcpaw.dom.local:443/portal, expecting an http code of 302 as an indication that the service is available. Don't add an authorization header to that request: You'll want it to redirect you to the login page (indicated by the 302 code).

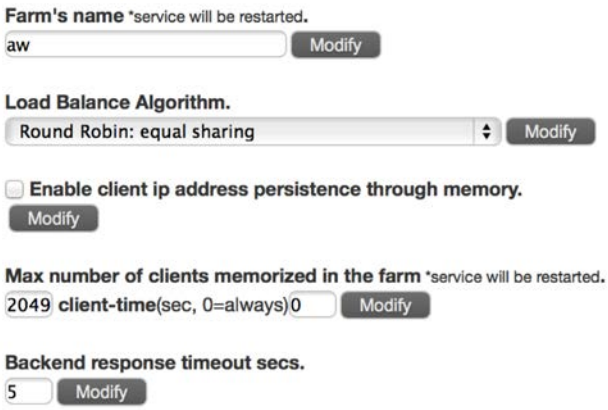
Figure 2. Example From zenLoadbalancer[2] Configuration



Real servers status 2 servers, 2 current

Server	Address	Port	Status	Pending Conns	Established Conns	Closed Conns	Clients	Weight	Priority
0	192.168.0.110	443		0	0	0	2	0	0
1	192.168.0.111	443		0	0	0	3	0	0

Figure 3. Round-Robin Example From zenLoadbalancer Configuration

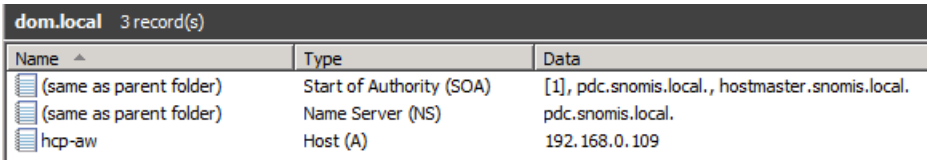


Configure DNS

In addition, it's required to configure DNS to answer with the server pool's virtual network IP address when queried for HCP Anywhere's Full Qualified Domain Name (FQDN). Simple addition of a record with the respective name and the virtual network IP address is sufficient (see Figure 4).

[2] (Sofitel IT Engineering, SL, 2014)

Figure 4. DNS Configuration for Hitachi Content Platform Anywhere



If HCP Anywhere shall be reachable from the Internet, external DNS needs to be configured to resolves to that IP, too.

Advanced Load Balancer Setup

Using an http(s)-based pool, more sophisticated configurations are possible.

Exercise: HCP Anywhere shall be available for desktop and mobile clients from the internal network, only. The only exceptions to this are public shared links (and folders), which shall be available from the Internet.

Solution:

On the load balancer, create two pools: one for internal access, the other for Internet access.

- For internal access, you can use the pool shown in the *Standard Load Balancer Setup* description above.
- For Internet access, create a separate pool:
  - Create a virtual network interface for the pool and assign a static network address to it.
  - Use ***https*** as the pool's base functionality.
  - Configure the pool's rules to:
    - Accept ***standard HTTP*** verbs (PUT, GET, HEAD), only.
    - Select the ***HTTPS*** listener.
    - Make sure that client IP addresses are ***not made persistent*** to a specific back-end node.
    - Allow access to virtual host ***<youranywhere.yourdomain.com>***, only.
    - Enable URL pattern matching, allowing the following patterns, only:
      - ***/mobile/links/public***
      - ***/mobile/links/js***
      - ***/portal/Bootstrap***
      - ***/portal/Bootstrap/dist/css***
      - ***/portal/css***
      - ***/portal/images***
      - ***/portal/js***
      - ***/u***

- Add your HCP Anywhere nodes as back-end servers to the pool.
- Ensure ***round robin*** for back-end server selection.

For DNS, there are different configurations for internal and external name resolution needed:

- Configure the internal DNS servers to resolve ***<youranywhere.yourdomain.com>*** with the internal pool's virtual IP address.
- Configure the external available DNS servers to resolve ***<youranywhere.yourdomain.com>*** with the Internet access pool's virtual IP address.

Make sure that the said virtual IP addresses are only available from the respective network (an external user should not be able to access the internal pool's virtual IP address!).

Hitachi Content Platform

HCP is a cluster consisting of at least four and up to 80 nodes, all serving client requests. Unlike HCP Anywhere, it needs to have a certain number of nodes up and running to provide full functionality. If the number of available nodes falls below of that, the cluster enters read-only mode, which a load balancer has to be concerned about.

Depending on whether HCP is standalone or is being replicated to a second HCP system, the setup of load balancers will vary. Several scenarios are described here.

Load balancing makes sense for:

- All http(s)-based traffic (data, MAPI and management console access).
- SMTP traffic.

Load Balance Single (standalone) HCP

Load balancing the traffic towards a single HCP is straightforward (see Figure 5):

- Create a server pool or farm for data access.
  - A TCP-based farm is fine [don't need the overhead of an http(s)-based configuration].
  - Contain all (!) HCP nodes as back-end servers.
  - Create a virtual network interface for the pool or farm and assign a static network address to it.
  - Configure the pool's rules to:
    - Be responsible for traffic received at port 443 (https).
    - Distribute traffic round robin (see Figure 6).
    - Give all nodes an equal weight and priority.
    - Make sure that client IP addresses are not made persistent to a specific back-end node.
      (This would lead a single client ending up using just a single node, even if making use of multiple connections in parallel. As using multiple connections is a proper way to gain performance or throughput, this is not what you want.)
    - Check for accessibility using TCP at port 443 (typically built in).

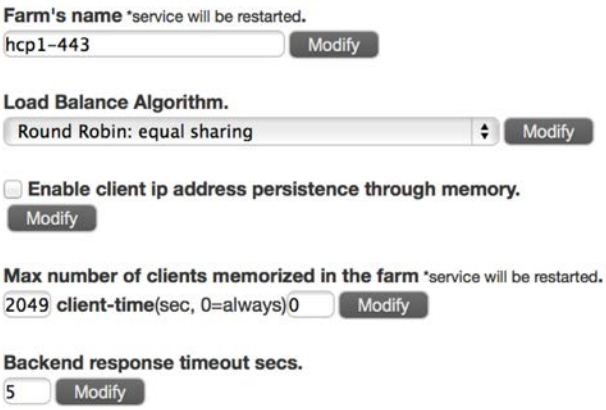
- Periodically check for service availability every few seconds:
  Do an HEAD request to https://<node IP address>:443/rest, expecting an http code of 302 as an indication that the service is available. You need to add an Host header to the request (host: namespace.tenant.<hcp_fqdn>), but must not add an authorization header to that request: You'll want it to redirect you to the login page (indicated by the 302 code).

Figure 5. Example of Load Balancing for HCP From zenLoadbalancer Configuration.

Real servers status 4 servers, 4 current

Server	Address	Port	Status	Pending Conns	Established Conns	Closed Conns	Clients	Weight	Priority
0	192.168.0.52	443		0	0	0	0	0	0
1	192.168.0.53	443		0	0	0	2	0	0
2	192.168.0.54	443		0	0	0	1	0	0
3	192.168.0.55	443		0	0	0	1	0	0

Figure 6. Example of Round Robin for HCP From zenLoadbalancer Configuration



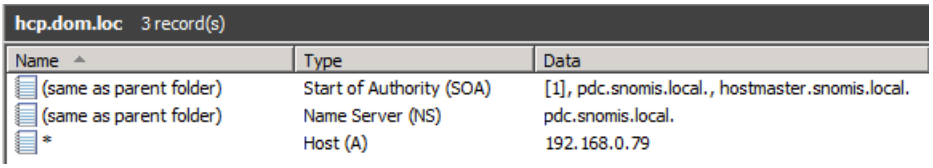
Configure DNS

The usual way to configure DNS for HCP is to set up a stub or >(better) a secondary zone for HCP's Full Qualified Domain Name (FQDN, hcp.dom.local in the examples). DNS will then respond with all node's IP addresses in round robin when queried for Tenants and Namespace.

As the load balancer takes care of using all nodes and needs to be the target for an application accessing HCP, a different approach is needed.

DNS has to be configured to respond with the pool's virtual networks IP address when queried for HCPs FQDN. A primary zone is required for the HCPs FQDN; simply adding a record with the respective name and the virtual network's IP address is sufficient. (See Figure 7.)

Figure 7. DNS Configuration for Hitachi Content Platform



hcp.dom.loc 3 record(s)

Name	Type	Data
(same as parent folder)	Start of Authority (SOA)	[1], pdc.snomis.local., hostmaster.snomis.local.
(same as parent folder)	Name Server (NS)	pdc.snomis.local.
*	Host (A)	192.168.0.79

Load Balance a Pair of Active-Active Replicated HCP Systems

Active-active replication (also called global access technology), if properly configured, allows access to both HCP systems participating in a replication link under the same name. That is, namespace.tenant.hcp1.dom.com is valid for both HCP systems. This allows for resource use on both HCP systems for active requests (compared to one HCP idling in an active-passive replication scenario). It also allows for switching over traffic to a single system, instead of initiating a failover in HCP. In certain situations, using active-active replication along with load balancers provides a higher availability for the overall environment.

Load balancing the traffic towards a active-active replication-enabled pair of HCPs adds a bit of complexity to the load balancers configuration (differences to single HCP configuration in red):

- Create a common server pool for data access:
  - A TCP-based farm is fine [don't need the overhead of an http(s)-based configuration].
  - Contain all (!) nodes of both (!) HCPs as back-end servers.
  - Create a virtual network interface for the pool and assign a static network address to it.
  - Configure the pool's rules to:
    - Be responsible for traffic received at port 443 (https).
    - Depending on the situation, distribute traffic:
      - Round robin, if both HCPs are comparable good accessible (bandwidth, latency);
        Give all nodes an equal weight and priority.
      - Prioritized, if one HCP has better network connectivity than the other. In this case, you want to keep traffic local as long as possible (that is, as long as the local HCP is alive and in read/write mode). For each HCP, give all nodes an equal weight and priority, but differ in priority between the two HCPs.
    - Disable all nodes belonging to a single HCP as soon as soon as this HCP enters read-only mode. This will make sure that applications don't run into failures while not having write access.
    - Make sure that client IP addresses are not made persistent to a specific back-end node.
    - Check for accessibility using TCP at port 443 (typically build-in).
    - Periodically check for service availability every some seconds:
      - *Do an HEAD request to https://<node IP address>:443/rest, expecting an http code of 302 as an indication that the service is available. You need to add a host header to the request (host: namespace.tenant.<hcp_fqdn>), but must not add an authorization header to that request. You'll want it to redirect you to the login page (indicated by the 302 code).*
      - *If you want to make sure that the load balancer is able to fail over the entire traffic to the partner HCP in case an HCP enters read-only mode, you need to adopt an approach like this:*

- *Create a separate Namespace (ex.: lb.it.<hcp_fqdn>) with minimal capacity (a few 10GB will do). DPL1, having the protocol in question enabled, along with versioning, set to one day. (This will allow the load balancer to write that file again and again while not over consuming storage in HCP.)*
- *Create a local user with read/write access rights to that Namespace.*
- *Do a PUT request to https://<node IP address>:443/rest/lbtest.txt, sending a minimal file (a few bytes), expecting an http code of 201 as an indication that the service is available.*
  You need to add a host header to the request (host: lb.it.<hcp_fqdn>) and an authorization header to that request. Read the ***Using a Namespace*** *manual on how to create those headers.*
- *You'll also have to add a rule to the load balancer's pool that recognizes that a failover is needed in case the nodes enter read-only mode.*

DNS Configuration

This configuration follows the same approach as that described in *Load Balancing for Single (standalone) HCP*.

Load Balancer Availability

A single load balancer in the data path to either HCP or HCP Anywhere is a single point of failure.

To prevent an instance in which HCP or HCP Anywhere become unavailable by a defective load balancer, the load balancers are typically deployed as clustered pairs. Configuration depends on vendor and model, but always leads to both of them sharing the pools virtual IP address.

Another deployment method is to use a global traffic manager to distribute traffic across all load balancers (local traffic managers) in charge for a pool. In this case, the GTM will monitor the availability of the LTMs and routes traffic by answering DNS queries accordingly.

Appendix A: zenLoadbalancer

zenLoadbalancer[3] is an open-source load balancer, available as Community and Enterprise Edition.

The Community Edition can be downloaded as an ISO image, which is easily installable on a virtual machine running on any hypervisor. Please see download and documentation section on the zenLoadbalancers website.

While configuration of zenLoadbalancer is mainly done through the provided webGUI, some tasks can become very annoying due to the need to enter the same information repeatedly. Knowing about the internal structure, it is possible to copy already existing configurations and adopt them to different needs.

All configuration files are located in the ***/usr/local/zenloadbalancer/config*** folder:

- ***global.conf*** – the base configuration file.
- ***<farm_name>_pen.cfg*** – configuration for TCP-based farms.
- ***<farm_name>_pound.cfg*** – configuration for HTTP(S)-based farms.
- ***<farm_name>_Err???.html*** – custom error messages per farm.
- ***if_eth?:?_conf*** – network interface configuration files.
- ***zlb-start***, ***zlb-stop*** – custom scripts that will be run after zenLoadbalancer has started / before it goes down

Log files can be found in ***/usr/local/zenloadbalancer/logs***.

[3] www.zenloadbalancer.com

Hitachi Data Systems

Corporate Headquarters
2845 Lafayette Street
Santa Clara, CA 95050-2639 USA
www.HDS.com **community.HDS.com**

Regional Contact Information
Americas: +1 866 374 5822 or info@hds.com
Europe, Middle East and Africa: +44 (0) 1753 618000 or info.emea@hds.com
Asia Pacific: +852 3189 7900 or hds.marketing.apac@hds.com

WP-538-B HDS September 2016