GSLB (Global Server Load Balancing) Functional specification and Design Document

Introduction

Global Server Load Balancing (GSLB) [1] represents set of technologies that is used for business continuance and disaster recovery. Depending on nature of deployment different goals are targeted, like

- disaster recovery: provide alternate location for accessing resource in the event of failure
- load sharing: distribute traffic across multiple locations
- performance: select a location based on client's geographic and network proximity or location that is least loaded etc

GSLB technologies rely on techniques like DNS redirection, HTTP redirection, Route Health Injection (RHI) etc provide above goals. Typically GSLB is configured in active-active data-center setup or active-standby setup. In active-active setup client requests are load balanced across active data centers. In active-standby setup, only on active data center failure, standby datacenter becomes operational. Application delivery controllers (ADC) that provide SLB (Server Load Balancing) also typically provide GSLB functionality as well.

At present CloudStack only supports SLB in the zone. For the public clouds and large enterprises that has more than one data centers it is valuable to provide functionality in CloudStack to orchestrate setting up GLSB. CloudStack already integrates with ADC's that provide load balancing with in a zone. Its logical to extend the functionality leveraging GSLB capabilities of ADC for disaster recovery solution. Also this feature will complement regions [2] as well.

Bug CLOUDSTACK-653 [5] has been opened to track this feature.

Purpose

Purpose of this document is present the functional requirements for supporting in GSLB in CloudStack and detail design aspects of how the functionality will be achieved.

Scope & Assumptions

- There are two parts to introducing GSLB feature in to CloudStack, like all the features of CloudStack.
  - First part is CloudStack framework/core changes like API, network element commands, network manager changes, schema changes etc that are generic (or abstracted) and irrespective of ADC that is providing GSLB functionality.
  - Second part is the changes to network element plug-ins to provide GSLB functionality. For this proposal only support of NetScaler ADC as GLSB provider is assumed. So only changes specific to NetScaler network element plug-in is in scope.
- DNS redirection is widely used technique to achieve GSLB. For this proposal it is assumed that 'DNS redirection' functionality will be used to achieve GSLB. So scope of this document is restricted to achieving GSLB functionality using DNS redirection.
- It is assumed that GSLB functionality CloudStack will only work with Active-Active data center setup.

References


Document History

<table>
<thead>
<tr>
<th>Version</th>
<th>Author</th>
<th>Date</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft</td>
<td>Murali Reddy</td>
<td>09-Jan 2012</td>
<td></td>
</tr>
</tbody>
</table>
Conventions

- ‘Admin’ persona refers to the administrator of CloudStack managed cloud deployment, who also has authority to configure the infrastructural components like domain name server.
- ‘User’ persona refers the consumer of cloud deployment and represents a tenant.
- ‘Word ‘cloud’ is used to represent entire deployment that can span multiple regions and multiple zone with in the region.
- ‘Virtual Server’ refers to the load balancing service with in a zone. A load balancer rule created in CloudStack would result in provisioning a ‘Virtual Server’. ‘Virtual Server’ will have a client facing public IP that is used to access the server. One or more VM's are bound to ‘Virtual Server’, traffic to virtual server gets load balanced across the VM's bound to virtual server.
- ‘GSLB virtual server’ is used to refer to the service launched by user across multiple zones, for which user uses CloudStack's GSLB functionality to load balance traffic across the VM's in multiple zones. One or more virtual servers from different zones are bound to ‘GSLB virtual server’. GSLB virtual server, unlike virtual server does not have a public IP associated with it, instead ‘GSLB virtual server’ will have FQDN DNS name.

Conceptual Model

It is important to understand the conceptual model of how GLSB functionality that shall be provided in CloudStack will work to understand the requirements and design detailed later. Here is example scenario to illustrate conceptual model. A xyztelco who has set up a public cloud that spans two zones zone 1 and zone 2 across geographically separated datacenters that are managed by CloudStack. Tenant A of the cloud wishes to launch a highly available solution out of the xyztelco's cloud. Tenant A, launches VM1, VM2 instances in zone 1 and VM5 and VM6 instances in zone 2. Tenant A acquires a public IP, IP1 in the zone 1, and sets up load balancer rule to load balance traffic between VM1, VM2 instances. CloudStack will orchestrate setting up a virtual server on LB service provider at zone1. Virtual server 1 that is setup on LB service provider in zone 1 represents a publicly accessible virtual server that client reach at IP1 address. Client traffic reaching virtual server 1 at IP1 will be load balanced across the VM1 and VM2 instances. Tenant A also acquires a public IP, IP2 in zone 2 and sets up a load balancer rule to load balance traffic between VM5, VM6 instances. CloudStack will orchestrate setting up a virtual server on LB service provider at zone2. Virtual server 2 that is setup on LB service provider in zone 2 represents a publicly accessible virtual server that client reach at IP2 address. Client traffic reaching virtual server 2 at IP2 will be load balanced across the VM5 and VM6 instances. At this point tenant A has service enabled in both the zones, but has no means to setup disaster recovery plan in case one of the zone fails. Also there is no way for tenant A to load balance the traffic intelligently to one of the zones based on load, proximity etc.

Cloud admin of xyztelco provisiones ‘GSLB service provider’ at both zones 1 and 2. A ‘GSLB provider’ is typically ADC's that has ability to act as ADNS (authoritative domain name server) server and has mechanism to monitor health of virtual servers both at local site and remote site. Cloud admin enables GSLB as a service to the tenants that use zones 1 and 2.
Tenant A, wishes to leverage the GSLB service provided by xyztelco's cloud. Tenant A configures GSLB rule to load balance traffic across the virtual server 1 at zone 1 and virtual server 3 at zone 2 and provides A.xyztelco.com as the domain name. CloudStack will orchestrate setting up GSLB virtual server 1 on GSLB service provider at zone 1. CloudStack will bind virtual server 1 at zone 1 and virtual server 3 at zone 2 to GSLB virtual server 1. GSLB virtual server 1 is configured to start monitoring the health of virtual server 1 and 3. CloudStack will also orchestrate setting up GSLB virtual server 3 on GSLB service provider at zone 2. CloudStack will bind virtual server 1 at zone 1 and virtual server 3 at zone 2 to GSLB virtual server 2. GSLB virtual server 2 is configured to start monitoring the health of virtual server 1 and 3. CloudStack will bind the domain A.xyztelco.com to both the GSLB virtual server 1 and 3. At this point tenant's A service is a globally reachable at A.xyztelco.com. Private DNS server for the domain xyztelcom.com is configured by admin (out-of-band) to resolve domain A.xyztelco.com to GSLB providers at both the zones which are configured as ADNS for the domain A.xyztelco.com. A client when sends a DNS request to resolve A.xyztelcom.com eventually gets DNS delegation to address of GSLB providers at zone 1 and 2. A client DNS request will be received by GSLB provider. GSLB provider, depending on the domain for which it needs to resolve, will pick the GSLB virtual server associated with the domain. Depending on the health of the virtual servers being load balanced, DNS request for the domain will be resolved to the public IP associated with selected virtual server.

### Functional Requirements

1. GSLB shall be added as a new network service provided by CloudStack.
2. GSLB service provider shall be able to be added to a physical network in the zone.
3. Admin shall be able to enable/disable GSLB functionality per region level.
4. Admin shall be able to configure a zone as GSLB capable/enabled.
5. A zone shall be considered as GSLB capable only if GSLB service provider is provisioned in to zone.
6. When users have VM's deployed in multiple availability zones which are GSLB enabled, user shall be able to use GSLB functionality to load balance traffic across the VM's in multiple zones.
7. Only when admin has enabled GSLB in the region, user shall be able to use GSLB to load balance across the VM's across the zones in that region
8. Users shall be able to load balance traffic across the availability zones in same region or different regions.
9. Admin shall be able to configure DNS name for the entire cloud.
10. User shall be able to specify an unique name (across the cloud) for the globally load balanced service. Provided name will be used as domain under the DNS name associated with the cloud.
11. User provided name along with Admin provided DNS name shall be used to produce a globally resolvable FQDN for the globally load balanced service of the user. For e.g. if Admin has configured xyztelco.com as DNS name for the cloud, and user specifies 'foo' for the GSLB virtual service, then foo.xyztelco.com would be the FQDN name of the GSLB virtual service.
12. While setting up GSLB, Users shall be able to select load balancing method (like round robin, least RTT etc) that would be used load balance traffic across the zones participating in GSLB.
13. User shall be able to set weight to zone level virtual server. Weight shall be considered by the load balancing method is distributing the traffic.
14. GSLB functionality shall support session persistence, where in series of client requests for particular domain name is sent to the same zone's virtual server.
Detailed Design

API changes

For providing GSLB functionality in CloudStack, it would have been much simpler if we can just relax current load balancing API (createLoadBalancerRule and assignToLoadBalancerRule) to be able to assign VM's across zones to load balancer rule. But current semantics of load balancer API's are tightly coupled with zone level resources. For e.g. createLoadBalancerRule API, takes the public IP, zone ID, network ID as parameters. Current set of API can not be easily enhanced to provide cross zone load balancing. Having a clear two-tier load balancing model seems natural fit to CloudStack. First tier of load balancing where GSLB will select (load balance traffic) a zone based on the configured LB method, persistence and health of service across the zones. Second tier would be SLB where traffic will be load balanced across the VM's in selected zone by GSLB in first tier. Semantics of GSLB and SLB have differences and warrants new set of API's for GSLB. Also having separate API's to manage the GSLB and SLB tiers separately will help in independent life cycle of load balancing rule.

Following new Async API's shall be introduced for creating and managing load balancing at GSLB tier.

- **createGlobalLoadBalancerRule**
  - name: name of the global load balancer rule
  - domain name: preferred domain name for the service.
  - lb algorithm: algorithm used to load balance the traffic across the zones. Round Robin, RTT, proximity
  - session persistence: source IP and HTTP cookie
  - account name
  - domain Id

- **assignToGlobalLoadBalancerRule**
  - id: UUID of global load balancer rule
  - Map of load balancer rule ID's and corresponding weight. List of the UUID's of load balancer rules that need to be associated with global load balancer. These are second tier load balancing rules created with createLoadBalancerRule API. Weight is optional, if not specified will be interpreted as 1

- **removeFromGlobalLoadBalancerRule**
  - id: UUID of global load balancer rule
  - load balancer rule ID's:

- **deleteGlobalLoadBalancerRule**
  - id: UUID of global load balancer rule

- **listGlobalLoadBalancerRule**
  - account name
  - domain id
  - id: UUID of the global load balancer rule

- **updateGlobalLoadBalancerRule**
  - id: UUID of global load balancer rule
  - name: name of the global load balancer rule
  - lb algorithm: algorithm used to load balance the traffic across the zones. Round Robin, RTT, proximity
  - session persistence: source IP and HTTP cookie

A new response object 'GlobalLoadBalancerResponse' shall be introduced which will have below details.

- name: name of the global load balancer rule
- domain name: domain name corresponding to global load balancer rule
- configured lb method
- configured session persistence method
- name of the account that owns the rule
- domain Id
- list of load balancer rule ID's assigned

Following API will be modified to support GSLB functionality.

- addNetworkServiceProvider: API shall be modified to accept NetScaler provider as GSLB service provider.

Service layer changes

GlobalLoadBalancingRulesService shall be added in to service layer as a new service. GlobalLoadBalancingRulesService shall provide following methods.

- **createGlobalLoadBalancerRule**
  - shall be called by createGlobalLoadBalancerRule API
  - shall ensure that domain name parameter specified in the createGlobalLoadBalancerRule API is unique and not used for other
GlobalLoadBalancingRules.
- shall ensure that LB method parameter specified in the createGlobalLoadBalancerRule API is one of Round Robin, RTT, proximity methods
- shall ensure that persistence method parameter specified in the createGlobalLoadBalancerRule API is one of source IP and HTTP cookie
- shall ensure that account and domain ID parameter specified in the createGlobalLoadBalancerRule API are valid
- shall result in persisting the global load balancer rule in global_load_balancing_rule table

assignToGlobalLoadBalancerRule
- shall be called by assignToGlobalLoadBalancerRule API
- shall validate id of global load balancer rule parameter passed in assignToGlobalLoadBalancerRule API is valid, and caller has authority to access/use the rule
- shall validate the list of load balancer rules ID's passed in assignToGlobalLoadBalancerRule API are valid and caller has authority to access/use all the rules.
- shall ensure the all load balancer rules corresponding to the ID's passed in passed in assignToGlobalLoadBalancerRule API are in active state
- shall persist the association between global load balancer rule and load balancer rule in 'global_load_balancer_lb_rule_map' table.
- shall invoke applyGlobalLoadBalancerRule on the GlobalLoadBalancingServiceProvider

removeFromGlobalLoadBalancerRule
- shall be called by removeFromGlobalLoadBalancerRule API
- shall validate id of global load balancer rule parameter passed in removeFromGlobalLoadBalancerRule API is valid, and caller has authority to access/use the rule
- shall validate the list of load balancer rules ID's passed in removeFromGlobalLoadBalancerRule API are valid and caller has authority to access/use all the rules.
- shall remove the association between global load balancer rule and load balancer rule in 'global_load_balancer_lb_rule_map' table.
- shall invoke applyGlobalLoadBalancerRule on the GlobalLoadBalancingServiceProvider

deleteGlobalLoadBalancerRule
- shall be called by deleteGlobalLoadBalancerRule API
- shall validate id of global load balancer rule parameter passed in deleteGlobalLoadBalancerRule API is valid, and caller has authority to access/use the rule
- shall obtain the list of load balancer rules associated (assigned to) with global load balancer rule
- shall obtain the list of zones corresponding to the all load balancer rules obtained in previous step
- shall create and send 'GlobalLoadBalancerConfigCommand' network element command to all the GSLB service provider configured for each of the zone obtained in previous step
- shall delete the global load balancer rule in global_load_balancing_rule table and corresponding association mapping present in 'global_load_balancer_lb_rule_map'

searchGlobalLoadBalancerRule
- shall be called by listGlobalLoadBalancerRule API
- shall return the GlobalLoadBalancerResponse object

updateGlobalLoadBalancerRule
- shall be called by updateGlobalLoadBalancerRule API
- shall validate id of global load balancer rule parameter passed in updateGlobalLoadBalancerRule API is valid, and caller has authority to access/use the rule
- shall update the name, lb method, persistence method for the record in global_load_balancing_rule table corresponding to the global load balancer rule

GlobalLoadBalancingServiceProvider

Notion of region level services and service provider shall be introduced.

A new service ‘GSLB’ shall be introduced as a region level service. Notion of GSLB service provider will be introduced which represents a service provider that will provider GSLB service and shall provide below methods

- applyGlobalLoadBalancerRules

Agent commands


- domain name
- load balancing method used for GSLB
- persistence method used for GSLB
• flag indicating to add/remove the global load balancer
• list of virtual servers details
  • virtual IP and port tuple
  • associated weight with virtual server
  • flag indicating virtual server is being associated/dis-associated with global load balancer rule
  • flag indicating virtual server is local or remote

NetScaler plugin changes

NetScaler GSLB service provider

A GSLB service provider specific to NetScaler shall be implemented which will be part of NetScaler plug-in. NetScaler GSLB provider shall extend ‘GlobalLoadBalancingServiceProvider’. Hence shall implement applyGlobalLoadBalancerRules() method. Following actions shall be taken in applyGlobalLoadBalancerRules implementation.

• get the details of global load balancer rule
• get the details of load balancer rule associated with global load balancer rule
• shall obtain the list of zones corresponding to all the load balancer rules obtained in previous step
• shall create and send GlobalLoadBalancerConfigCommand network element command to all the GSLB service provider configured for each of the zone obtained in previous step

NetScaler Server Resource changes

Please refer to [3] for the configuration steps required to setup GSLB on NetScaler.

NetScaler server resource shall be extended to support execution of GlobalLoadBalancerConfigCommand network element command. Following actions shall be taken while executing GlobalLoadBalancerConfigCommand command. In general the implementation should be idempotent i.e.) executing GlobalLoadBalancerConfigCommand with same details should succeed.

• create a GSLB site with site name formed from the domain name details present in GlobalLoadBalancerConfigCommand
• create GSLB virtual server
• enable GSLB virtual server created in previous step
• create a GSLB service for each of the virtual server details present in GlobalLoadBalancerConfigCommand
• enable GSLB service for all the services created in earlier step
• bind the GSLB services to GSLB virtual server
• bind domain name to GSLB virtual server. Domain name is obtained from the domain details in GlobalLoadBalancerConfigCommand.

Global Config change

A global param 'domain.name' shall be added in to global configuration. This parameter specifies the DNS name for the cloud deployment.

Schema

Following two tables shall be added for supporting GSLB

• global_load_balancing_rules
  • id
  • LB method
  • persistence method
  • name
  • domain-name
  • account ID
  • domain ID

• global_load_balancer_lb_rule_map
  • Id of the global_load_balancing_rule
  • Id of the load balancer rule from the load_balancer_rules table

UI

- On the ‘Regions’ page a new ‘Services’ button shall be available.
- On clicking the services, dialog box specific to GSLB configuration shall come up

- Dialogue box for GSLB, should have the ability to
  
  - add a GSLB rule: dialog box to add a new GSLB rule shall have following details. createGlobalLoadBalancerRule API shall be called on clicking 'Add' option.
    - name of the gslb rule
    - description
    - region id
    - account name (optional)
    - domain id (optional)
    - gslb load balancer method: roundrobin, leastconn, proximity
    - persistence method: sourceip
    - domain name
    - service type(tcp,udp)
  
  - delete a GSLB rule; shall call deleteGlobalLoadBalancerRule API. Id's owned by the called shall be obtained from list Gslb rules API
    - id: UUID of the global load balancer rule
  
  - list GSLB rule: shall call listGlobalLoadBalancerRules() api
    - id: UUID of the global load balancer rule
    - region id: id of the region in which gslb rules to be listed
  
  - assign a load balancer rule to GSLB rule; shall call assignToGlobalLoadBalancerRule() API call
    - id: UUID of the global load balancer rule
  
  - remove a load balancer rule from GSLB rule
    - id: UUID of the global load balancer rule
  
- add NetScaler device dialogue box should have following additional parameters
  
  - boolean flag: indicating if the NetScaler device added is GSLB service provider in the zone
  - public IP: site public IP corresponding to NetScaler
  - private IP: site private IP corresponding to NetScaler

new parameters be passed in to addNetscalerLoadBalancer() API command

Open Issues

- Currently CloudStack does not have any operations that require orchestration of services across the zones. It seems logical to implement such cross zones services on top of orchestration engine at service layer. I plan to introduce notion of services and service providers in region.