

Global Server Load Balancing

Version 1.0.0

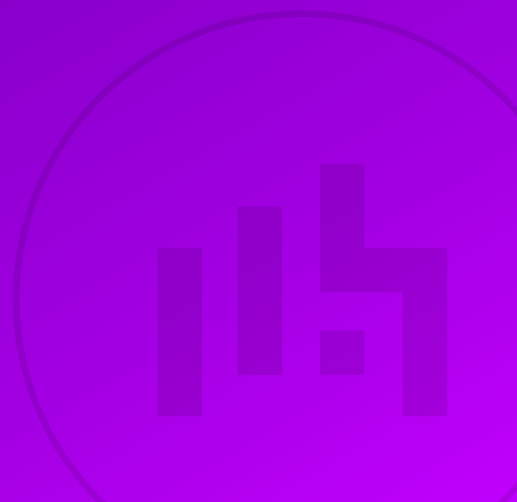


Table of Contents

1. About this Document	4
2. Understanding GSLB	4
2.1. Introduction	4
2.2. Core Components	6
2.2.1. Appliance Components	6
PowerDNS	7
Polaris	7
2.2.2. Environment Components	7
DNS Infrastructure	7
2.3. Features	7
2.3.1. Load Distribution	7
fogroup - Failover Group	7
wrr - Weighted Round Robin	8
twrr - Topology Weighted Round Robin	8
2.3.2. Health Checking	8
HTTP	8
TCP	9
Forced	9
External	9
External Dynamic Weight	10
2.3.3. SDNS (Smart DNS)	11
2.3.4. EDNS-CS	11
2.4. Use cases	12
2.4.1. Business Continuity and Disaster Recovery	12
2.4.2. Content Localization	12
2.4.3. Compliance	12
2.4.4. Data Sovereignty	12
2.4.5. Scale-out NAS	13
2.5. Emerging Trends	13
2.6. Conclusion	14
3. Configuration	14
3.1. Global Names	14
3.2. Members	16
3.2.1. Member IPs	18
3.3. Pools	19
3.3.1. Advanced Configuration	20
Monitor Interval	20
Monitor Timeout	20
Monitor Retries	21
Fallback	21
3.4. Topologies	21
3.5. DNS Setup	26
3.5.1. Non-Windows Servers	27
3.5.2. Windows Servers	27
4. Troubleshooting	28
4.1. GSLB Diagnostics	28
4.2. EDNS-CS	30
5. More Information	32

1. About this Document

This document provides a thorough introduction to Global Server Load Balancing (GSLB). Although it relates specifically to Loadbalancer.org appliances, it can also serve as a useful subject reference irrespective of appliance vendor.

The aim of the document is to provide the knowledge required to successfully and correctly implement GSLB using Loadbalancer.org appliances.

The document is broken down into the following three main sections:

1. Understanding GSLB
2. Configuring GSLB
3. Troubleshooting GSLB

2. Understanding GSLB

2.1. Introduction

Before looking at the configuration of GSLB we must first understand what it is and how it works. First, let's consider the differences between GSLB and local load balancing, also known as Server Load Balancing or SLB.

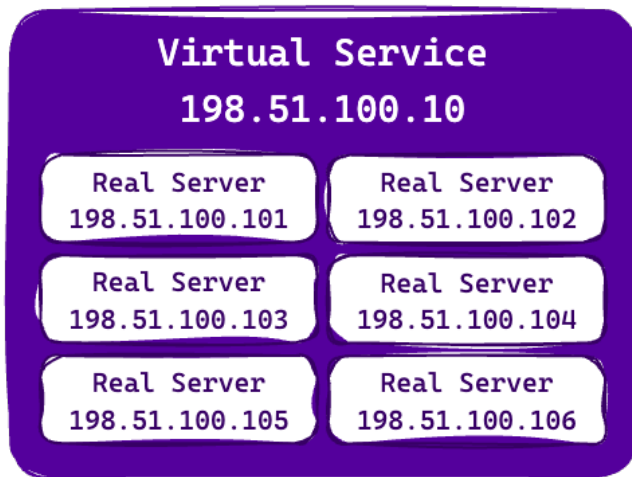
The need for local load balancing arises when we have an application or service which has a demand placed upon it that cannot be provided by a single server. Another server is added to perform the same function, effectively doubling the number of requests that can be handled; it is at this point that you should also add a load balancer to the solution. The reason for this is that as soon as you add a secondary server and create a pool of devices providing your service, you introduce a problem. That problem is "how should requests be distributed between the servers?". There are myriad ways to solve this issue, but the best and simplest is to use a load balancer.

The need for **Global Server Load Balancing** (GSLB) can arise from a myriad of reasons including, but not limited to:

- **Reduced Latency** by sending requests to the closest available servers
- **Additional redundancy** of a workload that is replicated between multiple data centres
- **Data Sovereignty** for information that must be kept within certain localities and/or geographical boundaries

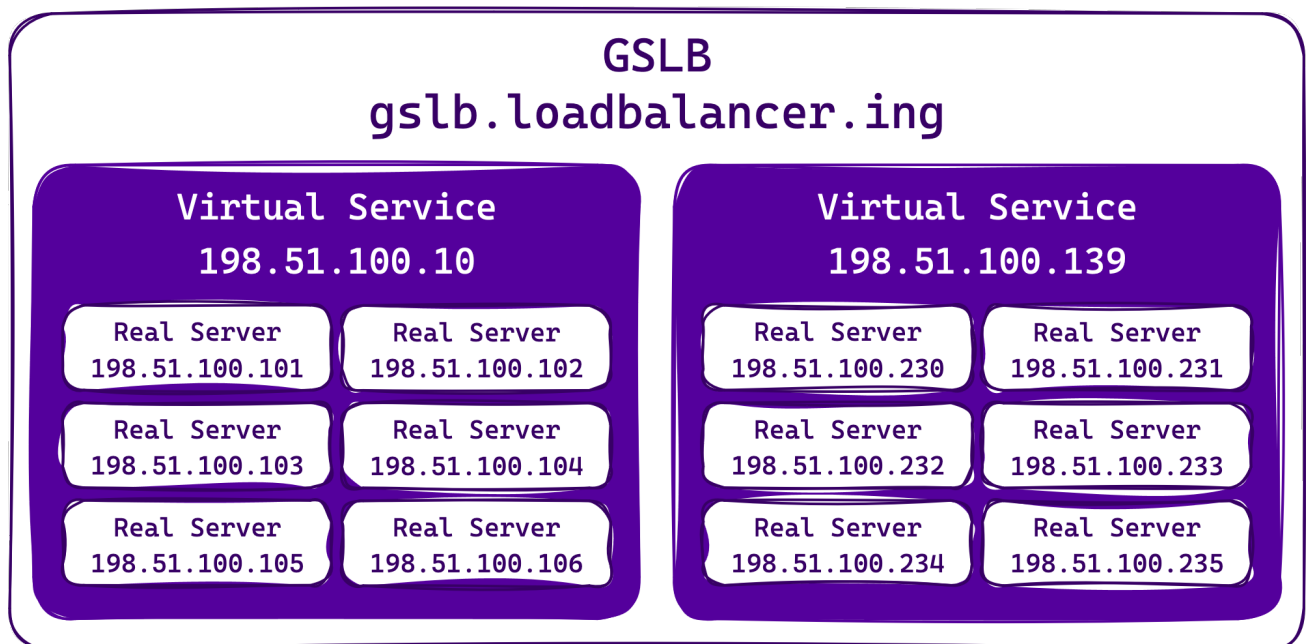
The following graphical representation shows the key elements involved in local load balancing.





With local load balancing there is a Virtual Service with a corresponding Virtual IP address (VIP) that receives incoming requests and a Pool of Real Servers to distribute the requests to.

The diagram can be expanded to show the key elements involved in GSLB.



With GSLB there are multiple sites each with a Virtual Service and associated Real Servers. Each site functions the same as with local load balancing. The difference with GSLB is that the initial connection point has moved upwards. GSLB is configured with a Global Name or FQDN (e.g. **gslb.loadbalancer.ing**) that resolves to one of the VIPs depending on how GSLB is configured. The client then connects to the selected VIP and is load balanced to one of the associated Real Servers in the normal way.

What this diagram shows is that GSLB is load balancing Virtual Services. That is, it is load balancing load balancers.

Note

It's also possible to route traffic directly to the Real Servers (aka Direct-to-Node or more recently Smart DNS) although routing traffic to another load balancer for local server load balancing is the most common approach.

Local Load Balancing - how it works

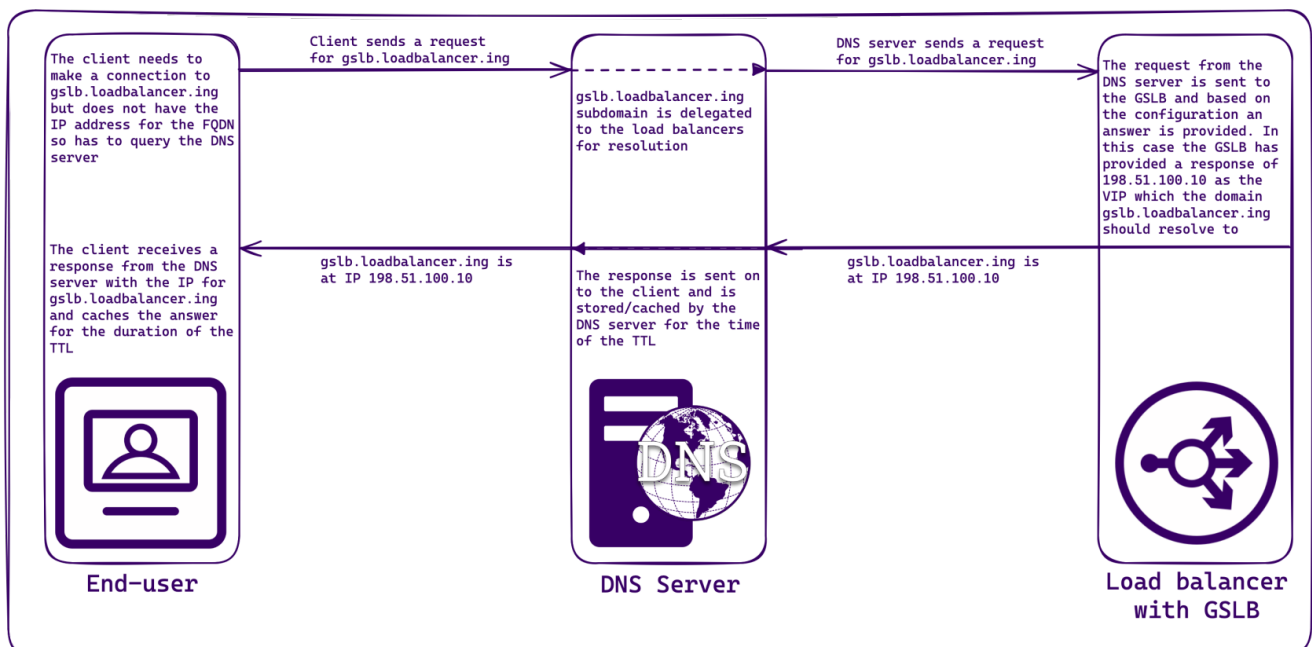
This process has been simplified to illustrate the differences between local load balancing and GSLB.

1. A client sends a connection to the VIP.
2. The virtual service evaluates the request and sends it to a Real Server.

Global Server Load Balancing - how it works

With GSLB, a number of DNS related processes must be performed first before the traffic can be passed to the correct destination. Please also refer to the diagram below.

1. The client needs to make a connection to **gslb.loadbalancer.ing** but does not have the IP address for the FQDN so has to query the DNS server.
2. Client sends a request for **gslb.loadbalancer.ing**.
2. The **gslb.loadbalancer.ing** subdomain is delegated to the load balancer(s) for resolution.
3. The DNS server sends a request for **gslb.loadbalancer.ing**.
4. The request from the DNS server is sent to the GSLB and based on the configuration, an answer is provided. In this case, the GSLB has provided a response of 198.51.100.10 as the VIP which the domain **gslb.loadbalancer.ing** should resolve to.
5. **gslb.loadbalancer.ing** is at IP 198.51.100.10.
6. The response is sent on to the client and is stored/cached by the DNS server for the time of the TTL.
7. **gslb.loadbalancer.ing** is at IP 198.51.100.10.
8. The client receives a response from the DNS server with the address for **gslb.loadbalancer.ing** and caches the answer for the duration of the TTL.



2.2. Core Components

2.2.1. Appliance Components

GSLB is built on and with open-source software. There are two core components: **PowerDNS** and **Polaris**.



PowerDNS is an extremely robust and well-supported software project. Polaris is an open source project which is currently principally maintained by Loadbalancer.org.

PowerDNS

PowerDNS is an open-source DNS server software designed to provide a robust, scalable, and versatile platform for routing internet traffic. It is widely used for its high performance, ease of management, and flexibility, supporting various back-ends such as BIND, MySQL, PostgreSQL, and Oracle. PowerDNS features a strong security model and is capable of handling a vast number of queries per second, making it suitable for deployment in high-traffic environments. It is also known for its advanced features like DNSSEC for securing DNS lookups and a versatile API for automation and integration with other tools and systems.

- [PowerDNS Website](#)
- [PowerDNS Documentation](#)

Polaris

The Polaris GSLB project provides an open-source GSLB solution that can be extended for high availability across data centres and beyond. Polaris enhances the capabilities of PowerDNS Authoritative Server with features such as load-balancing methods, customizable health monitors, dynamic server weighting. It supports running health checks against different IPs from the Member IP allowing a single Health Tracker to serve multiple DNS resolvers. Its setup is designed to be elastic and asynchronous, providing non-blocking communication between its internal components.

- [Polaris GitHub Project](#)
- [Polaris Documentation](#)

2.2.2. Environment Components

DNS Infrastructure

GSLB relies heavily on DNS. However, the appliance cannot take control of DNS requests and responses on its own. The network needs to be configured to pass DNS requests for the appropriate subdomains to the GSLB on the load balancer appliances. Whilst it is not **technically** a core component of the GSLB, it is a key part of the process, and needs to be understood for a successful implementation.

On Operating Systems other than Windows you're able to configure clients to direct themselves to particular DNS servers for certain subdomains. If you don't have any Windows systems in your environment, you don't necessarily have to make changes to the DNS infrastructure. However, we have not come across an environment where modification of the internal DNS infrastructure (domain delegation) is not required.

2.3. Features

2.3.1. Load Distribution

fogroup - Failover Group

The Failover Group load balancing method is a chained failover. That is, the Pool Members are configured in an order of preference. If a Member is marked as down due to a health check failure, the GSLB will use the next highest priority Member of the Pool. For example, assume there are three Members in a Pool that is configured with the **LB Method** set to **fogroup**. Connections will be sent to Member 1 at all times as it is the first configured Member in the Pool. If Member 1 fails then connections are sent to Member 2 as the second configured Member



in the Pool. If Member 2 also fails then all the requests will then be sent to Member 3. Now, if Member 1 were to come back online (Member 2 is still down) then all connections would be sent to Member 1 again.

wrr - Weighted Round Robin

When Weighted Round Robin is employed as the **LB Method** for a Pool, all the requests will be distributed amongst the Pool Members. The distribution of requests is performed in accordance with the weighting that each Member is given. The weight of the Member directly relates to the ratio of requests that should be sent to it relative to the weight of the other Pool Members. It is worth noting that the distribution of requests with regard to the weighting is eventually consistent as requests are sent in batches of three. For example, in a Pool configured with 3 Members and a **LB Method** of **wrr** the distribution of requests to Members of equal weight would not be 1, 2, 3, 1, 2, 3. It would actually be 1, 1, 1, 2, 2, 2, 3, 3, 3. Modifying the weight of the Members, understandably, introduces even more variance.

twrr - Topology Weighted Round Robin

Topology Weighted Round Robin, is the same as **wrr** but with an additional piece of evaluation. Before the requests are distributed based on the weighting of the Pool they are evaluated against a topology table. Topology tables are configured in the system and are used to associate Members with particular IP addresses or ranges. This feature allows administrators to preferentially set the destination for a request based on where the request is coming from. When a DNS request is received by the GSLB, one of the pieces of information that is available to it is the IP or subnet of where the request came from.

Let's say you have a data centre in London (10.100.100.10) and a data centre in Madrid (10.200.200.10). The users are also based in the UK (10.100.10.0/24) and in Spain (10.200.20.0/24). Either of the data centres is able to service requests from any user and GSLB is being used to increase redundancy and availability. During normal working, it makes sense for requests to be handled by the data centre which is geographically closest to the requester. This can be achieved by configuring a topology named "Spain" that contains the IPs 10.200.200.20 and 10.200.20.0/24. The topology tells the GSLB that requests which are sourced from 10.200.20.0/24 should be sent to the Member 10.200.200.20 as they are linked in the topology table. If the Madrid data centre Member was failing health checks then the request would be sent to another valid Member in the topology table such as the London data centre. In the case that there are no available Members the GSLB would revert to the fallback method which can be either **refuse** where the DNS request is simply refused or **any** where a random failing Member is sent as a response.

2.3.2. Health Checking

HTTP

HTTP health checks allow the GSLB to make an HTTP GET to the Member IPs (or their configured monitor IP, if set) to query their status. There configurable options for this health check are listed below. The HTTP GET that is made is expecting an HTTP code response which will be used to evaluate whether the health check has passed.

Option	Description
Monitor Use SSL	Whether to use SSL, default is No (false).
Monitor Hostname	Hostname to supply in HTTP Host: header, when using SSL this will also be supplied in SNI, default is "none".
Monitor URL Path	A url path to request, appended after the Member's IP address, default is "/".

Option	Description
Monitor Port	An integer between 1 and 65535, if value is not provided, port 80 will be used with use_ssl set to false, port 443 will be used with use_ssl set to true.
Monitor Expected Codes	An array of HTTP codes to match in a response for example "200", "301". Input range is between 100 and 599.

TCP

TCP health checks perform a TCP connect against the Member IPs (or their configured monitor IP, if set) to check if the GSLB is able to connect to the port of the Member. If the port is open/responds then the health check passes. You can optionally provide a send string and an expected response to that string to evaluate the status of the health check.

Option	Description
Monitor Port	An integer between 1 and 65535, if value is not provided, port 80 will be used with use_ssl set to false, port 443 will be used with use_ssl set to true.
Monitor Send String	A string to send after connecting to a socket, for example check .
Monitor Match Return	A reg exp to match in response, for example up (the reg exp compiles with re.IGNORECASE flag set).

Forced

Forced health checks will always evaluate as passing or failing dependent on the configuration that has been applied.

Option	Description
Monitor Status	A string, either up or down , default is up .

External

The power and the possibilities of health checking are really unleashed with the external monitor. External monitors are health check scripts that can be created by the administrator to perform any logic that can be programmed in to the GSLB. For example, you can create a script that checks a URL that requires authentication as per the example below.

This example is the default script that is shipped with Loadbalancer.org appliances and provides a useful starting point for users to create their own custom health checks.

```
#!/bin/bash
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin:/root/

# Simple URL Health Check

# Default variables passed by GSLB.
# $1 = Pool Member Address
# $2 = Monitor Port
# $3 and onwards provided by monitor parameters from GSLB Pools
```



```

# Set some easier to use variables
_POOL_MEMBER=$1
_PORT=$2
_HTTPTYPE=$3      # http or https
_HEALTHSCRIPT=$4   # Resource to check if site is available
_HTTPREQ=$5        # HTTP method (GET HEAD POST PUT DELETE CONNECT OPTIONS TRACE PATH)
_HTTPEXPRESCODE=$6 # Expected http status code
_USERPASS=$7       # username:password
_TIMEOUT=$8        # CURL timeout in seconds

# Check to see if a username/password has been set as an argument, if true add option "-u
username/password to the request"
if [[ "$_USERPASS" != "" ]]
then
    _USER="-u $_USERPASS "
else
    _USER=""
fi

# Check if _TIMEOUT is set, if not default to 5 seconds
if [[ "$_TIMEOUT" != "" ]]
then
    $_TIME=$_TIMEOUT"
else
    $_TIME="5"
fi

# Perform the test to get the http response code
_HTTPRESPCODE=$(curl -m $_TIME $_USER -X $_HTTPREQ --write-out '%{http_code}' -s -o /dev/null
${_HTTPTYPE}://${_POOL_MEMBER}:${_PORT}/${_HEALTHSCRIPT} -k)

# Test if the returned response code matches the expected response code
if [[ "$_HTTPRESPCODE" -eq "$_HTTPEXPRESCODE" ]]
then
    # Set to what is defined in monitor result
    echo "success"
else
    echo "failure"
fi

exit

```

External Dynamic Weight

External Dynamic Weight health checks take the power of External health checks and then also provide the ability to update the weight of the Members based on the results of those checks. The weights are changed dynamically.

As an example, a storage vendor could expose a metric based on disk, network, and CPU utilization of their nodes and make that information available to be queried by the GSLB external health check. Based on that metric the health check can compute and update the weighting that is set on that Member. Members with low utilization can have their priority raised, reducing the amount of work that high utilization Members are having to do. This allows you to proactively flatten the distribution curve for the Members providing a better user experience.

```

#!/bin/bash
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin:/root/

```



```

_DEBUG=0
_LOG=$(basename ${0}).log
_MEMBER_IP=$1
_MEMBER_PORT=$2

_WEIGHT="$(curl -X GET http://${_MEMBER_IP}:${_MEMBER_PORT}/healthcheck.php)"

if [ ${_DEBUG} -eq "1" ]; then
    echo "$(date) - ${_MEMBER_IP}:${_MEMBER_PORT} - weight is ${_WEIGHT}" >> /tmp/${_LOG}
    echo "$(date) - ${_MEMBER_IP}:${_MEMBER_PORT} - All variables: $" >> /tmp/${_LOG}
fi

echo $_WEIGHT

exit ${?}

```

The above bash script creates a health check that queries the Members at a URL. That URL returns a response which is a number between 0 and 10 each time the health check is run. That response is then used to set the weight of the server. In this case, the complexity of calculating the weight is left to the server and the GSLB is simply querying the server for what weight it should have based on system utilization.

2.3.3. SDNS (Smart DNS)

SDNS (Smart DNS) is a way of implementing GSLB in a specific way to leverage the benefits of round-robin DNS whilst dropping some of the biggest problems with it. Typical GSLB implementations on Loadbalancer.org appliances are configured with the VIPs of Virtual Services. This allows for the load balancer to handle the load balancing and sending the connections on to the Real Servers. This method allows you to keep all the benefits of local load balancing whilst also extending the deployment globally via a single namespace. Local load balancing provides features such as session persistence and greater control of connection tracking and distribution that GSLB cannot. With all that being said, there are some implementations that require Member IPs to be those of the Real Servers instead of the VIPs of the Virtual Services. In those instances we implement SDNS (Smart DNS) and the GSLB Members are the servers providing the service, the GSLB is sending those requests direct to the node.

The primary use case for this method, up until this point, has been where the load balancer could **potentially** be the bottleneck for network traffic. When the GSLB is configured to distribute requests through the VIPs, the load balancer is still in the path of the traffic as the GSLB is responding with a VIP that will direct the traffic through the load balancer. With SDNS, the GSLB will respond with the IP address for the Member/Real Server/node IP of the ultimate destination. As that traffic is not being directed to a VIP hosted on the load balancer, that traffic will not flow through the load balancer but around the load balancer. This means that the GSLB is able to direct traffic to the required destination not at the speed of the load balancer network interfaces but at the speed of the aggregate network connectivity between source and destination. In reality, most data centres are not connected at greater than 100Gbps.

2.3.4. EDNS-CS

EDNS-CS stands for Extension Mechanisms for DNS - Client Subnet. This is an extension to the DNS protocol that allows a DNS resolver to specify the network subnet for the client making the request, as part of the DNS query. This is particularly useful in the context of GSLB and Content Delivery Networks (CDNs), where it's important to route user requests to the geographically closest or most appropriate server.

EDNS-CS aims to improve the routing accuracy by providing more specific information about the client's location.



Normally, the DNS resolver receives the query and passes it on to the authoritative DNS server without any indication of where the original client is located. With EDNS-CS, the resolver can include part of the client's IP address in the request, enabling the authoritative server to make a more informed decision on how to route the client to the most appropriate endpoint.

However, it should be noted that the use of EDNS-CS has been subject to some privacy concerns, as it potentially exposes more information about the client to the authoritative DNS server. In summary, EDNS-CS is designed to optimize content delivery and server selection by including client subnet information in DNS queries.

2.4. Use cases

In this section of the course, we'll explore the practical side of GSLB. GSLB isn't just a technical concept; it's a key player in keeping services running smoothly, regardless of geographic boundaries. We'll look at how it helps in scenarios like improving performance globally and ensuring service continuity during server outages. By understanding these real-life applications, we'll see how GSLB is more than just a theory – it's an essential component in modern digital infrastructure, making our online interactions seamless and reliable.

2.4.1. Business Continuity and Disaster Recovery

Adding load balancing adds resilience, and you can actually add redundancy with Loadbalancer.org appliances. Adding redundancy with the Loadbalancer.org appliance works if the load balancer itself is still functional. Disaster recovery is another thing altogether though. In a disaster scenario, it is safe to assume that the entire data centre is unavailable. GSLB serves as a pivotal technology in disaster recovery strategies, as it provides a geographically distributed approach to directing traffic across multiple data centres. In the event of a local outage, GSLB can automatically reroute end-users to the nearest operational site with minimal disruption. This not only ensures continuous availability and business continuity but also helps to balance the load effectively during peak traffic times, thus maintaining performance standards. By using GSLB, organizations can protect their operations from regional failures, effectively manage traffic during planned maintenance, and distribute client requests according to customized policies that align with their specific disaster recovery objectives.

2.4.2. Content Localization

GSLB provides a way to perform content localization, which is a feature that enhances the end user experience by directing traffic to a preferred location. In the Loadbalancer.org appliances this is achieved with topologies and the **twrr** balancing method. This localization is crucial for delivering content rapidly and reducing latency. This capability allows businesses to offer a tailored experience, with faster loading times.

2.4.3. Compliance

For organizations needing to comply with regional data protection regulations, GSLB can be an important piece of the solution. Users can be directed to the appropriate data centre based on their geographic location, ensuring that data storage and processing adheres to local compliance mandates, such as the GDPR in Europe or CCPA in California. "Geo-awareness" is critical for multinational companies that must navigate a complex web of international laws. GSLB can help automate the process of routing traffic to comply with each jurisdiction's unique data residency requirements, thereby reducing the risk of costly legal penalties and enhancing customer trust.

2.4.4. Data Sovereignty

GSLB can play a vital role in maintaining data sovereignty, ensuring that digital information is stored and processed within the legal boundaries of a specific country or region. GSLB facilitates this by routing user



requests to data centres located within the appropriate geographical limits. Using this functionality, organizations can guarantee that data residency and sovereignty policies are upheld, which is particularly significant for entities operating across multiple jurisdictions with stringent data protection laws. This strategic routing minimizes legal risks and aligns with national regulations, providing a robust framework for global data management.

2.4.5. Scale-out NAS

One of the specific use cases that we are able to speak on with some authority is that of scale-out NAS. We are partnered with a number of storage providers, most notably with [Cloudian](#) through an OEM load balancer called [HyperBalance](#). The GSLB is deployed widely with object storage providers. The clusters that they deploy are usually replicated between data centres. As the data matches between all the locations all the GSLB options and methods are available. Usually, the installation method is the typical method of GSLB implementation and topology weighted round-robin is applied so that users are directed to a preferred data centre. This implementation allows the administrators to set which users/subnets should go to which data centre in the first instance. If the nearest data centre is failing health checks then the requester will be transparently directed to another data centre within the cluster.

In summary, GSLB offers a powerful way to manage global traffic, improve application performance, and increase availability. It is a critical component in any organization's multi-data centre strategy.

2.5. Emerging Trends

As we look to the future, we consider the features that could be required to support the evolving landscape of network infrastructure and where GSLB fits into it. The greater connectivity brought on by hyper-scale cloud providers and the internet as a whole is enabling the widespread implementation of GSLB. Once disparate systems and data centres are now connected to each other with high enough throughput and low enough latency for them to act as a single entity. As the interest in GSLB grows so do the requirements and expectations. With that in mind, we consider the direction and trends of the industry as a whole and what the future of GSLB might look like. Some of these trends are in early adoption whilst some have not been implemented or even developed yet.

- **Health-Based Load Balancing:** Advanced health checks and monitoring are being integrated into GSLB solutions, ensuring traffic is not just distributed based on geographic or static rules but also on the real-time health and performance of servers and applications. Whilst this is already a feature of the [Loadbalancer.org](#) GSLB it is not widely utilized; though there is a growing interest in this particular feature.
- **Geo-Location Based Routing:** Enhanced geo-location capabilities for directing users to the nearest or most suitable data centre based on their geographic location. This not only improves user experience by reducing latency but also helps in adhering to data sovereignty laws. [Loadbalancer.org](#) already provide this functionality with **twrr** based on topology tables. The logical next step for this is to automate the population of topology tables. This could be done by picking a result based on latency or by matching the source and potential destinations against geographic IP allocation information.
- **Cloud Integration and Hybrid Deployments:** As organizations increasingly adopt cloud services alongside traditional on-premises infrastructure, GSLB solutions are evolving to seamlessly manage traffic across hybrid environments. This includes balancing loads between on-prem data centres, public clouds, and private clouds. Integrated cloud and hybrid deployments are already quite prevalent in the world of GSLB, but that trend will only continue as direct connections to the cloud providers gain popularity and cost decreases.
- **Automation and AI-Driven Optimization:** With the ubiquity of AI integrated products in the market it only makes sense that there is a use-case for adding the capabilities of Large Language Models or Machine



Learning to GSLB. AI could be leveraged to analyze traffic patterns and predict load spikes and tune the distribution of requests to enhance user experience.

- **Security Enhancements:** With rising cyber-security threats, GSLB solutions are integrating advanced security features. This includes DDoS protection, and integrating with Web Application Firewalls (WAFs) to ensure secure traffic distribution across global networks.
- **Edge Computing Integration:** With the rise of edge computing, GSLB is being adapted to work in conjunction with edge networks, distributing loads not just across data centers but also to edge nodes, bringing content closer to the user and further reducing latency. Whilst this is not an implementation method that we have used to date, it is a natural extension to the **twrr** distribution method. Distributing requests to the nearest edge compute node could greatly benefit user experience by directing requests to compute resources that are geographically very close to the requester
- **API-Driven Configurations:** Pretty much all of the trends that precede this one are dependent on an API enabled GSLB. This will allow the GSLB to make configuration changes based on the ever changing environment which it is functioning in. Increased use of APIs for configuring and managing GSLB settings allows for better integration with DevOps practices and facilitates more dynamic, programmable load balancing setups.
- **Sustainability Focus:** With a growing emphasis on energy efficiency and sustainability, GSLB solutions are increasingly being designed to optimize resource usage across data centres, contributing to reduced energy consumption and operational costs.

These trends indicate a shift towards more intelligent, secure, and flexible GSLB solutions that are well-aligned with the modern demands of digital enterprises, cloud computing, and the global distribution of users and services.

2.6. Conclusion

Global Server Load Balancing is a critical component in ensuring high availability, performance, and resilience of digital services in a globally connected platform. GSLB is not just a traffic director, but also a strategic enabler for businesses operating across dispersed geographical locations. We've seen its role in enhancing user experience through reduced latency, its importance in disaster recovery, and its growing integration with cloud and hybrid environments.

As it grows in popularity and application, the GSLB landscape is continually evolving, adapting to new challenges and technological advancements. As you move forward, remember that the core principles of GSLB remain constant: intelligent distribution of traffic, additional redundancy and high availability, and the seamless global delivery of services. Whether you're managing enterprise networks, architecting cloud solutions, or ensuring the smooth operation of large-scale object storage clusters, GSLB stands as a cornerstone in your toolkit.

3. Configuration

3.1. Global Names

The first step is to configure the Global Names. Global Names are the FQDNs that GSLB must respond to requests on. They must correspond to the configuration that is set in the DNS infrastructure.

To create a new Global Name, navigate to **Cluster Configuration > GSLB Configuration > Global Names** and click the **New Global Name** button.



System Overview

Local Configuration

Cluster Configuration

Layer 4 - Virtual Services

Layer 4 - Real Servers

Layer 4 - Advanced Configuration

Layer 7 - Virtual Services

Layer 7 - Real Servers

Layer 7 - Advanced Configuration

Layer 7 - Manual Configuration

SSL Termination

SSL Certificate

CA Certificate Families

SSL - Advanced Configuration

High Availability Configuration

Heartbeat Configuration

WAF - Gateway

WAF - Manual Configuration

WAF - Advanced Configuration

GSLB Configuration

GSLB Configuration

Global Names

Members

Pools

Topologies

3

New Global Name

No Data

Copyright © Loadbalancer.org Inc. 2002 – 2023

ENTERPRISE VA Max - v8.9.1

English

Enter the required details.

System Overview

Local Configuration

Cluster Configuration

Layer 4 - Virtual Services

Layer 4 - Real Servers

Layer 4 - Advanced Configuration

Layer 7 - Virtual Services

Layer 7 - Real Servers

Layer 7 - Advanced Configuration

Layer 7 - Manual Configuration

GSLB Configuration

Global Names

Members

Pools

Topologies

New Global Name

4

Submit

Cancel

New Global Name

Name

gslb-loadbalancer-ing

1

?

Hostname

gslb.loadbalancer.ing

2

?

TTL

30

seconds

3

?

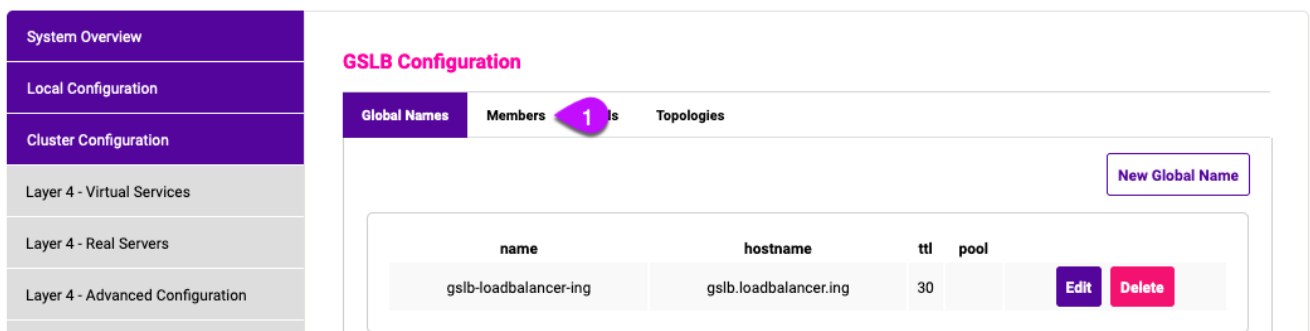
- **Name:** A local reference for the Global Name. This should be descriptive of the services that are that it is describing. The general guidance is to simply name it the same as the FQDN that it will be referencing. Name can be a combination of '0-9', 'a-z', 'A-Z', '-' (dash), '_' (underscore) or a '.' (dot).

- **Hostname:** The fully-qualified domain name that the GSLB is sending responses on behalf of.
- **TTL:** The Time To Live for the DNS record. DNS TTL represents the amount of time that a DNS response will be cached by the requestor's system,

3.2. Members

Members are the individual IP addresses that a request can be resolved to and are associated to the Global Name via a Pool.

If you are following on from the previous section "Global Names", then just click the **Members** tab at the top of the page.



The screenshot shows the 'GSLB Configuration' interface. On the left is a sidebar with navigation links: System Overview, Local Configuration, Cluster Configuration, Layer 4 - Virtual Services, Layer 4 - Real Servers, and Layer 4 - Advanced Configuration. The main area is titled 'GSLB Configuration' and has three tabs: Global Names, Members (which is selected and has a red badge with the number '1'), and Topologies. In the top right corner of the main area is a button labeled 'New Global Name'. Below the tabs is a table with the following data:

name	hostname	ttl	pool		
gslb-loadbalancer-ing	gslb.loadbalancer.ing	30		Edit	Delete

Alternatively, to create a new Member navigate to *Cluster Configuration > GSLB Configuration > Members*.

System Overview

Local Configuration

Cluster Configuration 1

Layer 4 - Virtual Services

Layer 4 - Real Servers

Layer 4 - Advanced Configuration

Layer 7 - Virtual Services

Layer 7 - Real Servers

Layer 7 - Advanced Configuration

Layer 7 - Manual Configuration

SSL Termination

SSL Certificate

CA Certificate Families

SSL - Advanced Configuration

High Availability Configuration

Heartbeat Configuration

WAF - Gateway

WAF - Manual Configuration

WAF - Advanced Configuration

GSLB Configuration 2

GSLB Configuration

Global Names

Members 3

Topologies

New Global Name

No Data

Copyright © Loadbalancer.org Inc. 2002 – 2023

ENTERPRISE VA Max - v8.9.1

English

Once there, click the **New Member** button.

System Overview

Local Configuration

Cluster Configuration

Layer 4 - Virtual Services

Layer 4 - Real Servers

GSLB Configuration

Global Names

Members

Pools

Topologies

1

New Member

No Data

Enter the required details.

System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers
Layer 4 - Advanced Configuration
Layer 7 - Virtual Services
Layer 7 - Real Servers
Layer 7 - Advanced Configuration
Layer 7 - Manual Configuration
SSL Termination

GSLB Configuration

Global Names
Members
Pools
Topologies

New Member

Name
dc1-lon-web
1
?

IP
198.51.100.10
2
?

Monitor IP
198.51.100.10
3
?

Weight
1
4

Submit
Cancel

- **Name:** As with the other elements, this is just a local reference for the Member that is displayed in the GSLB configuration and is not used in the consideration of any parts of the GSLB process. Name can be a combination of '0-9', 'a-z', 'A-Z', '-' (dash), '_' (underscore) or a '.' (dot).
- **IP:** The IP address of the Member. This is usually a VIP address but can also be a Node IP if implementing SDNS, as detailed below.
- **Monitor IP:** The IP address that the health check will be performed against. If this field is left empty then the value will default to the IP address. This should be left empty unless you specifically want to perform health checks against an IP address that is different from the Member IP.
- **Weight:** The relative frequency that this particular Member should be resolved in comparison to the other Members in the Pool.

If you have more Members - and you should - repeat the process to add them. The image below shows that two more Members have been added.

System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers
Layer 4 - Advanced Configuration
Layer 7 - Virtual Services
Layer 7 - Real Servers

GSLB Configuration

Global Names
Members
Pools
Topologies

New Member

name	ip	monitor_ip	weight	pools		
dc1-lon-web	198.51.100.10	198.51.100.10	1		Edit	Delete
dc2-mad-web	198.51.100.139	198.51.100.139	1		Edit	Delete
dc3-frk-web	203.0.113.10	203.0.113.10	1		Edit	Delete

3.2.1. Member IPs

Typically and historically, the IP that is configured for a Member is the VIP for the Virtual Service that you are performing GSLB for. With increasing frequency we are seeing a requirement for the Member IPs to be configured as the node server IPs themselves. When the GSLB is configured in this way it is referred to as SDNS (Smart

DNS), previously this was called GSLB direct-to-node.

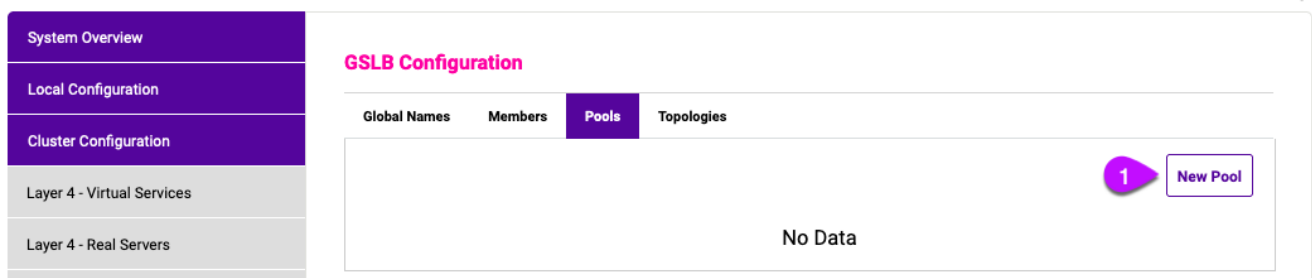
3.3. Pools

The Pool ties the elements of the GSLB configuration together. The Members are assigned to the Pool and the Pool is assigned to the Global Name. The majority of the configuration is applied within the Pool.

If you are following on from the previous section "Members", then just click the Pools tab at the top of the page.

Alternatively, to access the Pools configuration, navigate to *Cluster Configuration > GSLB Configuration > Pools*.

click **New Pool**.



Enter the required details.

System Overview

Local Configuration

Cluster Configuration

Layer 4 - Virtual Services

Layer 4 - Real Servers

Layer 4 - Advanced Configuration

Layer 7 - Virtual Services

Layer 7 - Real Servers

Layer 7 - Advanced Configuration

Layer 7 - Manual Configuration

SSL Termination

SSL Certificate

CA Certificate Families

SSL - Advanced Configuration

High Availability Configuration

Heartbeat Configuration

WAF - Gateway

WAF - Manual Configuration

WAF - Advanced Configuration

GSLB Configuration

Health Check Scripts

Floating IPs

Setup Wizard

GSLB Configuration

Global NamesMembersPoolsTopologies

New Pool

New Pool

Namegslb-web1?

MonitorHTTP2?

Monitor Use SSLNo3?

Monitor Hostnamegslb.loadbalancer.ing4?

Monitor URL Path/5?

Monitor Port806?

Monitor Expected Codes2007?

LB Methodwrr8?

Global Namesgslb-loadbalancer-ing9?

Members

Available Members

Members In Use

dc1-lon-web

dc2-mad-web

dc3-frk-web

Advanced

Submit

Cancel

- **Name:** As with the other elements, this is just a local reference for the Pool. Name can be a combination of '0-9', 'a-z', 'A-Z', '-' (dash), '_' (underscore) or a '.' (dot).
- **Monitor:** The type of health check monitor that will be perform on the Members. For more details, see [Health Checking](#).
- **LB Method:** The distribution model/method that the Pool should employ for incoming requests. For more details, see [Load Distribution](#).

Required Members can be dragged from the *Available Members* list to the *Members in Use* list.

3.3.1. Advanced Configuration

Monitor Interval

The number of seconds between each health check, the default is every 10 seconds. If you require health checks to be performed more (or less) frequently.

Monitor Timeout

The number of milliseconds that a health check response must be received within to be considered a successful

response. For example, by default if a response is received 5001 milliseconds after being sent, it will be marked as a failed health check.

Monitor Retries

The number of times that a health check will be retried before being declaring a Member down. The number of health check failures permitted by the system is always one more than the number of retries configured. The default configuration is for two retries which means there will a total of 3 health check failures before a Member is marked as being down.

Fallback

The fallback method defines what should happen if all the Members of a Pool are failing health checks. There are two options: refuse and any.

- **any:** This is the default and it will return an answer from the Pool for any node that does not have a weight of 0. That means when you have a fallback method of any you will get receive a response that points you to a known non-working IP.
- **refuse:** The lookup will simply fail in this instance. A DNS response will not be sent as none of the Members are online (according to health checks) to process requests sent to it.

If you are configuring a **LB Method** of **wrr** or **fogroup** then you are done. Restart GSLB to apply the new settings.

System Overview

Local Configuration

Cluster Configuration

Layer 4 - Virtual Services

Layer 4 - Real Servers

Layer 4 - Advanced Configuration

Layer 7 - Virtual Services

Layer 7 - Real Servers

Layer 7 - Advanced Configuration

Layer 7 - Manual Configuration

SSL Termination

Commit changes

The configuration of the following services has been changed. When reconfiguration is complete, restart/reload the services to commit the changes

Restart GSLB 1

GSLB Configuration

Global Names

Members

Pools

Topologies

New Pool

name	monitor	lb_method	fallback	members	
gslb-web	http	wrr	any	dc1-lon-web, dc2-mad-web, dc3-frk-web	<div>EditDelete</div>

3.4. Topologies

Topologies are slightly confusingly placed at the end of the UI. To configure a Pool with a **LB Method** of **twrr** you must first configure a topology for it. The simple way to overcome this little hurdle is to first define your Pool with the **LB Method** set to **wrr**, configure your topologies and then go back and update the Pool to use **twrr**.

If you are following on from the previous section "Pools", then just click the Topologies tab at the top of the page.

Alternatively, to access the Topologies configuration, navigate to **Cluster Configuration > GSLB Configuration > Topologies**.

System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers
Layer 4 - Advanced Configuration

GSLB Configuration

Global Names Members Pools Topologies

1

New Pool

name	monitor	lb_method	fallback	members	
gslb-web	http	wrr	any	dc1-lon-web, dc2-mad-web, dc3-frk-web	Edit Delete

Click **New Topology**.

System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers

GSLB Configuration

Global Names Members Pools Topologies

1

New Topology

No Data

Enter the required details.

System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers
Layer 4 - Advanced Configuration
Layer 7 - Virtual Services
Layer 7 - Real Servers
Layer 7 - Advanced Configuration
Layer 7 - Manual Configuration
SSL Termination

GSLB Configuration

Global Names Members Pools Topologies

New Topology

New Topology

Name	<input type="text" value="dc1_london"/>	1	?
IP/CIDR	<input type="text" value="198.51.100.10/32, 172.31.0.0/24"/>	2	?

3

Submit

Cancel

- **Name:** As with the other elements, this is just a local reference for the Topology. Name can be a combination of '0-9', 'a-z', 'A-Z', '-' (dash), '_' (underscore) or a '.' (dot).
- **IP/CIDR:** A comma separated list of valid IPv4 address or CIDR denoted IP blocks for example '10.1.1.0/24, 10.1.1.2'.

Then repeat for the next DC or IP range.



System Overview

Local Configuration

Cluster Configuration

Layer 4 - Virtual Services

Layer 4 - Real Servers

Layer 4 - Advanced Configuration

Layer 7 - Virtual Services

Layer 7 - Real Servers

Layer 7 - Advanced Configuration

Layer 7 - Manual Configuration

SSL Termination

SSL Certificate

CA Certificate Families

SSL - Advanced Configuration

High Availability Configuration

Heartbeat Configuration

WAF - Gateway

WAF - Manual Configuration

WAF - Advanced Configuration

Commit changes

The configuration of the following services has been changed. When reconfiguration is complete, restart/reload the services to commit the changes

Restart GSLB

GSLB Configuration

Global Names

Members

Pools

Topologies

New Topology

Name

dc2_madrid

1

?

IP/CIDR

198.51.100.139/32, 172.31.64.0/24

2

?

3

Submit

Cancel

name	ips		
dc1_london	198.51.100.10/32, 172.31.0.0/24	Edit	Delete

Repeat again if you have another DC or IP range.

System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers
Layer 4 - Advanced Configuration
Layer 7 - Virtual Services
Layer 7 - Real Servers
Layer 7 - Advanced Configuration
Layer 7 - Manual Configuration
SSL Termination
SSL Certificate
CA Certificate Families
SSL - Advanced Configuration
High Availability Configuration
Heartbeat Configuration
WAF - Gateway
WAF - Manual Configuration
WAF - Advanced Configuration
GSLB Configuration

Commit changes

The configuration of the following services has been changed. When reconfiguration is complete, restart/reload the services to commit the changes

Restart GSLB

GSLB Configuration

Global Names
Members
Pools
Topologies

New Topology

Name
dc3_frankfurt
1

IP/CIDR
203.0.113.10/32, 172.31.128.0/24
2

3
Submit
Cancel

name	ips		
dc1_london	198.51.100.10/32, 172.31.0.0/24	Edit	Delete
dc2_madrid	198.51.100.139/32, 172.31.64.0/24	Edit	Delete

Once you have configured all required topologies, restart GSLB.

System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers
Layer 4 - Advanced Configuration
Layer 7 - Virtual Services
Layer 7 - Real Servers
Layer 7 - Advanced Configuration
Layer 7 - Manual Configuration
SSL Termination
SSL Certificate
CA Certificate Families

Commit changes

The configuration of the following services has been changed. When reconfiguration is complete, restart/reload the services to commit the changes

Restart GSLB
1

GSLB Configuration

Global Names
Members
Pools
Topologies

New Topology

name	ips		
dc1_london	198.51.100.10/32, 172.31.0.0/24	Edit	Delete
dc2_madrid	198.51.100.139/32, 172.31.64.0/24	Edit	Delete
dc3_frankfurt	203.0.113.10/32, 172.31.128.0/24	Edit	Delete

If you need to go back to update your Pool from **wrr** to **twrr**, select Pools.



System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers
Layer 4 - Advanced Configuration
Layer 7 - Virtual Services
Layer 7 - Real Servers

GSLB Configuration

Global Names
Members
Pools
Topologies

New Topology

name	ips		
dc1_london	198.51.100.10/32, 172.31.0.0/24	Edit	Delete
dc2_madrid	198.51.100.139/32, 172.31.64.0/24	Edit	Delete
dc3_frankfurt	203.0.113.10/32, 172.31.128.0/24	Edit	Delete

Edit your wrr Pool.

System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers
Layer 4 - Advanced Configuration

GSLB Configuration

Global Names
Members
Pools
Topologies

New Pool

name	monitor	lb_method	fallback	members		
gslb-web	http	wrr	any	dc1-lon-web, dc2-mad-web, dc3-frk-web	1	Edit Delete

Update the *LB Method* to **twrr**.

Layer 4 - Advanced Configuration
Layer 7 - Virtual Services
Layer 7 - Real Servers
Layer 7 - Advanced Configuration
Layer 7 - Manual Configuration
SSL Termination
SSL Certificate
CA Certificate Families
SSL - Advanced Configuration
High Availability Configuration
Heartbeat Configuration
WAF - Gateway
WAF - Manual Configuration
WAF - Advanced Configuration
GSLB Configuration
Health Check Scripts
Floating IPs
Setup Wizard

Name

gslb-web

?

Monitor

HTTP

?

Monitor Use SSL

No

?

Monitor Hostname

gslb.loadbalancer.ing

?

Monitor URL Path

/

?

Monitor Port

80

?

Monitor Expected Codes

200

?

LB Method

twrr

1

?

Global Names

gslb-loadbalancer-ing

?

Members

Available Members

Members In Use

dc1-lon-web

dc2-mad-web

dc3-frk-web

?

Advanced

2

Submit

Cancel

Now restart GSLB.

System Overview
Local Configuration
Cluster Configuration
Layer 4 - Virtual Services
Layer 4 - Real Servers
Layer 4 - Advanced Configuration
Layer 7 - Virtual Services
Layer 7 - Real Servers
Layer 7 - Advanced Configuration
Layer 7 - Manual Configuration
SSL Termination

Commit changes

The configuration of the following services has been changed. When reconfiguration is complete, restart/reload the services to commit the changes

Restart GSLB

1

GSLB Configuration

Global Names

Members

Pools

Topologies

New Pool

name	monitor	lb_method	fallback	members	
gslb-web	http	twrr	any	dc1-lon-web, dc2-mad-web, dc3-frk-web	<div>Edit</div> <div>Delete</div>

3.5. DNS Setup

Before the GSLB and it's configuration can be tested in a meaningful way, DNS must be configured. Typically in a production environment it's best to configure the DNS delegation and configuration for the network before you go to configure the GSLB on the load balancer appliances. This does depend on your DNS infrastructure. Convergence of the DNS configuration can take hours to be fully synchronized. Because of this, it is practical to



complete the DNS configuration before implementing the GSLB. You must consider that it's a double-edged sword. If you add the DNS entries before the GSLB is configured then the network will not be able to correctly resolve those requests. Unfortunately, this can also be one of the most confusing parts of a GSLB implementation so getting it right in the first place is not always straightforward. For applications that have a corresponding [deployment guide](#) it's worth reading that first as some of them contain product-specific DNS delegation guidance.

3.5.1. Non-Windows Servers

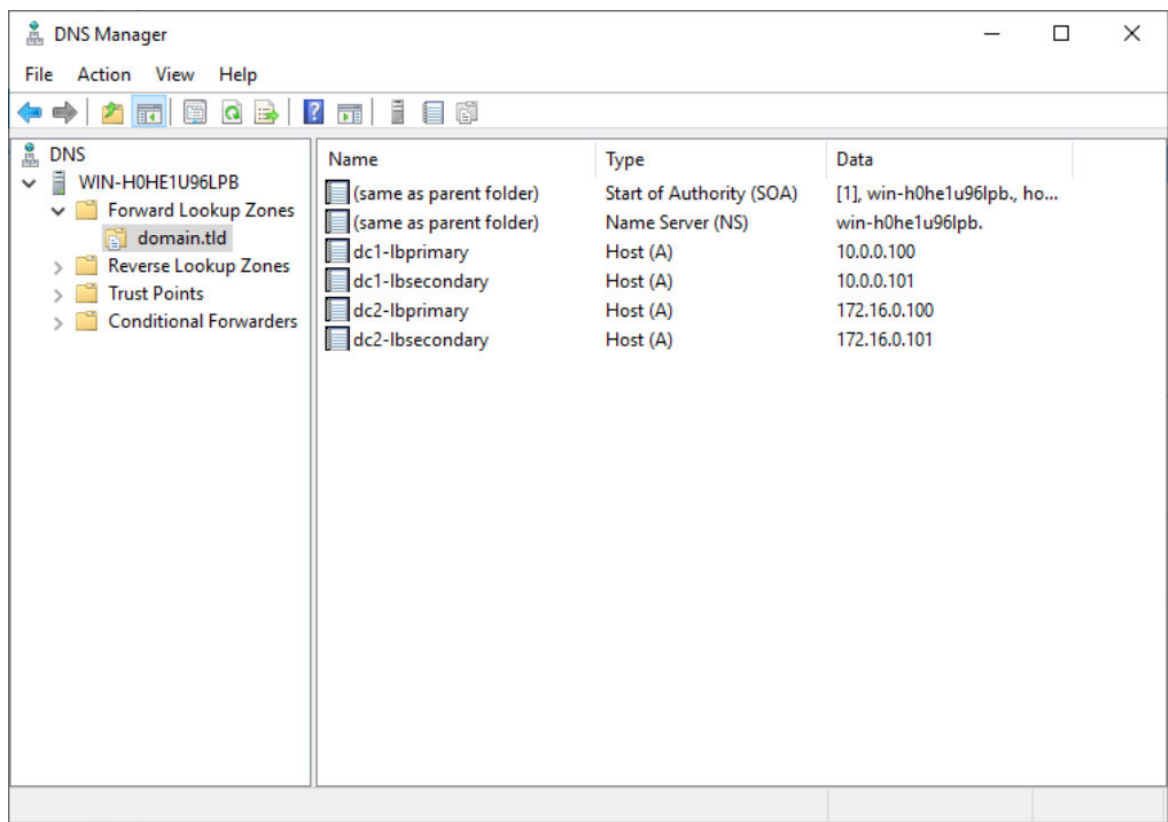
Since there are many different ways that DNS can be implemented, it's not feasible to provide information on all of them here. If you are running something other than a Windows-based DNS infrastructure, the designated subdomains are typically delegated using NS DNS records. You can also do this via conditional forwarding. Which ever method is used, DNS must be configured so that requests for the configured Global Names are resolved by GSLB on the load balancer appliances.

3.5.2. Windows Servers

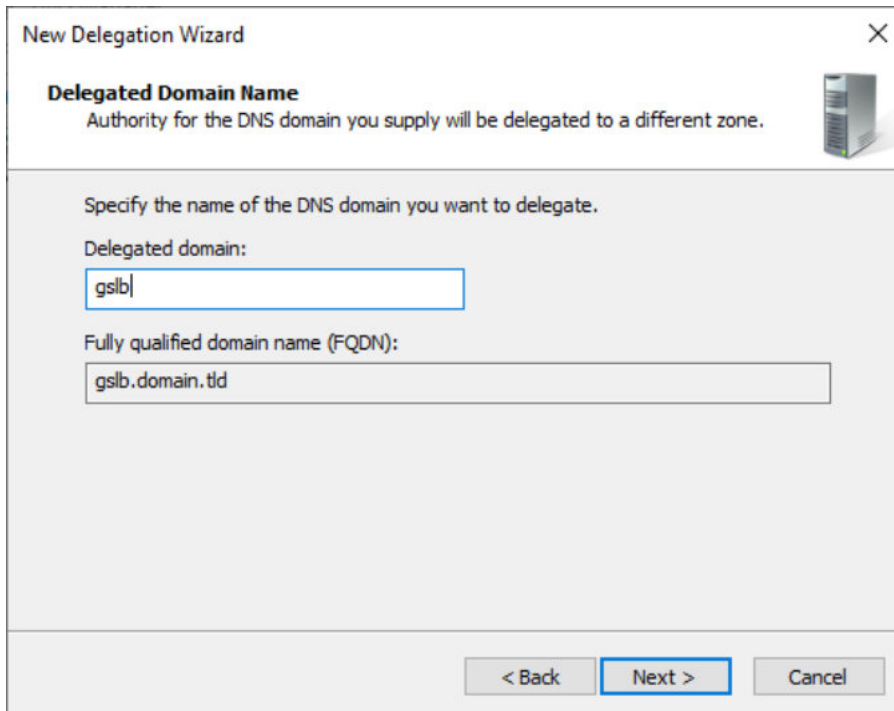
The configuration of the DNS delegation on Windows servers is also covered in [this blog post](#).

DNS entries should be created for each of the loadbalancer IP addresses. Similar to the following image.

Open DNS Manager and create A records for every load balancer at every site, using Action > New Host (e.g. **dc1-lbprimary.domain.tld**, **dc1-lbsecondary.domain.tld**, **dc2-lbprimary.domain.tld** and **dc2-lbsecondary.domain.tld**).



Enter the domain delegation, in this example that is "gslb" and it auto-populates the FQDN field. If that auto-populated field contains the correct value then click Next.



New Delegation Wizard

Delegated Domain Name
Authority for the DNS domain you supply will be delegated to a different zone.

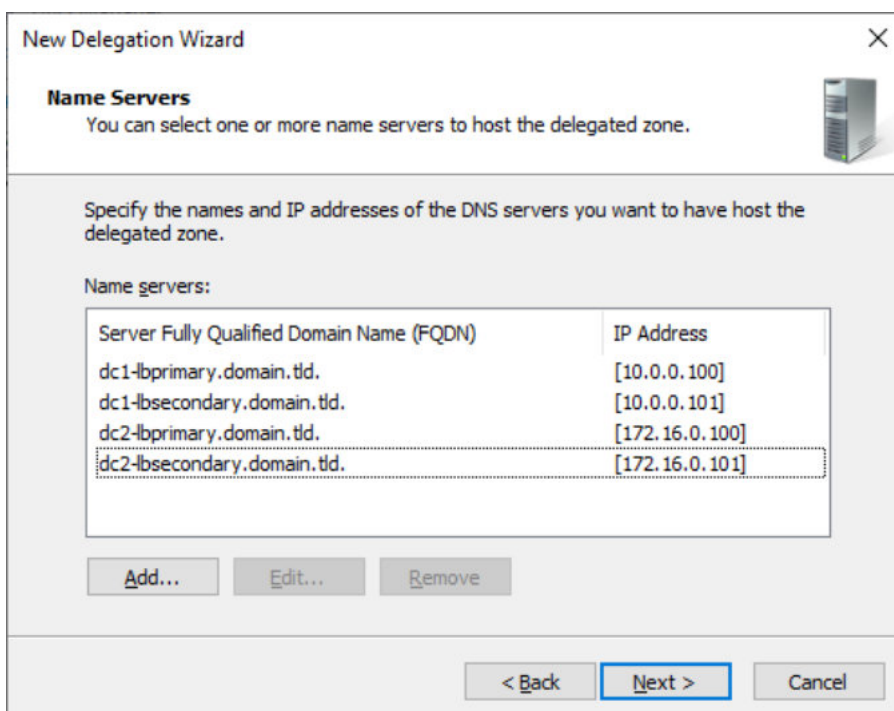
Specify the name of the DNS domain you want to delegate.

Delegated domain:
gslb

Fully qualified domain name (FQDN):
gslb.domain.tld

< Back Next > Cancel

Provided that the load balancer part of the GSLB configuration has been completed and is working, the New Delegation wizard should now be used to delegate the subdomain to the load balancers. The delegation will use the new FQDNs for the load balancers, as defined in the previous step. The delegation wizard is located at **Action > New Delegation**.



New Delegation Wizard

Name Servers
You can select one or more name servers to host the delegated zone.

Specify the names and IP addresses of the DNS servers you want to have host the delegated zone.

Name servers:

Server Fully Qualified Domain Name (FQDN)	IP Address
dc1-lbprimary.domain.tld.	[10.0.0.100]
dc1-lbsecondary.domain.tld.	[10.0.0.101]
dc2-lbprimary.domain.tld.	[172.16.0.100]
dc2-lbsecondary.domain.tld.	[172.16.0.101]

Add... Edit... Remove

< Back Next > Cancel

4. Troubleshooting

4.1. GSLB Diagnostics

Two reports are available to view the current state of the running GSLB service. These reports are very useful when first setting up GSLB and also when diagnosing any issues. They are available via the WebUI menu option:

Reports.

GSLB Generic State – This report shows information about the running configuration of GSLB and also the health state of each Member.

Example:

```
{
  "timestamp": 1658309678.8496737,
  "globalnames": {
    "service.domain.com": {
      "pool_name": "service-nodes",
      "name": "service.domain.com",
      "ttl": 30
    }
  },
  "pools": {
    "service-nodes": {
      "last_status": true,
      "max_addrs_returned": 1,
      "members": [
        {
          "name": "nodes-dc1",
          "status": true,
          "ip": "10.0.0.10",
          "weight": 1,
          "region": "None",
          "status_reason": "monitor passed",
          "retries_left": 2,
          "monitor_ip": "10.0.0.10"
        },
        {
          "name": "nodes-dc2",
          "status": false,
          "ip": "172.16.0.10",
          "weight": 1,
          "region": "None",
          "status_reason": "socket.connect()",
          "retries_left": 0,
          "monitor_ip": "172.16.0.10"
        }
      ],
      "name": "service-nodes",
      "monitor": {
        "send_string": "None",
        "timeout": 5,
        "name": "tcp",
        "match_re": "None",
        "retries": 2,
        "interval": 10,
        "port": 80
      },
      "fallback": "any",
      "lb_method": "twrr"
    }
  }
}
```

```
}
```

This shows that:

- Member 10.0.0.10 is passing its health check
- Member 172.16.0.10 is failing its health check

GSLB PPDNS State – This report shows information about the running configuration of GSLB and also shows which results will be returned to inbound queries based on the current state of all Members.

Example:

```
{
  "timestamp": 1658309678.9508624,
  "pools": {
    "service-nodes": {
      "lb_method": "twrr",
      "max_addrs_returned": 1,
      "status": true,
      "fallback": "any",
      "dist_tables": {
        "_default": {
          "index": 0,
          "num_unique_addrs": 1,
          "rotation": [
            "10.0.0.10"
          ]
        }
      }
    }
  },
  "globalnames": {
    "service.domain.com": {
      "pool_name": "service-nodes",
      "ttl": 30
    }
  }
}
```

This shows that:

- Only Member 10.0.0.10 is currently in the rotation of addresses being returned because as shown in the **GSLB Generic State** example above, Member 172.16.0.10 is failing its health check

4.2. EDNS-CS

Support for **E**xtension mechanisms for **D**omain **N**ame **S**erver - **C**lient **S**ubnet (EDNS-CS) was introduced to the Enterprise appliances from version 8.11. The principal method of troubleshooting issues with client subnet evaluation is to first log the entries that are being observed by PDNS to ensure that EDNS-CS information is correctly being sent to the GSLB for evaluation against the topology file.



Open the `/etc/pdns/pdns.conf` file in your preferred editor.

Add the following lines to the end of the `pdns.conf` file and save it:

```
loglevel=7
log-dns-queries=yes
log-dns-details=yes
```

Restart the PDNS service:

```
service pdns restart
```

Once PDNS logging is set to the appropriate level and the service has been restarted, the DNS queries are being logged to `/var/log/messages`. So, you can tail that file to see the DNS queries that are coming in. When we are talking about troubleshooting issues with Extended DNS Client Subnet (EDNS-CS)

```
Feb 30 12:34:56 lbprimary pdns[9001]: Remote 10.100.0.10<-192.168.65.0/24 wants
'cloudy.with.meatballs|A', do = 0, bufsize = 1232 (4096)
Feb 30 12:34:56 lbprimary pdns[9001]: Remote 10.100.0.10 wants 'cloudy.with.meatballs|A', do = 0,
bufsize = 1232 (4096)
```

In this case, the information that we are most concerned with is the Remote IP, the GSLB uses this to match against the topologies configured in the load balancer. If there is an issue with the Remote IP that is being offered up to the GSLB that can cause issues with twrr working the way that we need it to.

The first request is showing us the following:

```
Remote 10.100.0.10<-192.168.65.0/24 wants 'cloudy.with.meatballs'
```

What this is telling us is that the actual request that came in to the GSLB is sourced from 10.0.51.14 but the Client Subnet field has been set with the originating client subnet configured as 192.168.65.0/24. As the client subnet field is set, that is what the GSLB will use to evaluate against the topology configuration. The client subnet field can be set if the DNS request originates from a completely different than the IP address that has forwarded the address to the GSLB, in this case it is because I manually set the field on the lookup, like so:

```
dig @192.168.100.80 cloudy.with.meatballs +subnet=192.168.65.0/24
```

The second request doesn't have the extra information of the Client Subnet field:

```
Remote 10.100.0.10 wants 'cloudy.with.meatballs'
```

With this lookup the remote IP will be evaluated against the remote IP as the Client Subnet field is not set. This request came in after issuing this command on the client:



```
nslookup cloudy.with.meatballs 192.168.100.80
```

The DNS server was manually set as my DNS server is not set up to point at the GSLB for this domain so a manual override was used.

With the information from the PDNS logging you will be able to ascertain whether the DNS requests are coming from the correct location. You will also be able to see if the Client Subnet field has been populated. With these points of information in hand you should be able to understand whether the issue is in the topology configuration, the DNS infrastructure, or some other part of the platform.

5. More Information

For more information, please refer to:

- [Loadbalancer.org - Blog Posts](#)
- [Cloudflare - GSLB Article](#)
- [Efficient IP - What is GSLB](#)
- [Loadbalancer.org - Manuals](#)
- [Loadbalancer.org - Deployment Guides](#)

6. Loadbalancer.org Technical Support

If you have any questions regarding GSLB or need assistance with load balancing your application, please don't hesitate to contact support@loadbalancer.org.





Visit us: www.loadbalancer.org

Phone us: +44 (0)330 380 1064

Phone us: +1 833 274 2566

Email us: info@loadbalancer.org

Follow us: [@loadbalancer.org](https://twitter.com/loadbalancer.org)

About Loadbalancer.org

Loadbalancer.org's mission is to ensure that its clients' businesses are never interrupted. The load balancer experts ask the right questions to get to the heart of what matters, bringing a depth of understanding to each deployment. Experience enables Loadbalancer.org engineers to design less complex, unbreakable solutions - and to provide exceptional personalized support.

