Orchestration Server Deployment Guide

Load Balancing

12/14/2025

Contents

Load Balancing

The ORS **cluster** deployment introduces the possibility to distribute the load of incoming Interactions (voice calls, multimedia and http requests) across all ORS Nodes.

Load Balancing for Voice Interactions

Each Node in cluster serves the same T-Server/SIP Server so all Nodes are aware of all voice calls. However, each Node has responsibility for specific sessions.

The ORS cluster is designed to have all calls distributed across all existing ORS instances uniformly.

Load Balancing for Multimedia Interactions

The nature of the "pulling" mechanism provided by Interaction Server guarantees that new sessions created from this trigger are automatically distributed across the cluster.

Load Balancing of HTTP-Related Interactions

If a session is created via the HTTP interface, the session is created on the ORS running in Primary mode, and the session is assigned to that Node. An HTTP load balancer is placed in front of the ORS cluster and provides load balancing services for HTTP requests.

ORS exposes the RESTful Web Services interface, which is based on HTTP. ORS may accept and execute a set of HTTP requests. This interface uses standard mechanisms of load balancing that are typical for web servers. These mechanisms include:

- DNS-based load balancing
- Hardware-based load balancing.

These methods result in distribution of HTTP requests across all ORS Nodes in a cluster.

Load Balancing Optimization

To enable optimal load-balancing, ORS supports "stickiness" of HTTP sessions. This is achieved by providing a cookie with the SCXML session ID in the HTTP responses. Hardware load balancers may then use this cookie to ensure that HTTP requests are about the same SCXML session.

There could be a scenario when an HTTP request about a specific SCXML session is delivered to an

ORS Node that is not handling this SCXML session. In this case, the ORS Node replies with the HTTP response `307 Temporary Redirect` pointing to the ORS Node that is processing the needed SCXML session. The responsibility of the HTTP Client is to resend the same request to the given URI, which results in the request arriving at the correct Node.

For RESTful interface, you must set up the http port definition under **Server Info** during ORS installation, as described in Creating the ORS Application Object.

SCXML Session Startup Rules

When an existing SCXML session (Session1) starts another SCXML session (Session2), the child SCXML session (Session2) is always started on the same ORS Node as the parent SCXML session (Session1).

When an existing SCXML session (Session1) invokes another SCXML session (Session2), the invoked SCXML session (Session2) is always started on the same ORS Node as the existing SCXML session (Session1).

Load Balancing for RESTful Interface

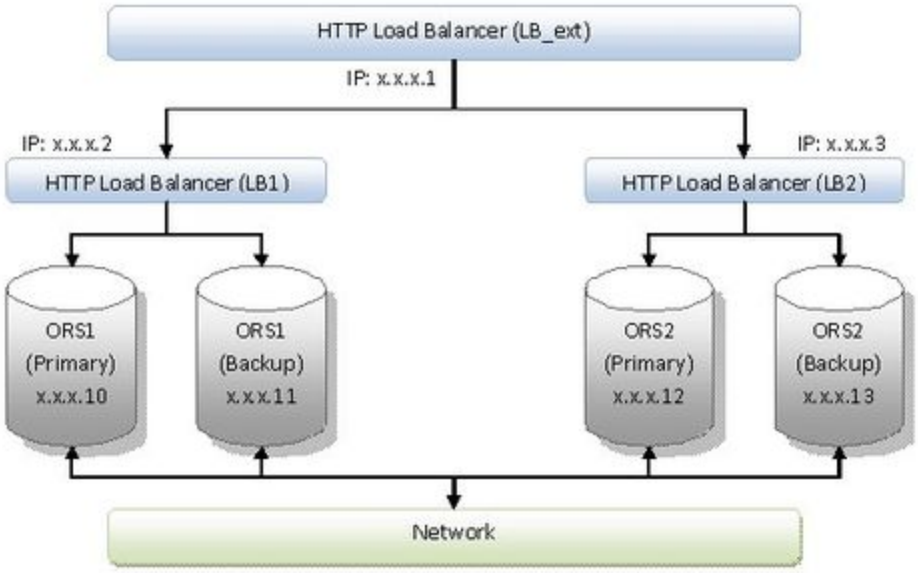
In support of the RESTful web services interface, when ORS accepts and executes a set of HTTP requests, it uses the standard mechanisms of load balancing that are typical for web servers. These mechanisms include:

- DNS-based load balancing
- Hardware-based load balancing

Sessions created via the RESTful web services interface will be assigned to the ORS node that services the initial request. If subsequent HTTP requests targeting the same session (by SCXML session ID) are not delivered to the same ORS Node, they will be redirected with the following priorities:

1. If the recipient of the request is not part of the target ORS Node, it will be redirected to the correct ORS Node using their configured external-url.
2. If the HTTP request was delivered to the ORS instance running in Backup mode (or if the target server has no `external-url` configured), it will be redirected to the ORS instance running in Primary mode by its configured `Host` and port.

Taking the above behaviour into consideration, the following diagram illustrates the recommended deployment of ORS and load balancers:



In the above diagram, ORS1 (both Primary and Backup servers) will have the address of `ORS1 HTTP Load Balancer` configured as the external-url. Similarly, the ORS2 will have `ORS2 HTTP Load Balancer` configured as its `external-url`. The load balancers will distribute HTTP requests across all ORS Nodes in the cluster and the ORS Nodes will automatically ensure that all requests corresponding to one session will be delivered to the correct Node.

To enable optimal load-balancing, ORS supports the following features:

- Session "stickiness" (to ensure requests are delivered to the correct ORS Node).
- Heartbeats/health monitoring (to ensure that requests are not delivered to offline ORS instances or instances running in backup modes).

Configuring Session Stickiness

Session "stickiness" is achieved by providing a cookie with the SCXML session ID in the HTTP responses. Load balancers (such as Apache, or HAProxy) may then use this cookie to ensure that HTTP requests are about the same SCXML session. ORS automatically sets the `Set-Cookie` header in the responses for `Create Session` requests and redirected requests. To configure session “stickiness” in Apache:

- Assuming a deployment similar to the above diagram, in `httpd.conf`, for LB_ext

```
ProxyPass / balancer://orsapcluster1/ stickysession=ORSSESSIONID
<Proxy balancer://orsapcluster1>
        BalancerMember http://apvm1.us.int.genesyslab.com:3482 loadfactor=50
route=<ORS1.node_id>
        BalancerMember http://apvm2.us.int.genesyslab.com:3482 loadfactor=50
route=<ORS2.node_id>
</Proxy>
```

- LB1 (apvm1.us.int.genesyslab.com:3482):

```
<Proxy balancer://node749>
        BalancerMember http://172.21.82.55:7031 route=node749
        BalancerMember http://172.21.82.55:7041 route=node749
</Proxy>
```

- LB2 (apvm2.us.int.genesyslab.com:3482):

```
<Proxy balancer://node753>
        BalancerMember http://192.168.14.245:7041 loadfactor=70 route=node753
        BalancerMember http://192.168.14.245:7031 loadfactor=30 route=node753
</Proxy>
```

- To configure session “stickiness” in HAProxy, assuming a deployment similar to the above diagram, in haproxy.conf, on LB_ext

```
listen my_loadbalancer <my_loadbalancer IP>
        mode http
        appsession ORSSESSIONID len 100 timeout 3h
        server LB_1 <LB_1.address>
        server LB_2 <LB_2.address>
```

- In haproxy.conf, on LB_1

```
listen my_loadbalancer <my_loadbalancer IP>
        mode http
        appsession ORSSESSIONID len 100 timeout 3h
        server ORS1_P <ORS1_P.address>
        server ORS1_B <ORS1_B.address>
```

- In haproxy.conf, on LB_2

```
listen my_loadbalancer <my_loadbalancer IP>
        mode http
        appsession ORSSESSIONID len 100 timeout 3h
        server ORS2_P <ORS2_P.address>
        server ORS2_B <ORS2_B.address>
```

Configuring Server Health Monitoring

ORS provides a special “heartbeat” interface which can be used by external clients to determine whether the ORS instance is operating in Primary or Backup mode. This interface may be accessed via an HTTP GET request to:

http://<ORS_host>:<ORS_HTTP_port>/heartbeat

If the requested ORS instance is running in Primary mode, it will return a `200 OK` response. Otherwise, if the requested ORS instance is running in Backup mode, it will respond with the configured response code as specified in the ORS options (under `orchestration/heartbeat-backup-status`). If no response code is configured ORS will respond with default response code, `503 Service unavailable`. Load Balancers (such as **F5** or **HAProxy**) can use this interface to avoid redistributing traffic to backup ORS or unresponsive servers. **Note**: The `orchestration/heartbeat-backup-status` option should be configured on both ORS applications-Primary and Backup.

To configure health-monitoring in **F5**:

1. Log into **F5** load balancer web console (or CLI).

2. Go to **Local Traffic** > **Monitors**.
3. Click **Create**.
4. In the new screen, set the **Type** to HTTP.
5. Set the **Send** string to: `GET /heartbeat HTTP/1.0\r\n\r\n`.
6. Set the **Receive** string to: `HTTP/1.[01] 20[0-6]`. The above example matches HTTP status codes 200-206.
7. Save the new monitor.
8. Go to the pool list (assuming that the load balancer pool has already been configured), and apply the monitor to the appropriate pools.

To configure health-monitoring in **HAProxy**:

- Assuming deployment similar to above diagram, in haproxy.conf, on LB_1

```
listen my_loadbalancer <my_loadbalancer IP>
        mode http
        option httpchk GET /heartbeat HTTP/1.1\r\nHost:\ www
        server ORS1_P <ORS1_P.address> check
        server ORS1_B <ORS1_B.address> check
```

- In haproxy.conf, on LB_2

```
listen my_loadbalancer <my_loadbalancer IP>
        mode http
        option httpchk GET /heartbeat HTTP/1.1\r\nHost:\ www
        server ORS2_P <ORS2_P.address> check
        server ORS2_B <ORS2_B.address> check
```