



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

# eServices Deployment Guide

Interaction Server Cluster

5/14/2025

# Interaction Server Cluster

Starting with release 8.5.106.x of Interaction Server and 8.5.107.x of Interaction Server Proxy, you can configure multiple Interaction Servers into a cluster that works with a single instance of Interaction Server Proxy.

## Cluster Deployment Overview



Interaction Server Cluster Deployment

Each server node in an Interaction Server cluster has its own interaction database and its own event log databases (or other types of **Event Logger**). And each node must be configured in pairs, with one server configured as primary and the other as backup.

Interaction Server Proxy connects to all of the Interaction Server pairs in the cluster, presenting them to its clients as a single Interaction Server application. This includes presenting a single, consistent reporting event stream to reporting clients.

The proxy routes requests to the appropriate server or servers within a cluster, based on the cluster configuration and on the interaction identifier or agent specified in the request.

The proxy also load-balances requests, where appropriate, based on the list of tenants supported by each Interaction Server, the list of media types supported by each server, and other request attributes.

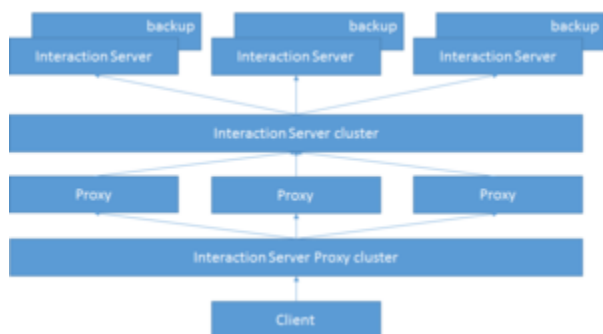
By dividing the incoming stream of interactions between multiple servers and databases, the cluster can be scaled to support a greater number of interactions than a single Interaction Server with a single database.

## Cluster Deployment with Multiple Proxies



Cluster Deployment with Multiple Proxies

Interaction Server Proxy does not maintain a persistent state, meaning that it can assume a specific state by making the appropriate requests to the appropriate Interaction Server nodes in the cluster. Because of this, you can set up several proxies working with the same cluster of Interaction Servers and each serving different clients. By using multiple proxies and distributing the clients between them, you can support a greater number of clients than you can with a single proxy.



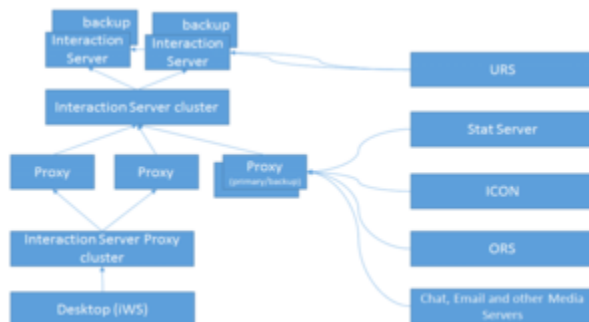
Interaction Server Cluster Configuration

Use the following two additional configuration objects of Application Cluster type to simplify cluster configuration:

- **Interaction Server cluster:**
  - The Interaction Server cluster configuration object holds the connections to all of the Interaction Server configuration objects in the cluster.
  - All of the proxy applications can share the same cluster configuration.
  - All proxies use the Interaction Server cluster configuration to connect to **all** Interaction Servers in the cluster **simultaneously**.
  - The proxy implementation can use either:
    - The Interaction Server cluster
    - Direct connections to Interaction Servers
    - Or a combination of the two
- **Interaction Server Proxy cluster:**

- The Interaction Server Proxy cluster configuration object holds the connections to all of the Interaction Server proxy configuration objects.
- All client applications can share the same proxy cluster configuration.
- The client applications use the Interaction Server Proxy cluster to choose one proxy randomly and are connected to a single proxy at any given moment.

## Suggested Deployment Configuration



Suggested Cluster Deployment Configuration

## Special Considerations

### Media Servers

Media servers (including chat and e-mail) should be connected to a primary/backup pair of Interaction Server Proxies. Depending on the expected load, media server can share a pair of Proxies with other applications or can use a dedicated pair.

### Genesys Desktop Apps

When using an Interaction Server Cluster with Genesys desktop applications such as Workspace Desktop Edition, you may use an N+1 architecture, where each Interaction Server Proxy configured for the Cluster of Proxies should only be run as a single application, with no backup instance. A desktop application configured with a connection to the Proxy Cluster will choose any available Proxy in the cluster.

### Reporting and Orchestration

Separate primary/backup proxy pairs should be configured specifically to be used by reporting clients (Stat Servers and ICON) and, if appropriate, Orchestration Server (ORS). For reporting clients and ORS, you can deploy additional primary/backup proxy pairs, as needed.

### Universal Routing Server

Universal Routing Server (URS) should be connected directly to all primary Interaction Servers in the cluster.

### Configuration Recommendations

Genesys recommends the following settings for Interaction Server nodes:

- login-session-timeout (this option was introduced in release 8.5.107)—1440, the default.
- not-ready-on-invitation-timeout—false.
- agent-login-control—Set to the same value on all Interaction Servers in all nodes.
- allow-multiple-agent-connections—false, in order to avoid the significant increase in the flow of notification events that Agent Desktop applications operating in a Cluster environment may experience if they are allowed to create multiple connections to Interaction Server for the same Agent/Place.

### Step by Step Cluster Configuration Process

1. Create the primary/backup pair of Interaction Servers with the required Database Access Point, following the process for a single server deployment.
2. Repeat Step 1 as many times as needed, keeping in mind that a single node within the cluster will be processing some dedicated solution or media type (for example, iWD or e-mail channel) or will be processing some media type in collaboration with other nodes.
3. Create Interaction Server Proxy applications using the Interaction Server application type, following the process for single Interaction Server Proxy deployment.
4. Repeat Step 3 as many times as necessary. Keep in mind that you can always add additional Proxy applications later if necessary.
5. At this point you can **use Cluster Manager Plug-in for GAX to create clusters and assign media types**, then continue with Step 6 below. If you are not using Cluster Manager, carry out Steps (a)–(f), then continue on to Step 6.
  - a. Optionally, create a cluster-settings section in the Interaction Server application objects and add the media-types option with the value consisting of a comma-separated list of media types supported by the server. If this option is not present, it is assumed that Interaction Server can process any media type defined for the tenant. These options are not used by Interaction Servers, but by Interaction Server Proxies to correctly route interaction related requests to the servers in the cluster. Interaction Server Proxy does not use the list of tenants configured for it, but the list matters for its clients, which connect to the Proxy whose configured tenants match their own.
  - b. Create the Interaction Server cluster application using the Application cluster application type. In its **[cluster-settings]** section, set the **cluster-type** option to Interaction Server.
  - c. Add connections to all primary Interaction Servers you have created in Steps 1 and 2 to the connections of the new cluster application.
  - d. Create the Interaction Server Proxy cluster using the Application cluster application type. In its **[cluster-settings]** section, set the **cluster-type** option to Interaction Server Proxy.
  - e. Add connections to all primary Interaction Server Proxy applications you have created in Steps 3 and 4 to the Interaction Server Proxy cluster that you created in Step (d).
  - f. For each Interaction Server Proxy application added to the Interaction Server Proxy cluster in step (e), add a connection to the Interaction Server cluster created in step (b)
6. Add connections to the Interaction Server Proxy cluster to all applications that support Interaction Server Proxy cluster configuration (for example, Desktop). Genesys recommends that you not use

primary/backup deployment for Proxy applications that will be serving client applications that support Proxy clusters.

7. Add connections to the dedicated Proxy for applications that do not support Interaction Server Proxy cluster configuration. Genesys recommends that you use primary/backup deployment for Interaction Server Proxy applications that serve such clients.
8. If you are using URS (rather than ORS), add connections to all primary Interaction Servers to the URS application. URS should be directly connected to each Interaction Server in the cluster.
9. If you are using ORS, add a connection to Interaction Server Proxy to ORS application.
10. For the Stat Server application there are two options: it can be connected to Interaction Server Proxy or directly to each Interaction Server in the cluster.

## Other Information

### Snapshots

Starting with release 8.5.107.x, Interaction Server clusters support snapshots.

The number of interactions in a snapshot can be limited either by the number specified in the request or by the Interaction Server option max-interactions-per-snapshot. Note that these limits are per server, not per client snapshot, so the number of interactions returned in a client snapshot may be more than any specified limit.

### Timeout for Pull for Strategy

You can tune the way that the cluster responds to pull requests from routing clients.

- For each Interaction Server in the cluster, set a timeout using the **[settings]/pull-hold-timeout** option.
- Use the Proxy option of the same name to set a timeout that overrides the setting of any Interaction Server instance in the cluster.

Setting the Proxy option to a relatively low value can help speed up operation in situations in which one server has no interactions and the Proxy must wait for its response before going on to the next server in the cluster.

## Cluster Limitations

### Nodes Are Irreplaceable

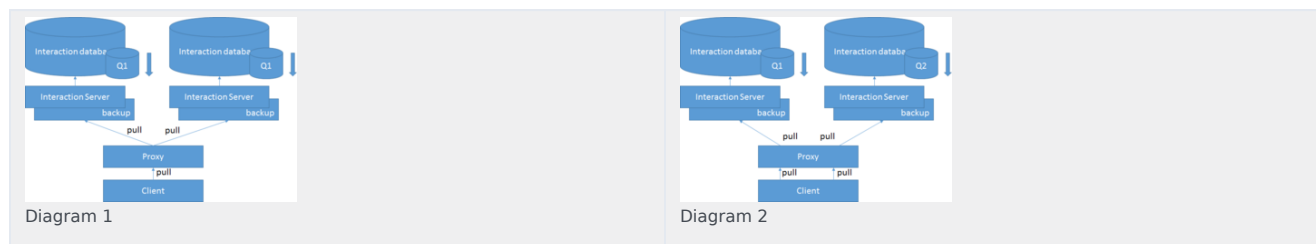
Interactions submitted to a given Interaction Server node cannot be processed by any other node.

Therefore, if you want redundancy of any kind, you have to deploy each node as a primary/backup pair, which gives you the level of reliability provided by the standard Genesys primary/backup functionality.

**Important:** A node for which both primary and backup Interaction Servers are down, or whose single node database is down, cannot process interactions, and no other nodes can complete any interactions that were in progress on this node. This also applies to the node's event log. If the event log database is not accessible for a significant time period, then events are lost.

### Interaction Queue FIFO Issue

Because each primary/backup pair works with its own database and the proxy requests interactions from the same queue in a round robin fashion, the order of interactions could be broken. This limitation is more important for long-lived interactions and large backlogs. For deployments with low volume and no backlog it is not very important.



In Diagram 1, two nodes handle the same type of interactions using the same business process and both handle the same queue Q1. Both servers maintain the ordering of the interactions, and respond to pull requests by the proxy in the correct order. But the order as seen by the client may not be correct, as the proxy has no way of tracking or handling the order correctly (it does take account of the semantics of order definition) and, therefore, cannot request the appropriate server (for example, to get older interactions or interactions with higher priority as defined by the interaction View order).

The same is true for URS, where each node pushes interactions from its own database in the correct order, but the order of routing of interactions by the URS is different. This also applies to the segmentation feature of an interaction View.

In certain situations, for example, when URS and interaction View use the same order by priority, URS can rectify the situation by processing interactions internally in the correct order, and, therefore, allowing one of the servers to push more interactions with higher priority.

If URS or ORS or any other client that pulls interactions from an interaction View does not impose its own order of processing, but relies solely on the order provided by Interaction Server (or, in this case, by the proxy), the imbalance between the servers may increase in time, because there is no synchronization point unless the queues become empty. In other words, the situation with broken queue order may be acceptable for short-lived interactions, like chat, and less acceptable for long living interactions, like email.

Obviously, if different nodes process different types of interactions, and as a result, only one server in the cluster maintains the order of some specific queue, there is no such problem. For example, in Diagram 2, the client pulls from different queues defined for different solutions and both streams are provided in the correct order.

## Duplicate Interaction IDs

Because clients can provide their own interaction IDs, it is possible that interactions with the same ID may be submitted to different nodes at some point when the cluster cannot check for duplicates. This issue does not arise if interaction identifiers are generated by the cluster itself (by any node in the cluster) or by eServices components (since they all follow a schema that provides uniqueness within a single configuration environment). But if customers wish to generate IDs themselves, they must guarantee that the IDs are unique.

## No Definite Reply to Some Operations Regarding Interaction Existence



Interaction is not accessible

If some node in the cluster is down (even for a short time, for example, due to switchover or host restart) the proxy cannot definitely tell if some interaction exists or not. In this case, the proxy generates a new error (Interaction is not accessible) when it cannot contact all the nodes in the cluster. In the figure at right, the first and second nodes provide negative replies, but the proxy receives no reply from the third node, and therefore the proxy cannot definitely tell the client that the interaction is not found.

In certain cases, a proxy might already have information about which server processed an interaction with a specific ID in its cache. In this case, the proxy only contacts the server that it has identified as having processed the interaction, and even if some nodes are down, it still may reply with the definitive answer (Interaction does not exist). But, generally, the proxy may restart or a new proxy can be instantiated, and in that case its cache is initially empty and it still needs to broadcast to all servers to try and find the interaction.

## Limited Scope for URS

Because URS is connected directly to each primary/backup Interaction Server in the cluster, it only sends requests (including ESP requests) to the server it received the EventRouteRequest from, and this limits the scope of these requests. For example, the ESP request FindInteraction can only access interactions in the context of a single node. However, in many situations this might be acceptable since the context of such URS requests is logically limited to the same scope.

## Increased Limits for Some Operations

For certain requests, for example, RequestFindInteractions, RequestGetWorkbinContent, that use configured limits for the maximum number of interactions that Interaction Server can return in request, those limits are not enforced in a cluster configuration. This is because, by design, the proxy forwards the same request to all relevant servers and then merges the replies. In the worst-case scenario, the number of returned interactions might be a multiple of the number of nodes.



In some specific scenarios, such as searching by external id, a short set of data from only one node may be sent back to the Desktop client.

However, if only one node handles a specific media type, then limits are enforced just as they are with a single Interaction Server deployment.

### Incorrect Order of Interactions in Some Requests

The EventWorkbinContent or EventInteractionsFound event, in cases when multiple servers return the results, contain interactions out of order after the proxy merges the results. This is because ordering and filtering are handled by the database and not by the server or proxy, which do not take account of the semantics of the condition and order. The desktop should sort the result in this case.

### Event Log Limitations

There are two options for configuring the **event logs** in the cluster:

1. Configure one Event Logger database for the entire cluster and let all servers in the cluster write to this single database.
2. Configure separate Event Logger databases for each primary/backup pair in the cluster and then process multiple databases at some point later.

The first option has some limitations:

- Since each server writes its own pair of events EventAgentLogin/EventAgentLogout, if some triggers are implemented in the database (including standard triggers that remove agent events on EventAgentLogout either with or without delay) then the first EventAgentLogout event removes all the event records while agents may still be logged in on other servers. This situation is rarely a problem however, since the proxy logs agents in on all servers and logs them out on all servers at the same time.
- If an ETL or any consumer of event logs is to handle agent sessions correctly, it must take into account the multiple overlapping pairs of EventAgentLogin/EventAgentLogout.

Note that since specific interactions are processed by a specific server, the issue of overlapping event pairs does not exist for interactions or ESP reporting events.

The second option requires event log consumers to process multiple databases and merge the event streams taking into account the same problem of overlapping agent state events.

### Licensing

- Each server in the cluster checks out the number of licenses specified in its configuration. Each primary/backup pair in the cluster uses its own unique license group ID (DBID of the primary server) and only shares the licenses between the pair.
- Distribution of licenses:
  - Different nodes do not share licenses and the entire cluster requires the number of licenses equal to the sum of all configured licenses for every primary/backup pair.
  - The lowest number of licenses that is specified in configurations of nodes defines the capacity of the

cluster. An unsuccessful login to one node (due to exceeded number of licenses) revokes that agent's logins from other nodes in the cluster.

As an example, suppose you have 100 available agent-seat licenses and four nodes. If you set **ics\_multi\_media\_agent\_seat** to 30 in every node, then three nodes will allocate 30+30+30 licenses, and the fourth node will allocate the remaining 10. Then with all nodes running, 10 will be the maximum number of agents who can log in to the Cluster.

- A dynamic decrease of licenses on one node may result in logout of agents from the cluster (if the new limit exceeds the number of agents currently logged on to this node).

## Capture Points

Capture Points are configured per node and the limitation is that each Capture Point can only access interactions that are processed by the node for which it is configured. That is certainly acceptable for a configuration in which nodes are configured to handle a specific media type/solution exclusively, and no two nodes handle the same media type. In this case, all Capture Point operations should work, including operations that request information from the server.

In configurations in which there are multiple nodes that handle the same media type and each node has a Capture Point configured, these Capture Points can be used to capture interactions and produce notifications towards external systems, provided they can work with the source in parallel. This might be true for JMS Capture Points that are configured to connect to the same JMS queues, for File Capture Points that are configured to monitor different directories, and for Web Service Capture Points for which some sort of load balancing is configured.

Because each Capture Point is limited to visibility of only one node, requests that are intended to request data from Interaction Server might not work correctly. For example, the request to Get Interaction/Task properties might be received by a Capture Point other than the one that created the interaction: the request would fail, even though the interaction might still exist in the database of the other node.

## Stat Server Java Extensions

In a cluster deployment, SSJE can only receive data from one Interaction Server.

## Proxy Disconnect/Mass Agent Logout/Panic Signal

The functionality implemented in Interaction Server to shield reporting clients from massive agent logout due to proxy disconnect does not work when a proxy application is connected to Interaction Server Proxy. The server is not notified of the disconnection of a proxy that is connected to Interaction Server Proxy. The server is only notified of individual disconnect events for each client that was connected to or handled by the disconnected proxy. Because of that, the server has no information on how many clients are disconnected and cannot tell when the threshold is reached that would generate the corresponding panic signal to reporting clients.

This can be mitigated by connecting Stat Server to all Interaction Server in the cluster directly. Stat Server has the same functionality of aggregating the reporting event streams from multiple servers as Interaction Server proxy does.

## ESP Configuration Must Be Symmetrical

All servers serving the same tenant and media type should have connections to the same ESP servers to avoid a situation in which one Interaction Server can execute a particular ESP request and another cannot.

## System Monitoring/Ping Request

The proxy does not support Ping requests. The components that monitor Interaction Server have to be connected directly to the individual Interaction Servers in the cluster.

## Disabled Media Type

Unlike Interaction Server, which permits submission of interactions by media server clients when a media type configuration object of the specified media type is disabled, Interaction Server Proxy does not allow submissions of such interactions.

## Interaction Server Application Type

Interaction Server Proxy connects only to Interaction Servers configured with the **Interaction Server application type**.

## Error Codes

Interaction Server Proxy handles some Agent state information internally, and may instantly respond to client requests. New error codes, applying to specific scenarios and conditions, have been introduced because of this change of behavior. Client applications must be adjusted so as to handle these error codes.

## Implementation Limitations

The proxy processes many requests by routing them to one or many Interaction Servers, based on data contained in the request. The proxy gets the tenant ID, media type, and interaction ID from the request (including ESP requests) in a generic way. Based on the values of these attributes, the proxy selects the Interaction Server.

In the following cases, Interaction Server Proxy (not Interaction Server) generates an error message:

- One of the attributes required by the proxy to route some specific request is not present.
- An attribute is provided with the incorrect type.
- An attribute is provided with an incorrect value.

Most of the time, the error message is the generic `Interaction is not accessible`.