### Cluster Node

**System > Cluster Node** is the screen for configuring the operational mode of the Logpresso server (node). When using two or more Logpresso servers to form a cluster, you can manage the communication settings of the nodes that make up the cluster.

#### Overview

Each server that constitutes the cluster is referred to as a node. A node is a general term for a host on a network or cloud. Here, a node refers to a single instance of the Logpresso server.

Classification by Role

The roles of the Logpresso server can be broadly divided into two categories:

**Collection Server**

The collection server is, quite literally, the server that collects logs or data. There are three main methods of collection:

* Receiving logs directly from managed servers or devices via network protocols such as Syslog or SNMP traps.
* Receiving logs collected and transmitted by agents (sentries) from managed servers.
* Accessing services such as FTP, SFTP, or databases provided by managed servers or network devices to collect data.

: Additionally, the collection server sends detected security events, either in real-time or periodically, to the analysis server.

**Analysis Server**

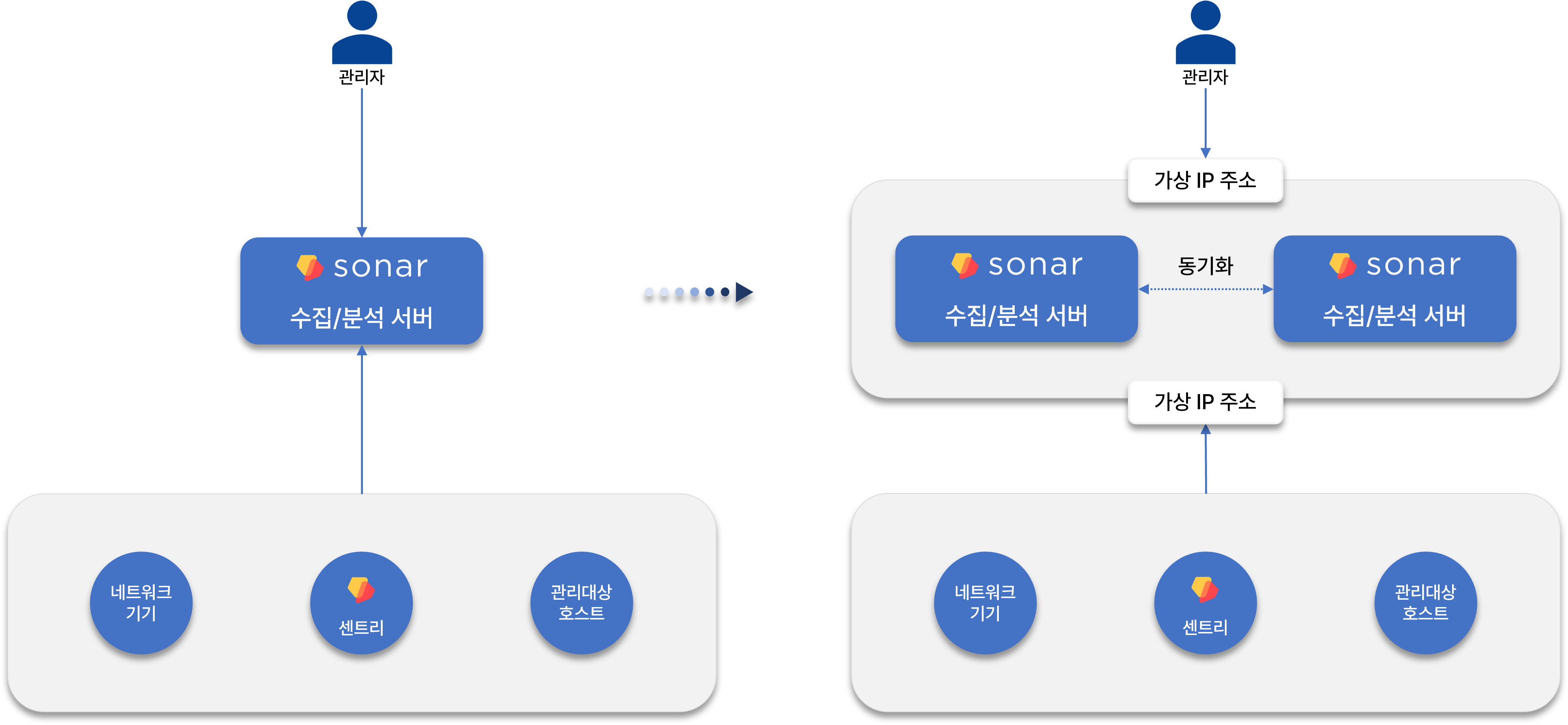
The analysis server processes various events and ticket data collected from the collection server. When a user accesses the web console of the analysis server to set policies, these policies are automatically synchronized with the collection server.

Standalone Configuration

In environments where the volume of logs to be collected is low, a single Logpresso server can perform the functions of both a collection server and an analysis server simultaneously. Based on Logpresso's reference hardware, a standalone server can collect up to 200GB of logs per day.

Redundant Configuration

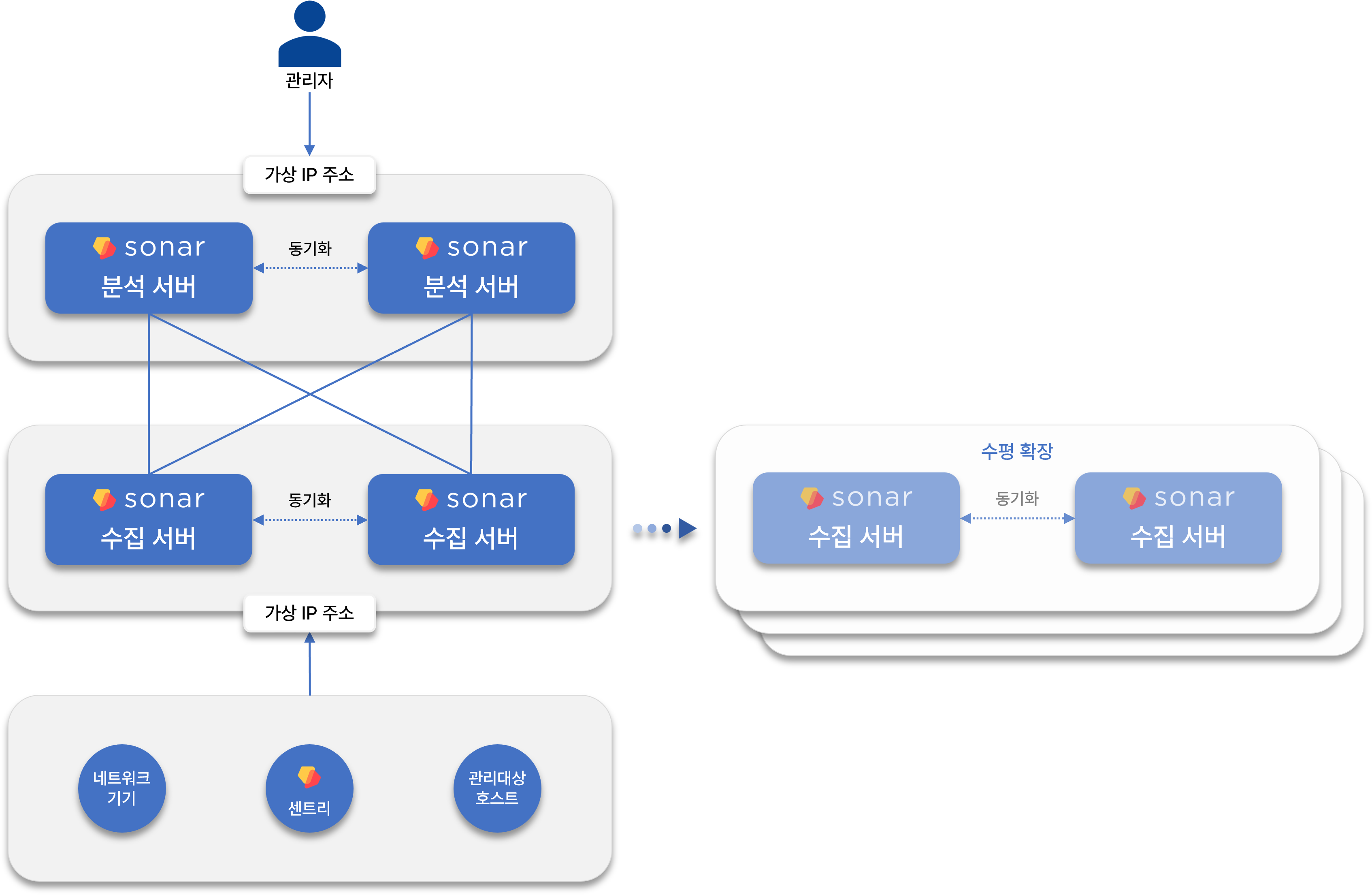
If service availability is critical, nodes can be configured redundantly. The simplest configuration is illustrated in the following diagram.



The left configuration represents a standalone node. Since both the collection server and analysis server roles are executed on a single server, it is also referred to as a collection-analysis server. This node can be made redundant to enhance service availability. In this case, it can handle the same daily log collection volume of 200GB based on reference hardware while improving availability.

Cluster Configuration

If the daily log collection volume exceeds 200GB, it is recommended to separate the roles of the collection server and analysis server and configure each redundantly. The collection server can be horizontally scaled as a single unit with a redundant configuration.



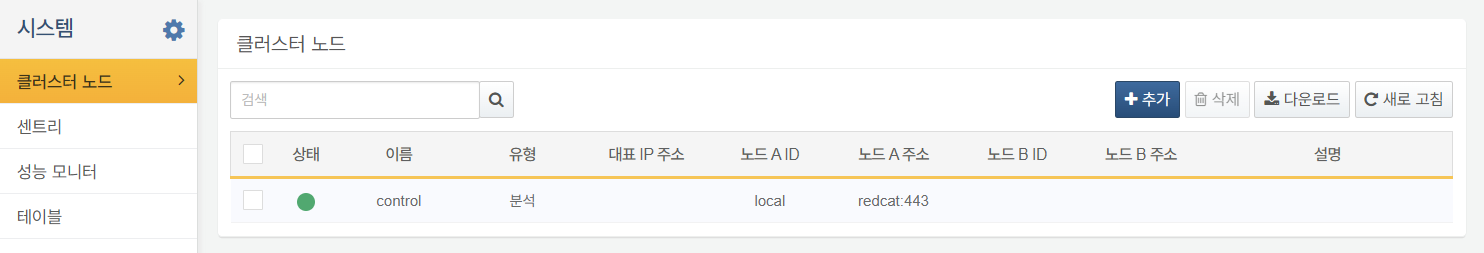
Network devices or managed hosts that send logs to the Logpresso server communicate by transmitting data to a virtual IP address (representative IP address) shared among the redundant nodes, with the active node among the redundant nodes processing the collected logs. Administrators also access the analysis server using the virtual IP address (representative IP address).

In a redundant configuration, data synchronization among nodes is achieved through the MariaDB Galera Cluster. The configuration method for the Galera Cluster is not covered in this documentation.

#### Node Management

Node Status Monitoring

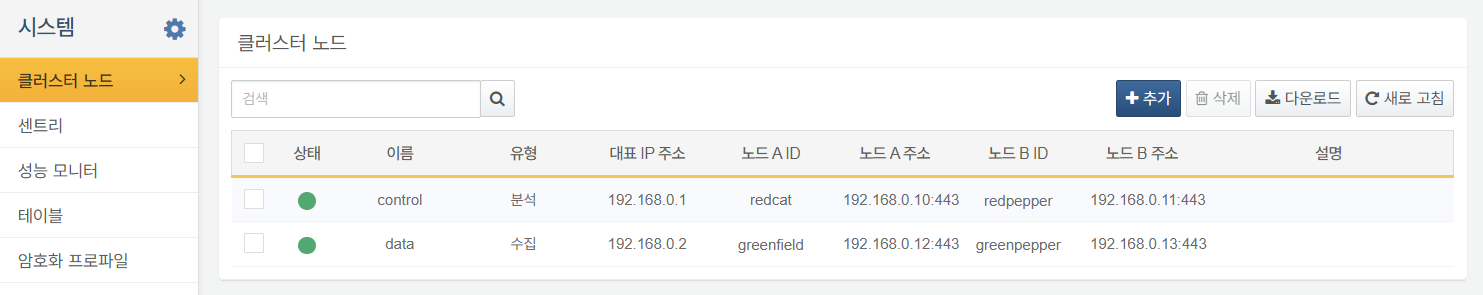
You can view the list of cluster nodes in **System > Cluster Node**. The following diagram shows a configuration where only **Node A** is registered, indicating a standalone operation.



The next diagram illustrates a redundant configuration with Node A and Node B. In this case, where the **Type** is **Analysis** and consists of **Node A** and **Node B**, both nodes perform the roles of collection server and analysis server simultaneously.



The following diagram depicts a cluster configured with redundant collection servers and redundant analysis servers.



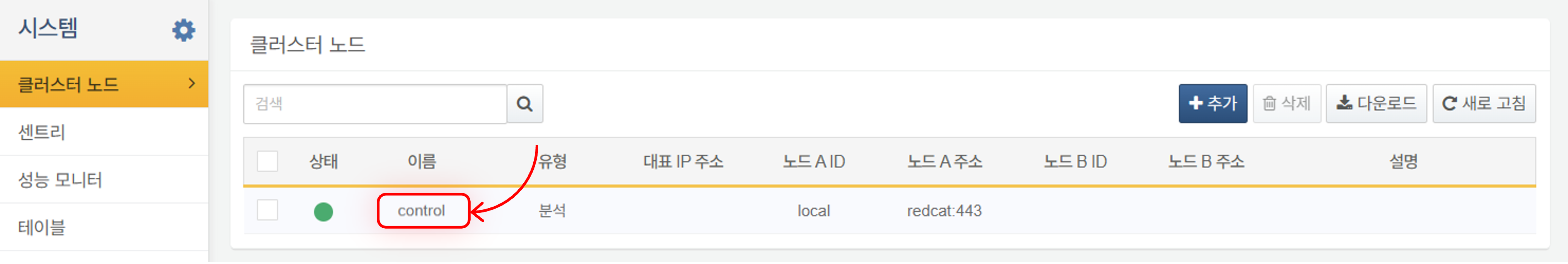
The information displayed in the cluster node list includes:

* **Status**: Connection status of the server (Green: Connected, Gray: Not Connected)
* **Name**: Name assigned to the redundant cluster node
* **Type**: Role of the server (**Analysis** or **Collection**)
* **Representative IP Address**: The virtual IP address shared among redundant nodes, such as Node A and Node B
* **Node A ID**: Identifier for Node A
* **Node A Address**: IP/domain address and port number for Node A
* **Node B ID**: Identifier for Node B
* **Node B Address**: IP/domain address and port number for Node B
* **Description**: Description of the cluster node

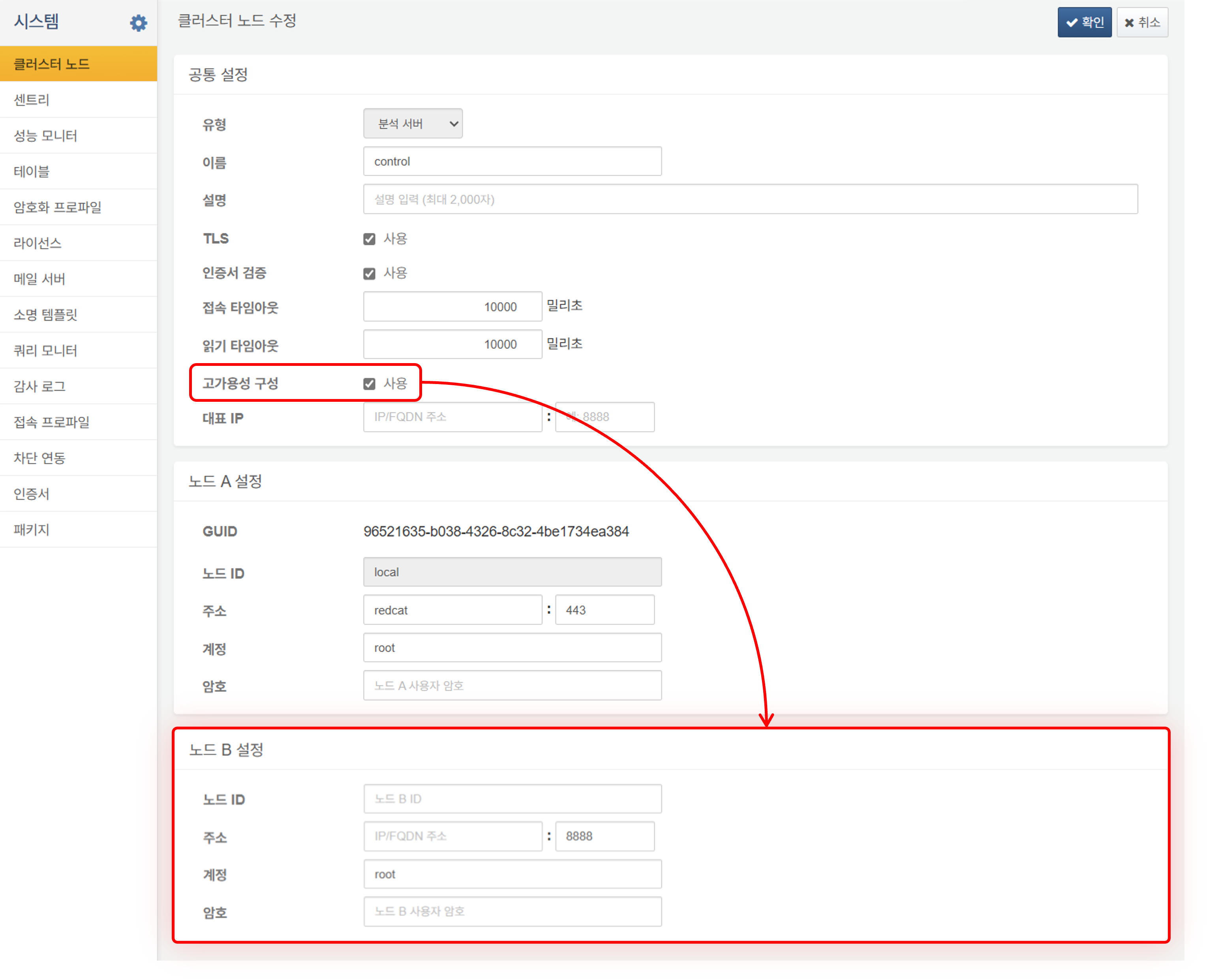
Node Redundancy

To create redundancy for a node, click on the name of the redundant configuration node and add Node B.

Click on the name of the cluster node to be made redundant.



In **Modify Cluster Node**, check **High Availability Configuration** under **Common Settings** to activate **Node B Settings**. Specify all the configuration properties required for node redundancy.



The properties to be configured in the cluster node modification are as follows:

**Common Settings**

Enter the identification information for the cluster and settings related to REST API access.

* **Type**: Role type of the cluster node (**Analysis Server** or **Collection Server**)
* **Name**: Unique name for the cluster node (default: **control**)
* **Description**: Description of the cluster node
* **TLS**: Whether to apply TLS for REST API communication between nodes (default: Enabled)
* **Certificate Validation**: Whether to validate the TLS certificate (default: Enabled). If using a private certificate, this checkbox should be unchecked.
* **Connection Timeout**: REST API communication connection timeout (default: 10,000 milliseconds)
* **Response Timeout**: REST API communication response timeout (default: 10,000 milliseconds). Since heartbeat packets are sent every 2 seconds, consider network communication delays and GC processing time when configuring.
* **High Availability Configuration**: Whether to enable node redundancy (default: Disabled)
* **Representative IP**: The virtual IP address shared by the redundant nodes

**Node A/B Settings**

Enter the unique settings for Node A and Node B that make up the cluster. **Node B Settings** can only be entered after checking **High Availability Configuration** in **Common Settings**.

* **GUID**: Automatically generated GUID for the node
* **Node ID**: Unique name for the node (default: **local**). It must consist of letters and numbers and be unique across the entire cluster.
* **Address**: Actual IP address or domain address of the node along with the communication port (default: 443)
* **Account**: REST API account ID for the node
* **Password**: REST API account password for the node

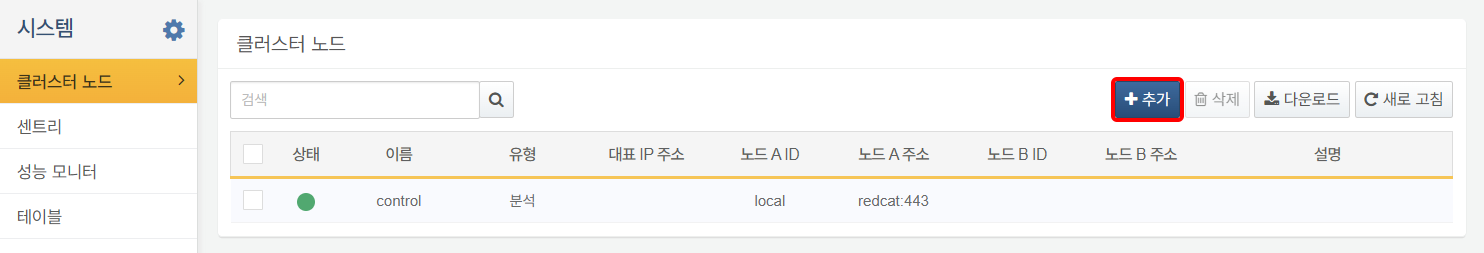
The initial cluster node configuration for the Logpresso server is standalone, and the Node ID "local" cannot be changed. To configure a cluster, you must delete the initial cluster node and reconfigure it. Both IP addresses and domain addresses are supported for the representative IP address and each node's address, but using IP addresses is advantageous considering network communication delays.

Verify that the entered settings are correct. Click **Confirm** to complete the node redundancy configuration. Click **Cancel** if you do not wish to configure redundancy.

Adding Cluster Nodes

To add a cluster node:

Click **Add** in the cluster node toolbar.



In the Add Cluster Node dialog, enter the necessary settings for the node configuration. Refer to **[Node Redundancy](section-cluster-nodes#outline9)** for the properties to be entered.

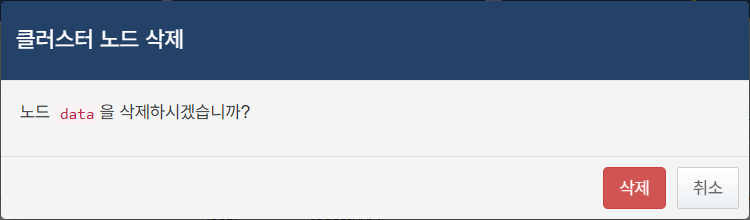
Deleting Cluster Nodes

To delete an added cluster node:

Ensure that there are no running loggers on the node you wish to delete. Check if there are any running [Loggers](https://docs.logpresso.comnull) on the node to be deleted.

If there are no running loggers on the node to be deleted, select the checkbox next to the **Status** of the node and click **Delete** in the cluster node toolbar.

In the **Delete Cluster Node** dialog, confirm the name of the node to be deleted. Click **Delete** to proceed with the deletion or **Cancel** if you do not wish to delete it.



When a cluster node is deleted, the data contained within that node, such as tables, is not deleted, but the associated metadata will be removed.