IceGrid Server Activation

On this page:

- Server Activation Modes
- Server Activation in Detail
- Requirements for Server Activation
- Efficiency Considerations for Server Activation
- Activating Servers with Specific User IDs
- Automating Endpoint Registration

Server Activation Modes

You can choose among four activation modes for servers deployed and managed by an IceGrid node:

- Manual
  You must start the server explicitly via the IceGrid GUI or `icegridadmin`, or programmatically via the `IceGrid::Admin` interface.

- Always
  IceGrid activates the server when its node starts. If the server stops, IceGrid automatically reactivates it.

- On demand
  IceGrid activates the server when a client invokes an operation on an object in the server.

- Session
  This mode also provides on-demand activation but requires the server to be allocated by a session.

Server Activation in Detail

On-demand server activation is a valuable feature of distributed computing architectures for a number of reasons:

- It minimizes application startup times by avoiding the need to pre-start all servers.
- It allows administrators to use their computing resources more efficiently because only those servers that are actually needed are running.
- It provides more reliability in the case of some server failure scenarios, e.g., the server is reactivated after a failure and may still be capable of providing some services to clients until the failure is resolved.
- It allows remote activation and deactivation.

On-demand activation occurs when an Ice client requests the endpoints of one of the server's object adapters via a locate request. If the server is not active at the time the client issues the request, the node activates the server and waits for the target object adapter to register its endpoints. Once the object adapter endpoints are registered, the registry returns the endpoint information back to the client. This sequence ensures that the client receives the endpoint information *after* the server is ready to receive requests.

Requirements for Server Activation

In order to use on-demand activation for an object adapter, the adapter must have an identifier and be entered in the IceGrid registry.

When using session activation mode, IceGrid requires that the server be allocated; on-demand activation fails for servers that have not been allocated.

The session activation mode recognizes an additional reserved variable in the server descriptor, `${session.id}`. The value of this variable is the user ID or, for SSL sessions, the distinguished name associated with the session.

Efficiency Considerations for Server Activation

Once a server is activated, it remains running indefinitely (unless it uses the session activation mode). A node deactivates a server only when explicitly requested to do so. As a result, server processes tend to accumulate on the node's host.

One of the advantages of on-demand activation is the ability to manage computing resources more efficiently. Of course there are many aspects to this, but Ice makes one technique particularly simple: servers can be configured to terminate gracefully after they have been idle for a certain amount of time.

A typical scenario involves a server that is activated on demand, used for a while by one or more clients, and then terminated automatically when no requests have been made for a configurable number of seconds. All that is necessary is setting the server's configuration property `Ice.ServerIdleTime` to the desired idle time.

Copyright © 2017, ZeroC, Inc.

For a server activated in session activation mode, IceGrid deactivates the server when the session releases the server or when the session is destroyed.

Activating Servers with Specific User IDs

On Unix platforms you can activate server processes with specific effective user IDs, provided that the IceGrid node is running as root. If the IceGrid node does not run as root, servers are always activated with the effective user ID of the IceGrid node process. (The same is true for Windows — servers always run with the same user ID as the IceGrid node process.)

For the remainder of this section, we assume that the node runs as root on a Unix machine.

The `user` attribute of the server descriptor specifies the user ID for a server. If this attribute is not specified and the activation mode is not `session`, the default value is `nobody`. Otherwise, the default value is `${session.id}` if the activation mode is `session`.

Since individual users often have different account names and user IDs on different machines, IceGrid provides a mechanism to map the value of the `user` attribute in the server descriptor to a user account. To do this, you must configure the node to use a user account mapper object. This object must implement the `IceGrid::UserAccountMapper` interface:

Slice

```
exception UserAccountNotFoundException {};

interface UserAccountMapper {
    string getUserAccount(string user)
            throws UserAccountNotFoundException;
};
```

The IceGrid node invokes `getUserAccount` and passes the value of the server descriptor's `user` attribute. The return value is the name of the user account.

IceGrid provides a built-in file-based user account mapper that you can configure for the node and the registry. The file contains any number of user-account-ID pairs. Each pair appears on a separate line, with white space separating the user account from the identifier. For example, the file shown below contains two entries that map two distinguished names to the user account `lisa`:

```
lisa O=ZeroC\\, Inc., OU=Ice, CN=Lisa
lisa O=ZeroC\\, Inc., OU=Ice, CN=Lisa S.
```

The distinguished names must be unique. If the same distinguished name appears several times in a file, the last entry is used.

You can specify the path of the user account file with the `IceGrid.Registry.UserAccounts` property for the registry and the `IceGrid.Node.UserAccounts` property for a node.

To configure an IceGrid node to use the IceGrid registry file-based user account mapper, you need to set the `IceGrid.Node.UserAccountMapper` property to the well-known proxy `IceGrid/RegistryUserAccountMapper`. Alternatively, you can set this property to the proxy of your own user account mapper object. Note that if this property is set, the node ignores the setting of `IceGrid.Node.UserAccounts`.

Automating Endpoint Registration

Servers must be properly configured to enable automatic endpoint registration. It should be noted however that IceGrid simplifies the configuration process in two ways:

- The IceGrid deployment facility automates the creation of a configuration file for the server, including the definition of object adapter identifiers and endpoints.
- A server that is activated automatically by an IceGrid node does not need to explicitly configure a proxy for the locator because the IceGrid node defines it in the server's configuration file.

See Also

- Getting Started with IceGrid
- IceGrid Architecture
- Resource Allocation using IceGrid Sessions
- Server Descriptor Element
- Locator Configuration for a Server

Copyright © 2017, ZeroC, Inc.

- Using IceGrid Deployment
- IceGrid Properties

Copyright © 2017, ZeroC, Inc.