Manually Installing a Service

This page describes how to manually install and configure an Ice service using the IcePatch2 service as a case study. For the purposes of this discussion, we assume that Ice is installed in the directory `C:\Ice`. We also assume that you have administrative access to your system, which is required by many of the installation steps discussed below.

On this page:

Selecting a User Account for the IcePatch2 Service

The IcePatch2 service can run in a regular user account, therefore we will follow our own recommendation and use the Local Service account.

Preparing a Directory for the IcePatch2 Service

The service needs a directory in which to store the files that it distributes to clients. A common mistake is assuming that a service will be able to access a file or directory that you created using your current account, which is likely to cause the service to fail in a way that is difficult to diagnose. To prevent such failures, we will ensure that the directory has the necessary permissions for the service to access it while running in the `Local Service` account.

Selecting a Directory for the IcePatch2 Service

The directory tree for our IcePatch2 service is shown below:

```
C:\Documents and Settings\
  LocalService\
    Local Settings\
      Application Data\
        ZeroC\
          icepatch2\
            data\
```

Note that this tree applies to Windows XP and Windows Server 2003, and is locale dependent. On Windows Vista (or later), we would use the following tree instead:

```
C:\Windows\
  ServiceProfiles\
    LocalService\
      AppData\
        Local\
          ZeroC\
            icepatch2\
              data\
```

For this example, we will use the Windows XP directory tree.

Creating the Directory for the IcePatch2 Service

Since `Local Service` is a built-in account, its user directory should already exist and have the proper access rights.

> If you open `C:\Documents and Settings` in Windows Explorer, the `LocalService` directory may not be visible until you modify your folder options to show protected files and folders.

If the directory does not exist, we can create it in a command window with the following steps:

```
> cd \Documents and Settings
> mkdir LocalService
```

At this point we could create the rest of the directory hierarchy. However, a newly-created directory inherits the privileges of its enclosing directory, and we have not yet modified the privileges of the `LocalService` directory to grant access to the `Local Service` account. At present, the privileges of the `LocalService` directory are inherited from `Documents and Settings` and require modification. In general, it is better to establish the necessary access rights on the parent directory prior to creating any subdirectories, so we will modify the `LocalService` directory first.

On all Windows systems, we can use the command-line utility `cacls`. The following command does what we need:

```
> cacls LocalService /G "Local Service":F Administrators:F
```

By omitting the `/E` option to `cacls`, we have replaced all of the prior access rights on the directory with the rights given in this command. As a result, the `Local Service` account and anyone in the Administrators group are granted full access to the directory, while all others are forbidden. (We grant full access to the Administrators group because presumably someone other than the `Local Service` account will need to manage the subdirectory, create the configuration file, an so on). You can verify the directory's current privilege settings by running `cacls` without options:

```
> cacls LocalService
```

Now we can create the remaining subdirectories, and they will automatically inherit the access rights established for the `LocalService` directory:

```
> cd LocalService
> mkdir "Local Settings\Application Data\ZeroC\icepatch2\data"
```

If you want to further restrict access to files or subdirectories, you can modify them as necessary using the `cacls` utility. Note however that certain actions may cause a file to revert back to the access rights of its enclosing directory. For example, modifying a file using a text editor is often the equivalent of erasing the file and recreating it, which discards any access rights you may have previously set for the file.

On some versions of Windows XP, and on Windows Server 2003 and Windows Vista (or later), you can manage privilege settings interactively using Windows Explorer. For example, right click on the `LocalService` directory, select Properties, and select the Security tab. Next select Advanced and Edit, uncheck "Include inheritable permissions from this object's parent," and select Copy. Remove all permission entries, then add entries for Local Service and the Administrators group and grant Full Control to each.

Populating the Directory for the IcePatch2 Service

Now you can copy the files that will be distributed to clients into the `data` subdirectory. The new files should inherit the access rights of their enclosing directory. For the sake of discussion, let's copy some Slice files from the Ice distribution into the data directory:

```
> cd "Local Settings\Application Data\ZeroC\icepatch2\data"
> copy \Ice\slice\Ice\*.ice
```

Next we need to run `icepatch2calc` to prepare the directory for use by the IcePatch2 service:

```
> icepatch2calc .
```

Configuration File for the IcePatch2 Service

IcePatch2 requires a minimal set of configuration properties. We could specify them on the service's command line, but if we later want to modify those properties we would have to reinstall the service. Defining the properties in a file simplifies the task of modifying the service's configuration.

Our IcePatch2 configuration is quite simple:

```
IcePatch2.Directory=C:\Documents and Settings\LocalService\
Local Settings\Application Data\ZeroC\icepatch2\data
IcePatch2.Endpoints=tcp -p 10000
```

The `IcePatch2.Directory` property specifies the location of the server's data directory, which we created in the previous section.

We will save our configuration properties into the following file:

```
C:\Ice\config\icepatch2.cfg
```

We must also ensure that the service has permission to access its configuration file. The Ice run time never modifies a configuration file, therefore read access is sufficient. The configuration file likely already has the necessary access rights, which we can verify using the `cacls` utility that we described earlier:

```
> cacls C:\Ice\config\icepatch2.cfg
```

Creating the IcePatch2 Service

We will use Microsoft's Service Control (`sc`) utility in a command window to create the service.

> See http://support.microsoft.com/kb/251192 for more information about the `SC` utility.

Our first `sc` command does the majority of the work (the command is formatted for readability but must be typed on a single line):

```
> sc create icepatch2 binPath= "C:\Ice\bin\icepatch2server.exe
  --Ice.Config=C:\Ice\config\icepatch2.cfg --service icepatch2"
  DisplayName= "IcePatch2 Server" start= auto
  obj= "NT Authority\LocalService" password= ""
```

There are several important aspects of this command:

- The service name is `icepatch2`. You can use whatever name you like, as long as it does not conflict with an existing service. Note however that this name is used in other contexts, such as in the `--service` option discussed below, therefore you must use it consistently.
- Following the service are several options. Note that all of the option names end with an equals sign and are separated from their arguments with at least one space.
- The `binPath=` option is required. We supply the full path name of the IcePatch2 server executable, as well as command-line arguments that define the location of the configuration file and the name of the service, all enclosed in quotes.
- The `DisplayName=` option sets a friendly name for the service.
- The `start=` option configures the start up behavior for the service. We used the argument `auto` to indicate the service should be started automatically when Windows boots.
- The `obj=` option selects the user account in which this service runs. As we explained, the `Local Service` account is appropriate for most services.
- The `password=` option supplies the password associated with the user account indicated by `obj=`. The `Local Service` account has an empty password.

The `sc` utility should report success if it was able to create the service as specified. You can verify that the new service was created with this command:

```
> sc qc icepatch2
```

Alternatively, you can start the Services administrative control panel and inspect the properties of the IcePatch2 service.

If you start the control panel, you will notice that the entry for IcePatch2 does not have a description. To add a description for the service, use the following command:

```
> sc description icepatch2 "IcePatch2 file server"
```

After refreshing the list of services, you should see the new description.

Creating the Event Log for the IcePatch2 Service

By default, programs such as the IcePatch2 service that utilize the `Service`) class log messages to the `Application` event log. Below we describe how to prepare the Windows registry for the service's default behavior, and we also show how to use a custom event log instead. We make use of Microsoft's Registry (`reg`) utility to modify the registry, although you could also use the interactive `regedit` tool. As always, caution is recommended whenever you modify the registry.

Using the Application Log for the IcePatch2 Service

We must configure an event log source for events to display properly. The first step is to create a registry key with the name of the source. Since the `Service` class uses the service name as the source name by default, we add the key `icepatch2` as shown below:

```
> reg add HKLM\SYSTEM\CurrentControlSet\Services\EventLog\Application\icepatch2
```

Inside this key we must add a value specifies the location of the Ice run time DLL:

```
> reg add HKLM\SYSTEM\CurrentControlSet\Services\EventLog\Application\icepatch2
/v EventMessageFile /t REG_EXPAND_SZ /d C:\Ice\bin\ice34.dll
```

We will also add a value indicating the types of events that the source supports:

```
> reg add HKLM\SYSTEM\CurrentControlSet\Services\EventLog\Application\icepatch2
/v TypesSupported /t REG_DWORD /d 7
```

The value `7` corresponds to the combination of the following event types:

- `EVENTLOG_ERROR_TYPE`
- `EVENTLOG_WARNING_TYPE`
- `EVENTLOG_INFORMATION_TYPE`

You can verify that the registry values have been defined correctly using the following command:

```
> reg query HKLM\SYSTEM\CurrentControlSet\Services\EventLog\Application\icepatch2
```

Our configuration of the event log is now complete.

Changing the Source Name for the IcePatch2 Service

Using the configuration described in the previous section, events logged by the IcePatch2 service are recorded in the event log using the source name `icepatch2`. If you prefer to use a source name that differs from the service name, you can replace `icepatch2` in the registry commands with the name of your choosing, but you must also add a matching definition for the property `Ice.EventLog.Source` to the service's configuration file.

For example, to use the source name `Ice File Patching Service`, you would add the registry key as shown below:

```
> reg add "HKLM\SYSTEM\CurrentControlSet\Services\EventLog\Application\Ice File Patching Service"
```

The commands to add the `EventMessageFile` and `TypesSupported` values must be modified in a similar fashion. Finally, add the following configuration property to `icepatch2.cfg`:

```
Ice.EventLog.Source=Ice File Patching Service
```

Using a Custom Log for the IcePatch2 Service

You may decide that you want your services to record messages into an application-specific log instead of the `Application` log that is shared by other unrelated services. As an example, let us create a log named `MyApp`:

```
> reg add "HKLM\SYSTEM\CurrentControlSet\Services\EventLog\MyApp"
```

Next we add a subkey for the IcePatch2 service. As described in the previous section, we will use a friendlier source name:

```
> reg add "HKLM\SYSTEM\CurrentControlSet\Services\EventLog\MyApp\Ice File Patching Service"
```

Now we can define values for `EventMessageFile` and `TypesSupported`:

```
> reg add "HKLM\SYSTEM\CurrentControlSet\Services\EventLog\MyApp\Ice File Patching Service"
/v EventMessageFile /t REG_EXPAND_SZ /d C:\Ice\bin\ice34.dll

> reg add "HKLM\SYSTEM\CurrentControlSet\Services\EventLog\MyApp\Ice File Patching Service"
/v TypesSupported /t REG_DWORD /d 7
```

Finally, we define `Ice.EventLog.Source` in the IcePatch2 service's configuration file:

```
Ice.EventLog.Source=Ice File Patching Service
```

Note that you must restart the Event Viewer control panel after adding the `MyApp` registry key in order to see the new log.

Registry Caching for the IcePatch2 Service

The first time a service logs an event, Windows' Event Log service caches the registry entries associated with the service's source. If you wish to modify a service's event log configuration, such as changing the service to use a custom log instead of the `Application` log, you should perform the following steps:

1. Stop the service.
2. Remove the unwanted event log registry key.
3. Add the new event log registry key(s).
4. Restart the system (or at least the Event Log service).
5. Start the service and verify that the log entries appear in the intended log.

After following these steps, open a log entry and ensure that it displays properly. If it does not, for example if the event properties indicate that the description of an event cannot be found, the problem is likely due to a misconfigured event source. Verify that the value of `EventMessageFile` refers to the correct location of the Ice run time DLL, and that the service is defining `Ice.EventLog.Source` in its configuration file (if necessary).

Starting the IcePatch2 Service

We are at last ready to start the service. In a command window, you can use the `sc` utility:

```
> sc start icepatch2
```

The program usually responds with status information indicating that the start request is pending. You can query the service's status to verify that it started successfully:

```
> sc query icepatch2
```

The service should now be in the running state. If it is not in this state, open the Event Viewer control panel and inspect the relevant log for more information that should help you to locate the problem. Even if the service started successfully, you may still want to use the Event Viewer to confirm that the service is using the log you expected.

Testing the IcePatch2 Service

Ice includes a graphical IcePatch2 client in the `demo/IcePatch2/MFC` directory of the Ice distribution. Once you have built the client, you can use it to test that the service is working properly.

See Also

- IcePatch2
- `Ice::Service` Class
- Installing a Windows Service