Filtering Proxy Endpoints

A proxy's configuration determines how its endpoints are used. For example, a proxy configured for secure communication will only use endpoints having a secure protocol, such as SSL.

The factory methods listed in the table below allow applications to manipulate endpoints indirectly. Calling one of these methods returns a new proxy whose endpoints are used in accordance with the proxy's new configuration.

Method	Description
`ice_secure`	Selects only endpoints using a secure protocol (e.g., SSL).
`ice_datagram`	Selects only endpoints using a datagram protocol (e.g., UDP).
`ice_batchDatagram`	Selects only endpoints using a datagram protocol (e.g., UDP).
`ice_twoway`	Selects only endpoints capable of making twoway invocations (e.g., TCP, SSL). For example, this disables datagram endpoints.
`ice_oneway`	Selects only endpoints capable of making reliable oneway invocations (e.g., TCP, SSL). For example, this disables datagram endpoints.
`ice_batchOneway`	Selects only endpoints capable of making reliable oneway batch invocations (e.g., TCP, SSL). For example, this disables datagram endpoints.

Upon return, the set of endpoints in the new proxy is unchanged from the old one. However, the new proxy's configuration drives a filtering process that the Ice run time performs during connection establishment.

The factory methods do not raise an exception if they produce a proxy with no viable endpoints. For example, the C++ statement below creates such a proxy:

C++

```
proxy = comm->stringToProxy("id:tcp -p 10000")->ice_datagram();
```

It is always possible that a proxy could become viable after additional factory methods are invoked, therefore the Ice run time does not raise an exception until connection establishment is attempted. At that point, the application can expect to receive `NoEndpointException` if the filtering process eliminates all endpoints.

An application can also create a proxy with a specific set of endpoints using the `ice_endpoints` factory method, whose only argument is a sequence of `Ice::Endpoint` objects. At present, an application is not able to create new instances of `Ice::Endpoint`, but rather can only incorporate instances obtained by calling `ice_getEndpoints` on a proxy. Note that `ice_getEndpoints` may return an empty sequence if the proxy has no endpoints, as is the case with an indirect proxy.

See Also

- Proxy Endpoints
- Proxy Methods
- Connection Establishment