Chapter 7
Name Service Multiplexor

Abstract

The Name Service Multiplexor (NSMux) redirects name service requests to the proper name service provider, thus allowing libraries and other NLMs to be written independent of any name service.

Contents

Introduction

The Name Service Multiplexor (NSMux) enables Client32 modules to be independent of a particular name service type, such as Bindery, NDS, or PNW. NSMux also provides registry functions that allow name service providers to register their services.

NSMux will generally be used by modules needing to resolve a NetWare name to either a network address or object ID. Network addresses are used by upper modules to build a connection to the supplied name for later operations.

The following terms will be used in the remainder of this document:

- *Name Service Multiplexor* - A module that provides the high-level name service interface to upper-level modules and then enumerates through the registered name service providers until the request is satisfied by one of them.

- *Name Service Provider* - A module that implements the name service interface for a particular name service type. A name service provider understands the specifics needed to resolve names in a particular name service's name space. In addition, the name service provider allows a caller to configure a preferred name in its name space.

The following diagram shows NSMux and two name service providers, Bindery NS and NDS NS, along with the Client32 Requester NLMS:

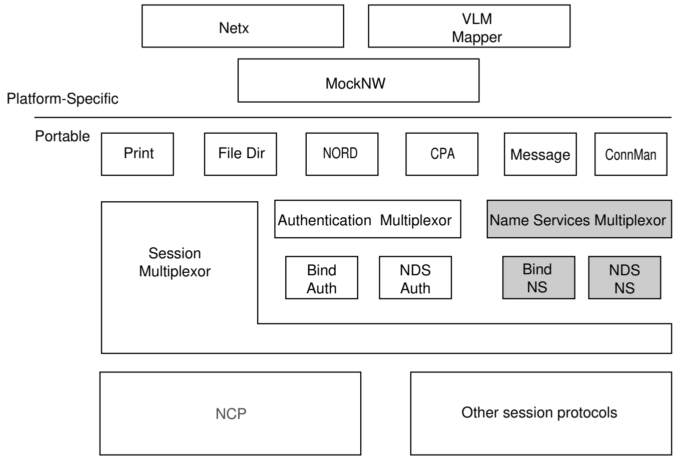

Figure 1. The Name Services Multiplexor routes requests to registered name service providers.

Requirements

The name service multiplexor and name service providers must be portable, fast, and well-defined so that any new name service provider can adapt to it. In addition, the multiplexor must allow the enumeration order to be configured, and must dynamically register and unregister name service providers.

Design Description

NSMux has two main functions: name service provider registry and multiplexing.

Registry

The name service multiplexor is a central registry with which name service providers can register their services. An API is defined for registry. There are currently three APIs defined:

NSMRegisterNameSvc Used by a name service provider to register its services when the provider is loaded.

NSMUnregisterNameSvc Used by a name service provider to unregister its services when the provider is unloaded.

NSMEnumerateNameSvc Used by a caller to enumerate the registered name service providers.

When a name service provider registers with NSMux, it passes a structure called *NAME_SVC_API_SET_TYPE*. This structure contains pointers to name-service-specific functions that the NSMux will call when it receives name service requests. The NSMux will allocate a structure for each supported name service provider.

Note: If a name service provider unloads, it must unregister with the NSMux so that NSMux does not continue using that provider's function calls.

Multiplexing

When the name service multiplexor gets a request from higher level modules, it either funnels the request to the specified name service provider, or enumerates through the registered name service providers until either one of them satisfies the request or the request is failed by all loaded providers.

Enumeration Order

Initially, the order in which NSMux calls the name service providers is the order in which name service providers registered with the mulitplexor. When a name service provider registers its services, it is placed at the end of the enumeration list and is called last to resolve any given request. But because names take time to resolve, a configuration option will be provided to allow a specified enumeration order instead of the default registration order.

Object IDs, when combined with a connection, uniquely identify objects in the name space of a particular name service provider. The APIs to resolve a name to address or a name to an object ID are **NSMResolveNameToAddress** and **NSMResolveObjectToID**. The name service multiplexor will enumerate through the registered name service providers trying to complete the request.

A parameter to these enumeration APIs is the connection handle to use when resolving a name. This allows the caller to influence the resolution process if there is prior knowledge of where the name might be located. If a NULL connection handle is passed, the name service provider should use the "preferred" connection, if one has been configured, to resolve the name.

In addition to the resolution APIs described above, a name service provider can also be configured for a preferred name of its name service type. The APIs to do this are **NSMGetPreferredName** to retrieve the currently configured name and **NSMSetPreferredName** to configure the preferred name.

Examples would be a preferred server name for the Bindery name service provider, and a preferred tree name for the NDS name service provider. The name service multiplexor for these API calls will simply route them to the specified name service multiplexor; no enumeration will occur.

The following is the complete list of functions (along with a short description) that the name service multiplexor must implement. All these functions except for the registry services must also be implemented by each name service provider.

Name Service Multiplexor API

NSMRegisterNameSvc	Registers a name service provider with the name service multiplexor.
NSMUnregisterNameSvc	Unregisters a name service provider with the name service multiplexor.
NSMEnumerateNameSvc	Allows the caller to enumerate the registered name service providers.
NSMResolveNameToAddress	Resolves a name in a particular name space to a transport address(s).
NSMResolveObjectToID	Resolves an object in a particular name space to a unique identifier and connection handle.
NSMGetPreferredName	Gets a previously-stored preferred name for a registered name service provider. No enumeration will occur; instead, a request is dispatched to the specified name service provider.
NSMSetPreferredName	Sets a preferred name for a name service provider to use when resolving name service requests. No enumeration will occur; instead, a request is dispatched to the specified name service provider.

Configuration

The name service enumeration order can be configured both statically and at runtime. The following keyword and enumeration order will be read from NET.CFG by the name service multiplexor at startup.

```
NAME SERVICE ENUM ORDER =  nameSvcProv  nameSvcProv ...
nameSvcProv
```

Though this option is mainly for improving performance in resolving names, it also helps the user resolve names that conflict across various name spaces. For example, if "JIMBOX" existed in the Directory Services name space and also in a Personal NetWare name space, this option allows a user to specify which name service provider should resolve the name.

Performance

Since the name service multiplexor passes requests on to name service providers, there is no way to measure throughput. The name service multiplexor must be coded efficiently so that it can quickly enumerate requests to name service modules for resolution.

Deliverables

The following are the deliverables for the name service multiplexor:

Documents:

NSMUXT.WP Unit test plan that describes tests and their expected results that the NsMux must pass before being accepted for cross-platform integration testing.

Product Executables:

NSMUX.NLM NLM executable that implements this design specification.

Product Source:

NSMUX.C C code that implements this design specification

NAME_SVC.H Header file that defines constants/structures defined in this design document.

NSMUX.MAK Makefile to build NSMUX.NLM

NSMUX.DEF Definition file that defines link/build information for NSMUX.NLM

NSMUX.IMP File that lists external functions required by NSMUX.NLM.

NSMUX.EXP File that lists functions that NSMUX will export for other modules to use.

Unit Test Executables:

NSMUXT.NLM NLM application to test the NSMUX module with.

Unit Test Source:

NSMUXT.C C code that unit tests the NSMUX module.

NSMUXT.MAK Makefile to build NSMUXT.NLM module.

NSMUXT.DEF Definition file that defines link/build information for NSMUXT.NLM.

NSMUXT.IMP File that lists external functions required by NSMUXT.NLM.