Chapter 14
NDS Licensing

Abstract

The 32-bit Client uses a new strategy for licensing NDS connections. The new algorithm views the network as a collection of services that can be individually licensed, rather than as a collection of servers which are licensed.

Introduction

The Client32 Requester takes an entirely new view of licensing NDS connections, tying licensing to specific services (for example, file or print services) rather than to server connections.

History

In the past, licensing was handled by APIs and was done on a server-by-server basis--if a user authenticated to a server, then that user was licensed to use all of the services available on that server, such as file and print services.

This licensing scheme presented three main problems. First, there was no realistic way to know when a user was finished with a connection. If a user printed something in the morning and then remained logged on but inactive for the rest of the day, the license would remain intact, making it impossible for any other user to use that license.

Second, there is no way to separately license different types of services. For example, a third-party vendor might want to license a certain type of service on a server (such as a CD-ROM library) and may wish to allow a different number of users than the number on the NetWare license. For example, the CD-ROM license may be for 100 users, while the NetWare license is for only 10. Unless licensing is tied to specific services and not to servers, the CD-ROM would be limited to ten users.

Third, with the advent of Directory Services, it had to be possible to authenticate to a server without a license, since DS services are free.

New Scheme

The new licensing scheme views a network as a group of services. The act of authenticating to a specific service takes up a license on that service. For example, authenticating to the file system would take up a license on a NetWare server since file services are a service for which NetWare limits the number of users. But authenticating to Directory Services, since it is defined as having unlimited access, would not take up a license.

This view makes client software aware of *how* a connection is used, allowing the client to determine when to tell the server to begin charging the user (by occupying a license).

Although applications view this scheme as authenticating to each service, in reality the underlying authentication method is shared by all services. This makes it so a user wouldn't have to enter a password for each service. Note also that applicatons will never have to deal with the concept of licensing.

Algorithm

Backward Compatibility

The Client32 Requester will continue to support Netx and VLMs in the following ways:

Netx

There are two ways to open a file with Netx: with a raw Request/Reply, and with a DOS Int21. If Netx receives an Int21, it will call **CheckIfLicenseNeeded**, and, when the connection is licensed, issue a 32-bit API to FileDir to open the file.

If a raw Request/Reply is used, the call is trapped by MockNW and checked to see if the call is an NCP that needs licensing (see Appendix A for a list of NCPs that need licensing). If it is an NCP requiring licensing, MockNW will call **NDSLicenseConnection** to establish the connection.

VLMs

Licensing. There are three VLMs that serve to license a connection: **TaskLockConnection**, **GoLicense**, and **IncHardResourceCount**. If any of these VLMs are used, the VLM Mapper will trap the call and issue a **CheckIfLicenseNeeded** call, determining whether the connection is already licensed.

The pseudocode for **CheckIfLicenseNeeded** is:

```
IF  (user is already authenticated to server)
   IF  (this connection is not already licensed)
      Call NDSLicenseConnection
   ELSE
      return success
ELSE
   return fail
```

Unlicensing. There are three VLMs that serve to unlicense connections: **GoUnlicense**, **TaskUnlockConnection**, and **DecHardResourceCount** (when it goes to 0). Each of these calls is ignored.

New Method

The new algorithm for dealing with connections is to offer 32-bit APIs which specifically license and unlicense specific services. This solves the problem of knowing when to unlicense, as well as the problem of allowing different services to license different numbers of users.

When the 32-bit APIs are not used, all licensed connections and services (including raw NCP calls, Int-21, VLMs) whether Bindery or DS will be set up and removed as follows:

Call received requesting authentication to some service .

Call is made to **NDSLicenseConnection.**

The connection remains licensed until the connection is LRU'd by ConnMan.

Modules Affected

VLM Mapper

Maps all license-related calls to **CheckIfLicenseNeeded**. See *Backward Compatibility* discussion above.

NWCalls

Instead of calling **Go License**, NWCalls will now use **NDSLicenseConnection**.

Print

Print produces events related to print services. The events produced by Print are:

Event_PrintDeviceOpen
Event_PrintDeviceClose

FileDir

FileDir produces events related to files, directory handles, search contexts, and semaphores. Creating these events allows NDS to monitor the usage of the file, directory and sempahore services, and to determine when to close a license.

The events created by FileDir are:

Event_FileOpenCreate
Event_FileCloseAbort
Event_DriveCreate
Event_DriveDestroyed
Event_SemaphoreOpenCreate
Event_SemaphoreCloseAbort
Event_SearchContextOpen
Event_SearchContextClose

NDS

NDS listens for NESL events, and determines when to license and unlicense.

MockNW

Receives raw Request/Reply calls, checks for NCPs that require a license, called **NDSLicenseConnection** if necessary. See *Backward Compatibility* discussion above.