Licensing Toolkit User's Guide

Version 9.1.5

Printed: April 2009

DCN LK-01-9105

Synergex
2330 Gold Meadow Way
Gold River, CA 95670 USA

http://www.synergex.com
phone 916.635.7300
fax 916.635.6549

Contents

Licensing Toolkit

Welcome to Licensing Toolkit

Synergy/DE™ Licensing Toolkit consists of the Licensing Toolkit API and the License Key Generator utility:

- The Licensing Toolkit API, which is distributed with Core Components, enables you to build Synergy/DE License Manager protection into your own Synergy and *xf*NetLink applications. The API includes both Synergy routines and a C API, **syncli_api.dll**, which contains three functions that can be called from a C, C++, .NET, or any other application that can call a DLL.
- The License Key Generator utility is used to generate configuration keys for your customers so that they may use your application. This utility is sent to you by Synergex™ (along with this documentation) when you request the product.

This document explains how to use Licensing Toolkit to secure your applications with License Manager. License Manager controls the installation and use of Synergy applications. When an application has been secured with License Manager, the application will not run unless License Manager is installed and configured to recognize it. You can configure an application to allow a maximum number of concurrent users, to secure specific components, and to run for a limited amount of time.

Licensing Toolkit can also be used to implement a reporting routine that can be used to satisfy license usage requirements for *xf*ServerPlus applications, as described in the Synergex License Agreement Terms and Conditions. See your Synergy/DE account manager for details on the reporting requirements.

Additional resources

- The "Configuring License Manager" chapter of the *Installation Configuration Guide* for information about License Manager, configuration keys, and Synergy Key files.
- **Licensing_TK_Examples.zip**, available from Synergy CodeExchange in the Resource Center on the Synergex web site. In addition to **lmkx.exe**, the interactive interface to the License Key Generator, this file includes other routines that illustrate the use of Licensing Toolkit. See the readme file included in the download for details.

Securing Synergy Applications with License Manager

Synergex will assign a producer code to your company, which you will need when using the Licensing Toolkit API.

Follow these steps to secure a Synergy application with License Manager.

1. Install the License Key Generator utility on a Windows machine and then install the configuration key you received from Synergex.
2. Create an application code for each application you want to secure with License Manager.

   The application code is passed when you make calls to LM_LOGIN and LM_LOGOUT. It is also used when you generate configuration keys. The application code can have a maximum of six alphanumeric characters.
3. Include a call to the LM_LOGIN subroutine at the beginning of the application you want to secure.

   LM_LOGIN calls License Manager and returns a status value and configuration information about the log-in. See page 5 for more information.
4. Include a call to the LM_LOGOUT subroutine at all exit points in your application.

   LM_LOGOUT frees the current user process from the tally of concurrent log-ins for the application and returns a status. See page 7 for more information.
5. Write a subroutine to handle your security, including any forced exit resulting from a security failure. Although LM_LOGIN and LM_LOGOUT return status values, these values do not affect License Manager. Your security subroutine must include code to handle the returned status.

You can use the .NODEBUG compiler directive to prevent debugging of this subroutine. The .NODEBUG directive deactivates debugging for modules that you don't want anyone to look at. You can place it anywhere in your subroutine. For more information, see .NODEBUG in the "Preprocessor and Compiler Directives" chapter of the *Synergy Language Reference Manual*.

6. (Recommended) Add an application-level TRY-CATCH-FINALLY statement to your mainline application module to ensure that the license is logged out if an untrapped error occurs.
7. (Recommended for Windows) Add code to log errors to the Windows event log.
8. After your application is installed at a customer site, request that the customer send you the licensee name and registration string.
9. Generate configuration keys for your customers, using the licensee names and registration strings they send you. See "Generating Configuration Keys" on page 15.
10. Send the configuration keys to your customers so that they can configure License Manager to run your applications.

LM_INFO – Return information about the system

```
xcall lm_info(lm_stat, lm_site)
```

Arguments

lm_stat

Returned with the License Manager site status. This will be LMSTS_OK (0) or one of the codes listed in the table on page 20. (**n**)

lm_site

Returned with the site information record: (**a**)

```
                ,a6   Blanks
lm_licensee     ,a50  Licensee name entered during registration
lm_regstr       ,a12  Registration string
lm_regdat       ,d8   Registration date (YYYYMMDD)
lm_timout       ,d8   Pre-install time-out in days (added to lm_regdat)
```

Discussion

The LM_INFO subroutine returns License Manager information about the system on which your application is running.

The status value returned in *lm_stat* is for checking success or providing an application-related license error message. This value has no effect on License Manager; you must write code to handle the returned status.

You can access system error codes with %SYSERR, which may assist in troubleshooting.

Examples

```
xcall lm_info(myStat, mySiteInfo)
```

See also **lm_auth.dbl** in **Licensing_TK_Examples.zip**, available from Synergy CodeExchange in the Resource Center on the Synergex web site.

LM_LOGIN – Log in and return a unique token and configuration information

```
xcall lm_login(app_stat, producer, application, token, lm_record, slot_num, [check]
&                [, seat])
```

Arguments

app_stat

Returned with the log-in status: (**n**)

0	LMSTS_OK	Product has been configured.
1	LMSTS_NOCONFIG	Product has not been configured.
2	LMSTS_MAXUSR	Concurrent user maximum reached.
7	LMSTS_TIMEOUT	Product demo has expired.

See the table on page 20 for additional codes that could be returned.

producer

The producer code for your company, provided by Synergex. (**a**)

application

The code that you created to refer to the application. It can be up to six characters long. (**a**)

token

Returned with the log-in token, which is a key that is maintained in License Manager. (**i4**)

lm_record

Returned with the information that was specified when the application was configured with License Manager: (**a**)

```
lm_applic ,a6    ;Application code
lm_usrmax ,d4    ;Maximum number of concurrent users
lm_expdat ,d8    ;Time-out date (YYYYMMDD)
lm_insdat ,d8    ;Installation date (YYYYMMDD)
lm_custom ,a100  ;Developer-defined information
```

slot_num

Returned with the slot number, which ranges from 1 to the number of users for which the product is configured. Slots are used as they become available; consequently, the slot number will not necessarily tell you the current number of users. (**n**)

check

(optional) Pass 0 (zero) to log in the license. Pass a non-zero value to check that the license exists. When *check* is non-zero, License Manager does not request a log-in slot, and the concurrent number of users is not incremented. (**n**)

seat

(optional) A 32-bit integer set to 0 (zero) or the value designated to represent the seat. (**i**)

Discussion

The LM_LOGIN subroutine requests a log-in "slot" from License Manager for the specified application or performs a license check (when *check* is non-zero), and then returns configuration information about the log-in.

For security reasons, we recommend that you encode the strings for producer code and application code in your routines.

When *check* is 0, the call to LM_LOGIN returns a token, which should be saved by your application and used in the call to LM_LOGOUT. License Manager will only recognize log-outs with a matching token. You need to retain the token only while the license is logged in, as a token has meaning only for the log-in for which it was obtained.

When using the *seat* argument, the value you should pass depends on the type of application. For a stand-alone application, where the user is running the application interactively from the same workstation that the user is logged in to, pass 0 for *seat* or just don't pass *seat* at all. In a server environment where there are multiple processes (such as an *xf*NetLink–*xf*ServerPlus application), you will need to decide how a "seat" should be represented within your application. It can be any 32-bit integer. The seat should be unique among seats, but the same for each instance of a particular seat. For example, you may want to use the client's IP address. Including the seat in these circumstances will enable you to log out the correct license.

The status value returned by LM_LOGIN in *app_stat* is for checking success or providing application-related license error messages. This value has no effect on License Manager; you must write code to handle the returned status. For example, if LM_LOGIN returns LMSTS_MAXUSR (2), you could deny the last user access to your application and present an error message stating the problem. You can access system error codes with %SYSERR, which may assist in troubleshooting.

Examples

See **lm_auth.dbl** in **Licensing_TK_Examples.zip**, available from Synergy CodeExchange in the Resource Center on the Synergex web site.

LM_LOGOUT – Log out a previously logged in license

`xcall lm_logout(`*lo_stat, producer, application, token[, seat]*`)`

Arguments

lo_stat

Returned with the log-out status: (**n**)

0	LMSTS_OK	Success.
3	LMSTS_BADTOK	Invalid token.

See the table on page 20 for additional codes that could be returned.

producer

The producer code for your company, provided by Synergex. (**a**)

application

The code that you created to refer to the application. (**a**)

token

The token obtained from the LM_LOGIN call (see page 5) and used here to log out. (**i4**)

seat

(optional) A 32-bit integer set to the value that was passed in the *seat* argument to the LM_LOGIN call. (**i**)

Discussion

LM_LOGOUT is used only when you have called LM_LOGIN with *check* set to 0 or not specified.

The LM_LOGOUT subroutine doesn't log out a process in the usual sense (that is, it does not prevent the application from continuing); rather, it requests that License Manager release the "slot" for the specified license so that another user can log in and use the available slot.

The token returned by the call to LM_LOGIN should be saved by your application and then passed to LM_LOGOUT. The token ensures that the process that is logging out is authorized to do so.

If you passed a non-0 value for *seat* in the call to LM_LOGIN, pass the same value for *seat* in the call to LM_LOGOUT. The seat, in combination with the token, ensures that the correct license is logged out in a server environment. If you pass *seat* with LM_LOGIN and fail to pass the same *seat* value with LM_LOGOUT, the results could be unpredictable.

On Windows, you must call LM_LOGOUT as part of your exit procedure, or the license will stay in use until the license server logs it out; this could take as long as 20 minutes. You should also call LM_LOGOUT in case of error.

On UNIX, although LM_LOGOUT is not required (normally licenses are logged out automatically when the runtime process has terminated), we recommend that you call LM_LOGOUT as part of your normal exit procedure, especially if your application STOP chains to another program. You should also call LM_LOGOUT in case of error.

The status values returned by LM_LOGOUT are for your information only; they have no effect on License Manager. You should write code to handle the returned status. You can access system error codes with %SYSERR, which may assist in troubleshooting.

Securing *xf*NetLink Applications with License Manager

Synergex will assign a producer code to your company, which you will need when using the Licensing Toolkit API.

Follow these steps to secure *xf*NetLink .NET or *xf*NetLink COM applications with License Manager. (The syncli_api DLL is not currently supported on *xf*NetLink Java.) These instructions presume you are using *xf*NetLink, but the functions in **syncli_api.dll** can be called from a C, C++, .NET, or any other application that can call a DLL.

1. Install the License Key Generator utility on a Windows machine and then install the configuration key you received from Synergex.
2. Install *xf*NetLink, which includes **syncli_api.dll**. This DLL contains the functions you will call from your client application to implement licensing.
3. Create an application code for each application you want to secure with License Manager.

   The application code is passed when you make calls to win_lm_login and win_lm_logout. It is also used when you generate configuration keys. The application code can have a maximum of six alphanumeric characters.
4. Include a call to the win_lm_login function at the beginning of the application you want to secure.

   The win_lm_login function calls License Manager and returns a status value. See page 11 for more information.
5. Include a call to the win_lm_logout function at all exit points in your application.

   The win_lm_logout function frees the current user process from the tally of concurrent log-ins for the application and returns a status. See page 13 for more information.
6. Write a subroutine to handle your security, including any forced exit resulting from security failure. Although win_lm_login and win_lm_logout return status values, these values do not affect License Manager. Your security subroutine must include code to handle the returned status.
7. (Recommended) Add an application-level try-catch-finally statement to your code to ensure that the license is logged out when an untrapped error occurs.
8. (Recommended) Add code to log errors to the Windows event log.

Deployment

This section describes what to do when you are ready to deploy your secured *xf*NetLink application at a customer site.

1. Set up a Synergy license server on a Windows machine at the customer site.
2. Install *xf*NetLink and your application on the client machine. This can be the same machine as the license server or it may be a different machine.

3. On the *xf*NetLink machine, run **lmu.exe** (included in the *xf*NetLink installation) to initialize License Manager as a license client to your license server machine:

   `lmu -c`*`server_name`* `-nc`

4. Run **lmu -b** on the license server machine to get the registration string.

5. Use the licensee name and registration string to generate configuration keys. See "Generating Configuration Keys" on page 15.

6. Install the keys on the customer's license server machine. This configures License Manager to run your application.

win_lm_login – Log in and return a unique token

```
int WINAPI win_lm_login(char *producer, char *application, int check, int seat,
                        int *token, int *syserr)
```

Arguments

producer

The producer code for your company, provided by Synergex. This is a null-terminated 8-bit ANSI string.

application

The code that you created to refer to the application. This is a null-terminated 8-bit ANSI string and can be up to six characters long.

check

An int32 set to 0 (zero) or non-zero. Pass 0 to log in the license. Pass a non-zero value to check that the license exists. When *check* is non-zero, License Manager does not request a log-in slot, and the concurrent number of users is not incremented.

seat

An int32 set to 0 (zero) or the value designated as representing the seat.

token

An int32 pointer returned with the log-in token, which is a key that is maintained in License Manager.

syserr

An int32 pointer returned with a system error code when the return status is other than 0.

Discussion

The win_lm_login function requests a log-in "slot" from License Manager for the specified application or performs a license check (when *check* is non-zero).

When *check* is 0, win_lm_login returns a token, which should be saved by your application and used in the call to win_lm_logout. License Manager will only recognize log-outs with a matching token. You need to retain the token only while the license is logged in, as a token has meaning only for the application log-in for which it was obtained.

When using the *seat* argument, the value you should pass depends on the type of application. For a stand-alone application, where the user is running the application interactively from the same workstation that the user is logged in to, pass 0 for *seat*. In a server environment where there are multiple users (such as an *xf*NetLink–*xf*ServerPlus application), you will need to decide how a "seat" should be represented within your application. It can be any 32-bit integer. The seat should

be unique among seats, but the same for each instance of a particular seat. For example, you may want to use the client's IP address. Including the seat in these circumstances will enable you to log out the correct license.

Do not attempt to call the win_lm_login function from the load event of a DLL because a thread cannot be created within a load event.

The win_lm_login function returns a status value. The most common ones are as follows:

0	LM_OK	Success.
1	LM_NOTINS	Product has not been configured.
2	LM_MAXUSR	Concurrent user maximum reached.
7	LM_EXPDEMO	Product demo has expired.

See the table on page 20 for additional codes that could be returned. These values are for checking success or providing application-related license error messages; they have no effect on License Manager. You must write code to handle the returned status. For example, if win_lm_login returns 2 (user maximum reached), you may want to deny the last user access to your application and present an error message stating the problem.

Examples

See **lmdltest.c** in **Licensing_TK_Examples.zip**, available from Synergy CodeExchange in the Resource Center on the Synergex web site.

win_lm_logout – Logout a previously logged in license

```
int WINAPI win_lm_logout(char *producer, char *application, int token,
                         int seat, int *syserr)
```

Arguments

producer

The producer code for your company, provided by Synergex. This is a null-terminated 8-bit ANSI string.

application

The code that you created to refer to the application. This is a null-terminated 8-bit ANSI string and can be up to six characters long.

token

The token obtained from the win_lm_login call (see page 11) and used here to log out.

seat

An int32 set to the value that was passed in the *seat* argument to win_lm_login.

syserr

An int32 pointer returned with a system error code if the call fails.

Discussion

The win_lm_logout function is used only when you have called win_lm_login with *check* set to 0.

The win_lm_logout function doesn't log out a process in the usual sense (that is, it does not prevent the application from continuing). Instead it requests that License Manager release the "slot" for the specified license so that another user can log in and use the available slot.

The token returned by the call to win_lm_login should be saved by your application and then passed to win_lm_logout. The token ensures that the process that is logging out is authorized to do so.

If you passed a non-0 value for *seat* in the call to win_lm_login, pass the same value for *seat* in the call to win_lm_logout. The seat, in combination with the token, ensures that the correct license is logged out in a server environment. If you pass *seat* with win_lm_login and fail to pass the same *seat* value with win_lm_logout, the results could be unpredictable.

You must call win_lm_logout as part of your exit procedure and on an exception, or the license will stay in use until the license server logs it out; this could take as long as 20 minutes.

The win_lm_logout function returns a status value. The most common ones are as follows:

0	LM_OK	Success.
3	LM_BADTOK	Invalid token.

See the table on page 20 for additional codes that could be returned. These status values are for your information only; they have no effect on License Manager. You must write code to handle the returned status.

Examples

See **lmdltest.c** in **Licensing_TK_Examples.zip**, available from Synergy CodeExchange in the Resource Center on the Synergex web site.

Generating Configuration Keys

Once your customers have installed your application, you will need to request that they send you the registration string and licensee name. You must then generate configuration keys, which the customer will use to configure License Manager to run your application.

There are two ways to generate configuration keys:

- Run the License Key Generator utility, **lmk,** from the command line. This method enables you to configure one product at a time. The configuration key is displayed on screen, or you can choose to put it in a Synergy Key file (**.skf**), which can be used by your customer to install the keys. See page 16.
- Use the interactive interface to the License Key Generator, **lmkx**. The **lmkx** program simplifies the collection of information required to generate configuration keys, and enables you to generate keys for several applications at once for each licensee name. It also creates a Synergy Key file. You can download **lmkx** from Synergy CodeExchange in the Resource Center on the Synergex web site. The CodeExchange download (**Licensing_TK_Examples.zip**) includes the source code for **lmkx**, so that you can modify it as necessary to suit your needs.

The **lmkx** program is also useful for creating a demo license that times out, because it enables you to enter either a time-out date or a specific number of days.

License Key Generator Utility

The License Key Generator utility, **lmk**, is used to generate configuration keys. As with other Synergy/DE products, you must configure License Manager to allow **lmk** to run, using the configuration key sent to you by Synergex. Once configured, it cannot be transferred to another machine. In addition, **lmk** is protected from accidentally being copied into a distribution to your customers. The **lmk** utility is available on Windows only.

Syntax

```
lmk [-option] [...]
```

Arguments

option

One or more of the following:

a*app_code*	The application code that you created to represent the application.
c*name*	The licensee name (obtained from the customer site).
d*days*	The number of days since January 1, 1992 that you want the keys to be valid. If you don't want the product to time out, either don't use the **-d** option or set *days* to 0 (zero).
fc*filename*	Create or overwrite a Synergy Key file named *filename* for a specified licensee. The **.skf** extension will be added if not specified.
fa*filename*	Append to an existing Synergy Key file named *filename*. The **.skf** extension will be added if not specified.
h	Display a help message that specifies the **lmk** syntax.
r*string*	The registration string (obtained from the customer site).
u*users*	The maximum number of users for *app_code*.
v	Display the version number of **lmk**.
x*string*	An extended, developer-defined data string. This string can contain up to 100 characters.

Discussion

Arguments that include spaces must be enclosed in quotation marks. For example, if you want to specify the licensee name ABC Consulting Corporation with the **-c** option, you'd enter it as

```
-c"ABC Consulting Corporation"
```

The **-d** option enables you to create a demo license that will expire after a specified number of days. It is easier to create a demo license using the **lmkx** program (included in **Licensing_TK_Examples.zip**, which is available from Synergy CodeExchange in the Resource Center on the Synergex web site) because you can enter either a time-out date or the number of days before time-out, and then **lmkx** calculates this value for you.

If you create a Synergy Key file with the **-fc** option, the file will be created in the current working directory unless you specify a complete path. You can also use a logical to specify the file location. The key file enables users to install keys without having to type the configuration key. If you don't specify the **-fc** option, the keys will display to the screen.

To put the keys in a Synergy Key file, first create the file with the **-c** and **-fc** options. For example,

```
lmk -cMickey -fcKEYS:Mickey
```

creates a file named **Mickey.skf** for licensee Mickey and places it in the directory specified by the KEYS logical. The file header will include the licensee name and the creation date.

Once the file is created, run **lmk** for each application code, and specify the key file with the **-fa** option. This will generate the keys and place them in the file. For example, to generate 10-user licenses for the application codes CON5 and MAR5 and place them in the **Mickey.skf** file, you would use the following:

```
lmk -acon5 -cMickey -r2580RCK7QG88 -u10 -faKEYS:Mickey
lmk -amar5 -cMickey -r2580RCK7QG88 -u10 -faKEYS:Mickey
```

For more information about key files, see the "Configuring License Manager" chapter of the *Installation Configuration Guide*.

The **-x** option enables you to include your own string for anything that you want to check regarding the product that is being licensed. For example, you could use this string to check the version number of your application. (Note that you will have to include code in your application to handle the information included in the string.)

Checking License Usage

The win_lm_stat function (in **syncli_api.dll**) returns the current license usage for an application that has been secured with the Licensing Toolkit API.

If you have an *xf*NetLink .NET or *xf*NetLink COM application, you can use the value returned by win_lm_stat to create a report of license usage, which can then be used to satisfy license usage requirements for *xf*ServerPlus applications, as described in the Synergex License Agreement Terms and Conditions. Contact your Synergy/DE account manager for details on license usage requirements.

You can use the win_lm_stat function to check license usage for any application that has been secured using the routines in the Licensing Toolkit API. When checking license usage in a Synergy application, use the Synergy DLL API to call win_lm_stat; see the "Synergy DLL API" chapter of the *Synergy Language Reference Manual* for information.

1. Secure the application using the routines in the Licensing Toolkit API. See "Securing xfNetLink Applications with License Manager" on page 9.
2. Write a routine to retrieve the license usage value from win_lm_stat. This function returns the current license usage for the specified application. (See page 19.) We recommend that you write a separate utility to do this.
3. Gather the license usage information into a report.

win_lm_stat – Return current license usage

```
int WINAPI win_lm_stat(char *producer, char *application, int *usage,
                       int *syserr)
```

Arguments

producer

The producer code for your company, provided by Synergex. This is a null-terminated 8-bit ANSI string.

application

The code that you created to refer to the application. This is a null-terminated 8-bit ANSI string and can be up to six characters long.

usage

An int32 pointer returned with the current number of users.

syserr

An int32 pointer returned with a system error code if the call fails.

Discussion

The *usage* argument returns the current license usage for the specified application code. This value can be used to prepare a report of license usage, as required for *xf*ServerPlus customers by the Synergy License Agreement Terms and Conditions.

The win_lm_stat function returns a status value. The most common ones are as follows:

0	LM_OK	Success.
1	LM_NOTINS	Product has not been configured.

See the table on page 20 for additional codes that could be returned. If the call fails, *syserr* may be loaded with a system error number.

Examples

See **lmstattest.c** in **Licensing_TK_Examples.zip**, available from Synergy CodeExchange in the Resource Center on the Synergex web site.

Error Messages

The table below shows errors that could occur when using the Licensing Toolkit API. Errors that do not have a Synergy mnemonic occur only when using the C functions.

Number	C mnemonic	Synergy mnemonic	Message	Comments
1	LM_NOTINS	LMSTS_NOCONFIG	License not configured	
2	LM_MAXUSR	LMSTS_MAXUSR	Exceeded concurrent user maximum	
3	LM_BADTOK	LMSTS_BADTOK	Bad logout token	
5	LM_MFAULT		Unexpected failure in UNIX License Server	
6	LM_INTCON		LM internal consistency failure	Run **lmu -k**. See “The lmu utility” in the “Configuring License Manager” chapter of the *Installation Configuration Guide*.
7	LM_EXPDEMO	LMSTS_TIMEOUT	Demo period has expired	
171	INIT_NOFILE		Cannot access license file (or Windows registry)	
172	INIT_BADSND		LM communications error (msgsnd/msgrsv). Likely config problem.	UNIX only
173	INIT_BADRCV		LM communications error (msgsnd/msgrsv). Likely config problem.	UNIX only
174	INIT_NOSYND		LM communications timeout	Windows only
175	INIT_NOLICENSE		Cannot access license file (or Windows registry)	
176	INIT_NOPIPE		Cannot access Network License Server	Windows only

Number	C mnemonic	Synergy mnemonic	Message	Comments
177	INIT_OLDLMDB		Old License Manager version–install latest License Manager	
179	INIT_BUSYPIPE		Network License Server is busy	Windows only
180	OPS_INTCON		LM internal consistency failure	Run **lmu -k**. See "The lmu utility" in the "Configuring License Manager" chapter of the *Installation Configuration Guide*.
181	OPS_BADSND		LM communications error (msgsnd/msgrsv). Likely config problem.	UNIX only
182	OPS_BADRCV		LM communications error (msgsnd/msgrsv). Likely config problem.	UNIX only
183	OPS_NOSYND	LMSTS_NOSYND	Cannot access LM server (not running or MSGWAIT timeout)	UNIX only
184	OPS_MFAULT		LM internal consistency failure	Run **lmu -k**. See "The lmu utility" in the "Configuring License Manager" chapter of the *Installation Configuration Guide*.
185	OPS_NOPIPE	LMSTS_NOACCESS	Cannot access Network License Server	Windows only
186	OPS_EXCLOGINS		Exceeded concurrent user maximum	
188	OPS_BUSYPIPE		Network License Server is busy	Windows only

Glossary

application code — A string of characters that you create and use to identify your application. The application code may have a maximum of six alphanumeric characters. Each application that you want to secure with License Manager must have its own application code.

configuration key — A string of characters generated by the License Key Generator utility based on a registration string provided by your customer. Your customer needs this key to configure License Manager to allow your products to run.

key file — Refers to a Synergy Key file (**.skf**), which is a file that contains configuration keys for one or more licensees. You can use a Synergy key file to automatically configure all products for a workstation at once.

producer code — A number given to you by Synergex that identifies your company.

registration string — A string of characters generated by License Manager. Your customers must send you this string before you can generate configuration keys for them.

skf — Synergy key file. See key file.

slot — A slot is allocated by License Manager each time someone starts the application. When you configure a product for a certain number of users, that number of slots is reserved.

token — A key that is maintained in License Manager to keep track of the licenses that are logged in. The token is obtained from the log-in process and used when logging out to ensure that the process that is logging out is authorized to do so.