Licensing Toolkit User's Guide

Version 10.3.3

Printed: May 2016

DCN LK-01-10.3_04

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 enables you to build Synergy/DE License Manager protection into your own Synergy and xfNetLink applications. The API, distributed with Core Components, includes both Synergy DBL 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, available for Windows only, is used to generate configuration keys for your customers so that they may use your application. This utility is sent to you by Synergex® upon request, along with a configuration key that must be installed before you can use the utility. At the same time, Synergex will assign a producer code to your company, which you will need when using the Licensing Toolkit API.

This document explains how to use Licensing Toolkit to secure your applications with License Manager. When a Synergy 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 Synergy/DE Product License Agreement Terms and Conditions (PLA). See your Synergy/DE account manager for details on the reporting requirements.

Additional resources

- The “Configuring License Manager” chapter of the *Installation Configuration Guide* has 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 website. In addition to **lmkx.exe**, an 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.
- LicensingToolkitExample.zip, available from Synergy CodeExchange in the Resource Center on the Synergex website. This file contains examples of how to create license keys, install license keys onto runtime systems, and implement license checking in your applications. It includes both Workbench and Visual Studio projects; the latter demonstrates using the Licensing Toolkit API from a C# application. See the **readme** file included in the download for details.

If you cannot find the information you need in this manual or in the resources listed above, you can contact Synergy/DE Developer Support department at 800.366.3472 (in the U.S. and Canada), or 916.635.7300 (in all other locations). To learn about your Developer Support options, contact your Synergy/DE account manager at one of the above numbers.

Securing Synergy Applications with License Manager

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 LM_LOGIN on page 5.
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 LM_LOGOUT on page 8.
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 your application. Your security subroutine must include code to handle the returned status.

   TIP 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 DBL Language Reference Manual*.
6. (Recommended) Add an application-level TRY-CATCH-FINALLY statement to your main 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 18.
10. Send the configuration keys to your customers so that they can configure License Manager to run your applications.

LM_INFO – Return license information about the system

WT	WN	U	

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

Arguments

lm_stat

Returned with the License Manager site status. This will be 0 (success) or one of the codes listed in the "Licensing Toolkit Error Codes" table on page 23. (**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 date (YYYYMMDD)
```

Discussion

The LM_INFO subroutine returns License Manager information about the system on which your application is running. The default pre-install time-out date is 14 days after the registration date.

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 your application; 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 website.

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

WT	WN	U	

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

Arguments

app_stat

Returned with the log-in status. The most common ones are listed below; see the "Licensing Toolkit Error Codes" table on page 23 for additional codes that could be returned. (**n**)

0	Product has been configured or a license is available.
1	Product has not been configured.
2	Concurrent user maximum reached.
7	Product demo has expired.

producer

The producer code for your company. (This code was assigned to your company by Synergex when you requested the License Key Generator.) (**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

(optional in traditional Synergy and on UNIX) 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

(optional in traditional Synergy and on UNIX) 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) One of the following options:

0	Log in the license if it is available. (default)
1	Check whether the license is available.
2	Check whether the license is configured.
3	Log in the license if it is configured.

Passing anything other than 0, 1, 2, or 3 is the same as passing 1. An *available* license indicates either that the license is configured or that the 14-day evaluation period is in effect. When *check* is 1 or 2, License Manager only checks on the license; it does not request a log-in slot and so the concurrent number of users is not incremented. See the Discussion for details. (**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, and then returns configuration information about the log-in.

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

When *check* is 0 (or not passed), the call to LM_LOGIN returns a token if the license is configured and a slot is available. If the license is not configured, but the 14-day evaluation period is in effect, the call returns success (0) and a token of 0. In this case, no license is consumed, and so a log-out is not required. If the 14-day evaluation period has expired (and the license has not been configured), the call returns failure (1).

When *check* is 1, if the license is configured or if the 14-day evaluation period is in effect, the call to LM_LOGIN returns success (0) and a token of 0. If the 14-day evaluation period has expired, the call returns failure (1).

When *check* is 2, if the license is configured, the call to LM_LOGIN returns success (0) and a token of 0. Any other condition returns failure (1).

When *check* is 3, the call to LM_LOGIN returns a token if the license is configured and a slot is available, but if the 14-day evaluation period is in effect, the call returns failure (1).

The token returned by LM_LOGIN 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. (All licenses logged in and out are based on the workstation being the seat.)
- In a server environment, where there are multiple processes running on the same machine (such as an xfNetLink–*xf*ServerPlus application), you will need to decide how a "seat" should be represented within your application. *Seat* 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. Each license taken by the same seat for the same product code will be shared. In a server environment, you must include the seat 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 your application; you must write code to handle the returned status. For example, if LM_LOGIN returns 2 (concurrent user maximum reached), you could deny the 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 website.

LM_LOGOUT – Log out a previously logged in license

WT	**WN**	**U**	

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

Arguments

lo_stat

Returned with the log-out status. The most common ones are listed below; see the "Licensing Toolkit Error Codes" table on page 23 for additional codes that could be returned. (**n**)

0	Success.
3	Invalid token.

producer

The producer code for your company. (This code was assigned to your company by Synergex when you requested the License Key Generator.) (**a**)

application

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

token

The token obtained from the LM_LOGIN call 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 3 (or not passed) and the returned token was non-zero.

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.

If the token returned by the call to LM_LOGIN is non-zero, it should be saved by your application and passed to LM_LOGOUT. The token ensures that the process that is logging out is authorized to do so.

If you passed a non-zero 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 an application 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 an application error.

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

Securing xfNetLink Applications with License Manager

Follow these steps to secure an xfNetLink .NET application with License Manager. (**Syncli_api.dll** is not currently supported on xfNetLink Java.) These instructions presume you are using xfNetLink, 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 xfNetLink, 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 win_lm_login on page 12.
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 win_lm_logout on page 15.
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 your application. Your security subroutine must include code to handle the returned status.
7. Provide a way to log out all licenses in case of untrapped errors or when the user exits by clicking the Close button (the X in the upper right corner of an application window). You can add an application-level try-catch-finally statement and use win_lm_logout to ensure that the license is logged out or use the win_lm_cleanup routine to log out all licenses in use by a process.
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 xfNetLink application at a customer site.

1. Set up a Synergy license server on a Windows machine at the customer site.
2. Install xfNetLink 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 xfNetLink machine, run **lmu.exe** (included in the xfNetLink distribution) 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 18.
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

WT	WN		

```
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. This is a null-terminated 8-bit ANSI string. (This code was assigned to your company by Synergex when you requested the License Key Generator.)

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 one of the following values:

0	Log in the license if it is available.
1	Check whether the license is available.
2	Check whether the license is configured.
3	Log in the license if it is configured.

Passing anything other than 0, 1, 2, or 3 is the same as passing 1. An *available* license indicates either that the license is configured or that the 14-day evaluation period is in effect. When *check* is 1 or 2, License Manager only checks on the license; it does not request a log-in slot and so the concurrent number of users is not incremented. See the Discussion for details. (**n**)

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 0, the call to win_lm_login returns a token if the license is configured and a slot is available. If the license is not configured, but the 14-day evaluation period is in effect, the call returns success (0) and a token of 0. In this case, no license is consumed, and so a log-out is not required. If the 14-day evaluation period has expired (and the license has not been configured), the call returns failure (1).

When *check* is 1, if the license is configured or if the 14-day evaluation period is in effect, the call to win_lm_login returns success (0) and a token of 0. If the 14-day evaluation period has expired, the call returns failure (1).

When *check* is 2, if the license is configured, the call to win_lm_login returns success (0) and a token of 0. Any other condition returns failure (1).

When *check* is 3, the call to win_lm_login returns a token if the license is configured and a slot is available, but if the 14-day evaluation period is in effect, the call returns failure (1).

The token returned by win_lm_login 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 xfNetLink–*xf*ServerPlus application), you will need to decide how a “seat” should be represented within your application. *Seat* 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. In a server environment, you must include the seat 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 listed below; see the “Licensing Toolkit Error Codes” table on page 23 for additional codes that could be returned.

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

These values are for checking success or providing application-related license error messages; they have no effect on your application. You must write code to handle the returned status. For example, if win_lm_login returns 2 (concurrent user maximum reached), you may want to deny the 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 website.

win_lm_logout – Log out a previously logged in license

WT	WN		

```
int WINAPI win_lm_logout(char *producer, char *application, 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. (This code was assigned to your company by Synergex when you requested the License Key Generator.)

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.

seat

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

token

The token obtained from the win_lm_login call and used here to log out.

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 or 3 and the returned token was non-zero.

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.

If the token returned by the call to win_lm_login is non-zero, it should be saved by your application and 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-zero 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. If you do not, the license will remain in use until the license server logs it out, which could take as long as 20 minutes. For another way to release licenses see win_lm_cleanup on page 17.

The win_lm_logout function returns a status value. The most common ones are listed below; see the "Licensing Toolkit Error Codes" table on page 23 for additional codes that could be returned.

0	Success.
3	Invalid token.

These status values are for your information only; they have no effect on your application. 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 website.

win_lm_cleanup – Release all licenses for a process

WT	WN		

```
void WINAPI win_lm_cleanup()
```

Discussion

The win_lm_cleanup routine releases (logs out) all licenses that are in use by a process. You do not have to specify a token as you do with win_lm_logout.

We recommend you create a ProcessExit handler for your current AppDomain with a call to this routine to ensure that licenses are released when the process exits regardless of how the user exits the application, as well as when there is an exception.

Examples

The example below shows how to use a ProcessExit handler for the current AppDomain to call the win_lm_cleanup routine.

```
AppDomain.CurrentDomain.ProcessExit += MyProcessExit;

void MyProcessExit(object sender, EventArgs e)
{
   win_lm_cleanup();
}
```

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.exe,** 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 "License Key Generator Utility" on page 19.
- 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 website. 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.

TIP 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 omit this 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. See LmkEvalDate.zip, available from Synergy CodeExchange in the Resource Center on the Synergex website, for a utility that calculates the number of days between a given date and January 1, 1992. It may be easier to create a demo license using the **lmkx** program, because it allows you to enter either a time-out date or the number of days before time-out. The **lmkx** program is included in Licensing_TK_Examples.zip, also available from Synergy CodeExchange.

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.)

Using Synergy key files

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.

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.

To put the keys in a Synergy key file,

1. 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.

2. 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*.

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 xfNetLink .NET 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 PLA. Contact your Synergy/DE account manager for details on license usage requirements.

1. Secure the application using the routines in the Licensing Toolkit API. See "Securing xfNetLink Applications with License Manager" on page 10.
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 win_lm_stat on page 22.) We recommend that you write a separate utility to do this.
3. Gather the license usage information into a report.

TIP 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 with the DLL_TYPE_WINAPI convention to call win_lm_stat. See the "Synergy DLL API" chapter of the *Synergy DBL Language Reference Manual* for information.

win_lm_stat – Return current license usage

WT	WN		

```
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. (This code was assigned to your company by Synergex when you requested the License Key Generator.)

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 PLA.

The win_lm_stat function returns a status value. The most common ones are listed below; see the "Licensing Toolkit Error Codes" table on page 23 for additional codes that could be returned.

0	Success.
1	Product has not been configured.

These status values are for your information only; they have no effect on your application. You must write code to handle the returned status.

If the call fails, *syserr* may be loaded with a system error number.

When calling win_lm_stat from a Synergy application, use the Synergy DLL API with the DLL_TYPE_WINAPI convention; see the "Synergy DLL API" chapter of the *Synergy DBL Language Reference Manual* for information

Examples

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

Error Messages

The table below shows errors that could occur when using the Licensing Toolkit API.

Licensing Toolkit Error Codes

Number	Meaning	Comments
1	License not configured.	
2	Exceeded concurrent user maximum.	
3	Bad logout token.	
5	Unexpected failure in UNIX license server.	
6	License Manager internal consistency failure.	Run **lmu -k**. See "The lmu utility" in the "Configuring License Manager" chapter of the *Installation Configuration Guide*.
7	Demo period has expired.	
171	Cannot access license file (or Windows registry).	
172	License Manager communications error (msgsnd/msgrsv). Likely configuration problem.	UNIX only
173	License Manager communications error (msgsnd/msgrsv). Likely configuration problem.	UNIX only
174	License Manager communications time-out.	Windows only
175	Cannot access license file (or Windows registry).	
176	Cannot access network license server.	Windows only
177	Old License Manager version is installed.	Install the latest License Manager.
179	Network license server is busy.	Windows only
180	License Manager internal consistency failure.	Run **lmu -k**. See "The lmu utility" in the "Configuring License Manager" chapter of the *Installation Configuration Guide*.

Licensing Toolkit Error Codes

Number	Meaning	Comments
181	License Manager communications error (msgsnd/msgrsv). Likely configuration problem.	UNIX only
182	License Manager communications error (msgsnd/msgrsv). Likely configuration problem.	UNIX only
183	Cannot access License Manager server (not running or MSGWAIT time-out).	UNIX only
184	License Manager internal consistency failure.	Run **lmu -k**. See "The lmu utility" in the "Configuring License Manager" chapter of the *Installation Configuration Guide*.
185	Cannot access network license server.	Windows only
186	Exceeded concurrent user maximum.	
188	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.
available license	Refers to a license that is configured or when the 14-day evaluation period is in effect.
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.
PLA	Product License Agreement. Refers to the Synergex Synergy/DE Product License Agreement Terms and Conditions.
producer code	A number given to you by Synergex that identifies your company. This code is used by LM_LOGIN, LM_LOGOUT, win_lm_login, win_lm_logout, and win_lm_stat.
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.