



CRYPKEY SDK 7

Security that Works

SDK
User Manual

SECURITY THAT WORKS

CrypKey SDK 7 User Manual

SOFTWARE VERSION: 7

RELEASE DATE: MARCH 2007

DOCUMENT REVISION NUMBER: 7

© CrypKey (Canada) Inc.
The Devenish Heritage Building
908 - 17th Avenue SW
Suite 200
Calgary, Alberta
T2T 0A3 Canada
Phone 403.258.6274 • *Fax* 403.264.8838
Sales email Sales@CrypKey.com
Support email Support@CrypKey.com
<http://www.CrypKey.com>

CrypKey License Agreement

Grant of Rights. In consideration of payment, CrypKey (Canada) Inc. grants to the purchasing company (the "Customer") the non-exclusive right to possess, use, make, and distribute unlimited copies of the machine-executable code version of the CrypKey Software Licensing System, including all revisions, modifications, and updates thereto furnished to the customer, together with the written User Manuals in relation thereto (collectively the "Software").

Intended Purpose. The Software is intended to be linked with and incorporated into the product, including, but limited to, the foundation module and installation routines forming a part thereof. No right to distribute the Software on a stand-alone basis is intended by this License Agreement. Any other use of the Software requires written consent from CrypKey. The customer may not reverse engineer, modify, nor create derivative works based on the Software without written consent from CrypKey, except as permitted by acceptable law.

Ownership. All patents, copyrights, and other proprietary rights in the Software are, and shall remain, the exclusive property of CrypKey or its suppliers. The Customer may not assign nor transfer the rights granted in the License Agreement to any person, except the Customer's subsidiaries and affiliates, without written consent from CrypKey.

Limited Warranty. CrypKey warrants to the Customer that the Software will operate essentially as described in CrypKey brochures and documentation. CrypKey warrants that the media on which the Software is recorded shall be free from defects in materials and workmanship under normal use and service for 60 days from the date of the Customer's invoice. If failure of the media is a result of accident, abuse, or misapplication of the Software, CrypKey shall not be responsible for its replacement. Applicable law may imply warranties that cannot be excluded or can be excluded only to a limited extent. This Agreement shall be reached and construed subject to such laws.

No Other Warranties. The limited warranty set forth herein is in lieu of, and CrypKey disclaims, any and all other warranties (express or implied) with respect to the software, including any and implied warranty of merchantability or fitness for a particular purpose.

Remedies. Except in connection with a claim for infringement of copyright, patent, or other intellectual property right, the Customer's sole and exclusive remedy for a breach of warranty shall be: (a) the return of the initial fee paid for the rights granted herein or (b) the correction or replacement of defective software or media. Corrected or replaced Software or media will be warranted to the same extent as the original Software or media for the remainder of the original warranty period or 30 days from the date of receipt by the customer, whichever is longer.

Limitation of Liability. In no event shall CrypKey be liable to the customer for indirect, incidental, special, or consequential damage. In no event shall CrypKey Controls Ltd.'s aggregate liability to the customer exceed the amounts paid to CrypKey by the customer for the software.

The software described in this manual is furnished under a License Agreement and may only be used in accordance with the terms of this agreement.

Microsoft, Windows, Visual Basic, Access, and Windows NT are trademarks of Microsoft Corporation. Watcom is a trademark of Watcom International Corporation. Novell is a registered trademark of Novell, Inc.

Copyright ©1992-2007, CrypKey (Canada) Inc. All rights reserved

Table of Contents

Chapter 1: Introduction	1
In this Chapter.....	1
About CrypKey (Canada) Inc.	1
About CrypKey SDK 7	2
CrypKey SDK 7 Standard Features	2
CloneBuster Technology	2
EasyLicense	3
Dynamic Encryption	3
Enhanced Anti-Hacking	3
What's New in CrypKey SDK 7	4
CrypKey SDK 7 -- New Features	4
VistaEasy.....	5
Seamless 64-bit platform support for 32-bit programs.....	5
Seamless 64 bit program support.....	6
Wide Area Network support.....	6
CrypKey and Vista Behavior.....	6
Automatic License Repositioning	7
About Your License.....	8
Testing CrypKey SDK 7	8
Authorized License	9
Register with CrypKey (Canada) Inc. to Get Your Developer Keys.....	10
Using the Temporary License for Development	10
How to Implement Licensing	11
System Requirements.....	12
Technical Support	13
Frequently Asked Questions	13
CrypKey Contact Information	13
Chapter 2: Upgrading	15
In this Chapter.....	15
CrypKey Instant vs. CrypKey SDK.....	15
CrypKey SDK 7 Installation Folder.....	15
Activating and Configuring CrypKey SDK 7	15
Activate a License for CrypKey Stealth.....	15
Activate a License for the CrypKey Site Key Generator	17
Transfer your old configuration file to the new install	18

Re-protect your application	18
Re-Link and Recompile your application.....	18
Stealth your new Executable (optional).....	18
Distribution	18
Upgrade the files.....	18
Installation considerations – Upgrade CrypKey License Server	19
Implementation Considerations – .NET	19
PRE 6504 implementation	19
Post 6504 implementation	20
Using an Existing License	21
Assume a License.....	21
Transfer a License	21
Issue a New License	21
Chapter 3: Installation.....	23
Install CrypKey SDK 7.....	23
Procedure for Installing CrypKey SDK 7	23
Readme Files.....	31
CrypKey SDK 7 Directories	31
CrypKey License Service Installation	35
Libraries and Distribution Files	35
Connectivity to Microsoft Access.....	37
Chapter 4: Getting Started.....	39
New in CrypKey SDK 7	39
CrypKey SDK 7 Utilities	39
Enterprise License Manager.....	40
Network License Manager.....	40
Client License Configuration	40
Installing the Utilities	40
Who Should Use the Utilities.....	40
Step 1. Determine the mode of operation for your application	41
Step 2. Determine Which Library to Link to.....	41
Step 3. Implement CrypKey functions in your application	42
Step 4. Stealth protect your application	43
Step 5. Determine which CrypKey files you need to distribute:.....	44
Step 6. Configure Distribution files.....	45
Step 7. Modify your install to distribute CrypKey files	46
Step 8. Distribute your application	48
Step 9. License the application on your customer's computer	48

Chapter 5: Demo Programs.....	49
Chapter 6: SKG.....	57
Authorizing Site Key Generator.....	57
Features of Site Key Generator	62
Levels and Options	62
Feature Levels.....	63
Feature Options.....	63
Licensing Options	64
License Duration Settings.....	64
Clock Manipulation Checks	64
EasyLicense	64
EasyLicense Backup and Restore.....	65
CloneBuster Technology	65
Transaction Summary	66
Configuring Products with Site Key Generator.....	67
Configuring Site Key Generator for Network Use	77
Editing Products with SKG.....	78
Changing License Usage for your Customers	79
Generating Site Keys	82
Distributor Authorizing License	84
Chapter 7: Programming	85
Basic Programming Steps.....	85
Establishing the License	85
GUI Programming Tasks.....	88
Authorizing the Program	88
Registering the Authorization Transfer on the Target Computer	88
Transferring the Authorization Out of the Source Computer	89
Transferring the Authorization into the Target Computer.....	89
Compiling and Linking	89
CrypKey Header File	89
Linking Your To Application.....	89
COM Object.....	93
COM Object Identifiers	93
Installing a COM Object.....	94
Referencing a COM Object Explicitly	94
Referencing a COM Object by Name	95
Pre-defining DLL functions	96
Using a COM object in C++.....	97
COM Object Constants.....	99

Internal COM Status Messages	100
Referencing the Floating License Snapshot Record	101
Installing the CrypKey License Service	103
Error Codes	103
Run Mode	105
Installation Strategies	105
Uninstalling the CrypKey license service	105
Testing the Installer	107
Supported Versions of Windows	107
Chapter 8: Protection	109
Why Stealth	109
Visual C++ Custom Stealth Example	110
Stealth File Auto-Distribution	111
Using the Interface (STEALTHUI)	111
Counter-Hacker Strategies	117
Anti-tampering Measures	117
Error Codes	119
Chapter 9: Network Licenses	121
Installing the Utilities	122
Who should use the Utilities	122
Chapter 10: Moving Protected Programs	125
Direct License Transfers	126
Media Transfer	126
Chapter 11: ELM	129
Overview	129
CrypKey SDK 7 Network Licensing Mode	129
Server Mode	129
New to CrypKey 7	129
Implementation Notes	131
ELM User	131
Sample Configurations	131
Using the Enterprise License Manager	137
Purpose of the ELM	137
Running in Server Mode	137
Edit Menu	137
View Menu	137
Actions Menu	138

Transferring Licenses	141
License Transfer Management Screen.....	141
Some notes about license transfers	142
Troubleshooting.....	143
Chapter 12: NLM.....	145
Network License Manager (NLM)	145
New Server Wizard.....	146
Open Manager.....	153
Open Folder.....	154
Start License Server	154
Stop License Server	154
Delete License Server	154
Find Inactive Server.....	154
Example configuration file.....	155
Chapter 13: CLC	157
Client License Configuration	157
Using the Client License Configuration Utility	157
Chapter 14: Function Reference	163
Function Summary.....	163
General Error Values	167
Function Descriptions	171
AcquireLicense().....	171
Prototypes	172
Parameter.....	172
Return Value	173
CKChallenge32()	174
Function Usage	174
Prototypes	174
Parameters.....	175
CrypKeyBuildNum()	176
Description	176
Function Usage	176
Declaration (C/C++)	176
Declaration (Visual Basic)	176
Parameters.....	176
Return value.....	177
CKTimeString().....	177
Prototype.....	177

Parameters	177
Return Value	177
CrypkeyVersion()	177
Description	177
Function Usage	178
parameters	178
Prototypes	178
Return Value	178
DirectTransfer()	178
Function Usage	179
Prototypes	179
Parameter	179
Return Value	179
EndCrypkey()	181
Function Usage	182
Prototypes	182
ExplainErr()	182
Function Usage	182
Prototypes	182
Parameters	183
Return Value	184
ExplainErr2()	184
Function Usage	184
Prototypes	184
Parameters	185
FloatingLicenseGetNumEntries()	186
Prototypes	186
FloatingLicenseGetRecord()	186
Prototypes	187
FloatingLicenseSnapshot()	187
Prototypes	187
Parameters	188
FloatingLicenseTakeSnapshot2()	190
Prototypes	190
Get1RestInfo()	190
Function Usage	190
Prototypes	190
Parameter	191
GetAuthorization()	191
Function Usage	191

Prototypes	192
Parameters	192
GetAuthorization2()	200
Function Usage	200
Prototypes	200
Parameter	200
GetCustomInfoBytes	201
Function Usage	201
Prototypes	201
Parameters	202
GetDrivePermanentSerialData()	202
Function Usage	202
GetLastError()	202
GetLastErrorCode()	203
Prototypes	203
GetLevel()	203
Function Usage	203
Prototypes	203
Parameter	204
GetNetHandle()	204
Function Usage	204
Prototypes	204
GetNumCopies()	205
Function Usage	205
Prototypes	205
GetNumMultiUsers()	206
Function Usage	206
Prototypes	206
GetOption()	206
Function Usage	206
Prototypes	206
Parameters	207
GetRestrictionInfo()	207
Function Usage	208
Prototypes	208
Parameters	208
GetSiteCode()	211
Function Usage	211
Prototypes	211
Parameter	211

GetSiteCode2()	212
Function Usage	212
Prototypes	212
InitCrypkey()	213
Function Usage	213
Prototypes	213
Parameters	214
IsEasyLicense()	218
Description	218
Function Usage	218
Declaration (C/C++)	218
Declaration (Visual Basic)	218
Parameters	218
Return value	218
KillLicense()	219
Description	219
Function Usage	219
Declaration (C/C++)	219
Declaration (Visual Basic)	219
Parameters	219
Return Values	219
ReadyToTry()	220
Description	220
Usage	220
Declaration (C/C++)	221
Declaration (Visual Basic)	221
Parameters	221
Return values	221
ReadyToTryDays()	224
Prototypes	225
Parameters	225
ReadyToTryRuns()	226
Function Usage	226
Prototypes	226
Parameters	227
RegisterTransfer()	227
Function Usage	228
Prototypes	228
Parameter	228
SaveSiteKey()	230

Function Usage	230
Prototypes	230
Parameters	231
SetNetHandle()	232
Function Usage	232
Prototypes	232
Parameter	233
SetCustomInfoBytes()	233
Description	233
Function Usage	233
Declaration (C/C++)	233
Declaration (Visual Basic)	233
Parameters	234
Return values	234
TransferIn()	234
Function Usage	234
Prototypes	235
Parameter	235
TransferOut()	239
Function Usage	240
Prototypes	240
Parameter	240
Appendix A: Quick Reference	244
Glossary	248
Index	251

List of Tables

Table 1: Files required for Distribution with 32-bit Applications	36
Table 2: Link to Library	41
Table 3: OS Specific Files	44
Table 4: CrypKey Libraries	44
Table 5: Mode Specific Files	45
Table 6: Mode Specific Tasks	47
Table 7: Libraries	89
Table 8: Library Linkages	92

Table 9: COM Object Identifiers	93
Table 10: Data type: <i>CKExplainEnum</i>	99
Table 11: Data type: <i>CKRestrictionTypeEnum</i>	99
Table 12: Data type: <i>CKBooleanValueEnum</i>	100
Table 13: Data type: <i>CKMaxLengthEnum</i>	100
Table 14: Internal COM Status Messages	100
Table 15: COM C++ and C#.NET Prototypes for FLS Record Data Items	101
Table 16: Operation Mode and Installation Configuration	122
Table 17: Function Summary	163
Table 18: General Errors and Solutions	168
Table 19: DirectTransfer() Values	179
Table 20: ExplainErr() Functioncode Values	183
Table 21: ExplainErr(2) Functioncode Values	185
Table 22: Floating License Snapshot – Return Values	188
Table 23: Floating License Snapshot – Record Structure	189
Table 24: Get1RestInfo() Values	191
Table 25: GetAuthorization() Values	192
Table 26: GetNumCopies() Values	205
Table 27: GetOption() Values	207
Table 28: Authopt Values	208
Table 29: Num_used Values	209
Table 30: GetSiteCode() Values	212
Table 31: InitCrypkey() Values	215
Table 32: RegisterTransfer() Values	229
Table 33: SaveSiteKey() Values	231
Table 34: TransferIn() Values	235
Table 35: TransferOut() Values	240

List of Figures

Figure 1 Select Stealth Utility	16
Figure 2 Stealth – Unauthorized License	16
Figure 3 Stealth – License Agreement	17

Figure 4 Stealth License Authorization email	17
Figure 5 Replace files in your Installation folder.....	19
Figure 6 Install screen 1 – Enter Password.....	24
Figure 7 Install screen 2 – Welcome to Wizard	25
Figure 8 Install screen 3 – License Agreement	26
Figure 9 Install screen 4 – User/Company Information	27
Figure 10 Install screen 5 – Directory location	28
Figure 11 Install screen 6 – Start Install.....	29
Figure 12 Install screen 7 – Installation in progress	30
Figure 13 Install screen 8 – Installation complete	31
Figure 14 Documentation directory	33
Figure 15 Supporting Documentation directory	33
Figure 16 CrypKey SDK 7 Program Group	49
Figure 17 Sample Compiled Code	50
Figure 18 Test Application window.....	51
Figure 19 Test Application window with Site Code.....	52
Figure 20 Validated Site Code	53
Figure 21 Generated Site Key.....	54
Figure 22 TESTAPP Window with Authorized Site Key	55
Figure 23 Select Site Key Generator.....	58
Figure 24 Site Key Generator splash screen	58
Figure 25 Site Key Generator main menu	59
Figure 26 SKG License – Not Authorized	60
Figure 27 SKG License – Authorized	61
Figure 28 SKG – Authorized	62
Figure 29 SKG – License Summary window.....	66
Figure 30 Configure a New Product.....	67
Figure 31 Site Key Generator with New Product Pop-up Window.....	68
Figure 32 New Product listed in Configure window	69
Figure 33 Configure New Product – Enter Password.....	70
Figure 34 Site Key Generator – Options tab	71
Figure 35 Site Key Generator – Add Options	72
Figure 36 Site Key Generator – Rename Option.....	73

Figure 37 Site Key Generator – Level tab	74
Figure 38 Site Key Generator – Assign Levels to a Product.....	75
Figure 39 Site Key Generator – License Summary window	76
Figure 40 Product License Details.....	77
Figure 41 Configure Existing Product.....	78
Figure 42 Rename Product	79
Figure 43 Add to Existing License	80
Figure 44 Generate New Site Key for Existing License	81
Figure 45 CrypKey Site Key Generator window	82
Figure 46 Site Key Generator – Check Customer Site Code.....	83
Figure 47 License Agreement pop-up	112
Figure 48 Enter Site Key for Stealth	113
Figure 49 Stealth Configuration window.....	114
Figure 50 Stealth Configuration window.....	115
Figure 51 Stealth pick file names message	115
Figure 52 Activate Pick folder option	116
Figure 53 Stealth Compression begun	117
Figure 54 Configuration 1: Stand-alone installation – Normal mode	132
Figure 55 Configuration 2: Client/Server Configuration	134
Figure 56 Configuration 3: Drive Share Installation	135
Figure 57 Configuration 4: License Transfer Configuration	136
Figure 58 Edit Menu: Edit Configuration option	137
Figure 59 View Menu: Network Clients option.....	138
Figure 60 View Menu: All CrypKey Servers option.....	138
Figure 61 View Menu: All CrypKey Servers with Available Licenses option	138
Figure 62 Actions Menu: Start Server	139
Figure 63 Actions Menu: Stop Server.....	139
Figure 64 Actions Menu: Transfer License	140
Figure 65 Actions Menu: Select a License	140
Figure 66 Actions Menu: Manage Server Licenses	140
Figure 67 Enterprise License Manager screen.....	141
Figure 68 Launch NLM from the Control Panel	145
Figure 69 Network License Manager Main screen	146

Figure 70 Server Configuration Wizard	147
Figure 71 Select License.....	148
Figure 72 License Selection processing.....	149
Figure 73 License Domains screen.....	150
Figure 74 Initializing Data screen	151
Figure 75 License Machines screen.....	151
Figure 76 License Users screen.....	152
Figure 77 License Status Selection screen	153
Figure 78 Network License Manager screen with enabled license.....	153
Figure 79 Browse for Inactive Servers	154
Figure 80 Find Inactive Server	155
Figure 81 Select License Configuration file.....	158
Figure 82 Client Properties screen.....	159
Figure 83 License Domains page.....	160
Figure 84 License Machines page	161
Figure 85 License Status page	162
Figure 86 Options and Levels	199

Chapter 1: Introduction

CrypKey SDK 7 offers true code encryption for protecting your software.

CrypKey SDK 7 (Software Developer's Kit) is a customizable software licensing and copy protection system. Protect your software against unauthorized use and duplication using true code encryption.

For an explanation of the difference between true encryption as offered by CrypKey SDK 7, and simple obfuscation, see our white paper online at <http://www.crypkey.com/whitepapers.asp>.

In this Chapter

In this chapter you will find information on:

- CrypKey Canada Inc.
- what CrypKey SDK 7 does
- what's new in CrypKey SDK 7
- CrypKey SDK 7 trial licenses
- system requirements
- technical support

About CrypKey (Canada) Inc.

To find out more about CrypKey and our approach to preventing software piracy – one of the most costly and intimidating problems for software developers – visit our web site at www.CrypKey.com.

About CrypKey SDK 7

CrypKey SDK 7 is a Software Licensing Application Program Interface (SLAPI) that provides invisible security and complete flexibility over the licensing of your products. Its capabilities include:

- protect a software product from unauthorized use on a turnkey basis
- grant customers different levels of privileges
- grant customers usage privileges limited by time or number of uses
- uses no hardware key or disk key
- includes a software library that easily integrates with the developer's existing programs
- can be used to track and control protected software sold by third party distributors
- defeats hard disk drive (HD) cloning of your software.

CrypKey SDK 7 Standard Features

CrypKey SDK 7 carries forward proven, robust technology from previous versions:

CLONEBUSTER TECHNOLOGY

Software developers are discovering they are losing billions of dollars in a worldwide license theft on a large scale. Fueled by the recent surge in use of legal, low-cost hard drive (HD) cloning, or “ghosting” software, easily-obtained \$49 utilities with names like Ghost and XXCopy are rendering once-popular copy protection techniques useless, generating fast lucrative returns for software pirates.

Many popular copy protection technologies are no longer an effective option. They depend on placing some type of simple software footprint and are now easily defeated by the new HD cloning techniques. Many unproven, start-up, venture-capital backed, rights management platform providers are not practically viable. They offer grand, and often expensive, protection schemes that fail to recognize all the numerous firewall, integration, distributed deployment, and support issues generated by their self-described “connected product vision”. Hardware dongles provide limited protection against HD cloning, but are expensive and disliked by end users. CrypKey SDK 7, which includes CloneBuster technology, effectively solves this problem.

Most modern IDE hard drives have burned-in Hard Drive Serial Numbers (HDSNs). Not to be confused with the easily changed volume serial number, the

HDSN is a permanent, read-only attribute of a personal computer hard drive. The HDSN is very difficult to access, but CrypKey SDK 7 with CloneBuster technology has this ability. Leveraging CrypKey's 15-year, proven protection architecture, CloneBuster detects the HDSN, model number, and firmware identifier and uses this information in its proprietary licensing algorithms. Although HD cloning programs succeed in breaking other simple copy protection schemes in order to copy an entire hard drive, they cannot achieve this when you are armed with CrypKey SDK 7 with CloneBuster.

EASYLICENSE

Some customers need the simplest possible form of copy protection when an application runs only on a particular computer. These customers do not want to deal with any associated issues of transfer, network license, trial period, or number of copies. Whether the software is offloaded and put back, deleted and restored later, or the entire hard drive is erased and restored, the application still runs on the computer. The EasyLicense technology can only be used if a HDSN is read off of the user's machine as CrypKey ties the license to the HDSN. **EasyLicense** offers this type of protection coupled with simplicity.

EasyLicense now supports days limited licenses.

DYNAMIC ENCRYPTION

At CrypKey we are never satisfied with just good enough. We continue to have a dynamic multiple encryption key scheme, so that encryption keys are constantly changing on the fly. This is truly a code breaker's nightmare. This also halts any attempts to create a rogue key generator.

ENHANCED ANTI-HACKING

Stealth™, our anti-hacking layer included with CrypKey, is a critical component of CrypKey SDK 7. It stops the hacker from reverse engineering and circumventing copy protection. Stealth™ already encrypts your binary file and guards against program patches while it is in memory. We have enhanced Stealth so it partially un-encrypts programs while it runs in memory and re-encrypts the parts that are not running. The program teams up in memory with a special symbiotic process that un-encrypts parts as needed, dramatically decreasing vulnerability.

What's New in CrypKey SDK 7

CrypKey SDK 7 offers advanced enhancements:

- True Network Licensing – CrypKey SDK 7 allows your software to find its license anywhere on the Enterprise network automatically – no need to share the drive. This is faster, more secure, and opens up a host of opportunities for flexibility and license control.
- Redundant License Servers – If one License Server crashes, CrypKey SDK 7 can automatically find another, on the fly.
- Network License Control – the server can specify which domains, users or machines can be included or excluded from a license. The client can also specify which license servers it can obtain a license from.
- Smart License Server Balancing – if more than one qualified server exists, CrypKey SDK 7 will pick the server with the most licenses available. If all are busy, it will wait on the server with the least clients ahead in the queue.
- 4 Billion Network Seats – the limit of 127 seats is removed*. Just let CrypKey know if you need to allow more than 10 seats.

* Note: You can now have more than 127 network license seats if needed. Talk to us at CrypKey regarding how to achieve this.

- Enterprise License Transfer – individual Licenses can be transferred out from a pool of licenses, and transferred back in again (with no drive share required). This allows a “checkout” model of license sharing.
- Separate NGN process - permits users to debug a program while using CrypKey.
- CloneBuster for SCSI – CrypKey SDK 7 will now get a serial number from the PC if it can't get one from the hard drive (as is usually the case with SCSI drives). This will prevent illegal license duplication by cloning SCSI drives.
- eTransfer – allows licenses to be quickly and easily transferred from one computer to another via the Internet.
- CrypKey USBKey support – allows you to give your customers a hardware key without code changes vs. an electronic one.

CrypKey SDK 7 – New Features

CrypKey SDK 7 features a new suite of utilities to make license configuration simpler and extremely versatile:

- **Enterprise License Manager (ELM):** With CrypKey SDK 7, the sharing of licenses is communicated invisibly and automatically through the network, and the ability to do this is built in to every C7PP (CrypKey 7 Protected Program).
- **Network License Manager (NLM):** The Network License Manager is used to control server licenses on the computer where it is installed.
- **Client License Configuration (CLC):** The end user can use the Client License Configuration (CLC) wizard to manage license configuration of a client on a network. CLC enables the user to define where on the network the client application should look for a license.
- **VistaEasy:** Normally, applications are installed in the “Program Files” directory, and more often than not, the application will put the license files in the same directory as the application. It is possible to change where the license is stored, in both CrypKey SDK and CrypKey Instant, but with the VistaEasy feature, you don’t have to. VistaEasy takes care of the problems created by the new Vista behavior, so you don’t have to do anything to make CrypKey run on Vista. See the following section on VistaEasy for details about CrypKey behavior on Vista.

VistaEasy

CrypKey has taken Vista compatibility a giant step forward by making CrypKey Instant and CrypKey SDK “Vista Easy”.

We have noticed that many people are struggling with the way Vista handles file writes in the Program Files directory. Some of the time Vista will redirect these files to a user specific “Virtual Store” directory that is not available to other users of the computer. CrypKey has found a way around this problem by automatically detecting your program’s attempt to put a license in a “Program Files” (or other Vista-illegal directory), and before Vista redirects it to a directory you don’t want, we redirect it to directory that you do want – one that works for all users on the computer.

With CrypKey SDK 7, you don’t have to change anything to make CrypKey work seamlessly with Vista.

SEAMLESS 64-BIT PLATFORM SUPPORT FOR 32-BIT PROGRAMS

Your 32-bit program will run on 64-bit computers – but will the licensing work on 64-bit machines? CrypKey’s licensing and copy protection “magic” is contained in the CrypKey driver, and we have developed a 64-bit version of this driver. The driver install that comes with CrypKey *automatically* detects 32/64-bit machines and installs the correct version of the driver. The driver handles the differences

between 32-bit and 64-bit platforms, and your program will continue to operate normally on either platform.

With CrypKey SDK 7, you don't have to change anything to make CrypKey work seamlessly on 64-bit Windows computers.

SEAMLESS 64 BIT PROGRAM SUPPORT

Are you contemplating making your application 64-bit now or in the future? CrypKey has already got you covered, with the addition of 64-bit support in the new version of CrypKey SDK. Using the new 64-bit library in the SDK, you can implement CrypKey in your 64-bit application just as easily as in a 32-bit application – while maintaining the same functionality.

If you are migrating your 32 bit app, you don't have to change anything to make CrypKey SDK 7 work seamlessly in the 64-bit application.

WIDE AREA NETWORK SUPPORT

CrypKey SDK 7 brings additional network functionality to your application. CrypKey already supports the placement of licenses anywhere on a Local Area Network; in addition, CrypKey SDK 7 allows licenses to be accessed over a Wide Area Network.

This means that if your customers have computers that access a network from outside of the building, as is the case when they use VPN, they can still access the CrypKey license on a computer inside the building.

With CrypKey SDK 7, you don't have to change anything to support Wide Area Network licensing.

CRYPKEY AND VISTA BEHAVIOR

CrypKey behavior on Vista is summarized here. For more information, see the documentation on the Microsoft web site.

If a program tries to write files to the directory or a sub directory of the system directory or "Program Files", Vista behavior is very different from the preceding operating system. Vista does not allow programs to be written to these directories. There is much more detail as to the how, when and why than included here, but in summary, depending on circumstances, Vista will do one of three things:

1. Vista redirects the file write (and reads) to a Virtual Store directory in the users profile.

Example: A write to

C:\Program Files\License\license.lic

will be redirected to

C:\Users\Jim\AppData\Local\VirtualStore\Program Files\License\license.lic

2. Vista fails the write.
3. Vista allows the write.

This Vista behavior will affect CrypKey pre-Version 7 licensing if your application tries to put the license in these directories – it will make the license available only to the user who created it – or it may fail.

If your application is writing its license in a Vista-protected directory (most likely under “Program Files”) CrypKey will automatically detect this, and will always redirect the license files to a directory under the “All Users” directory.

For example, CrypKey will intercept the call to put a license in

C:\Program Files\License\license.lic

and redirect it to

C:\ProgramData\Program Files\ License\license.lic

This has two very important consequences:

1. The license will be available to all users of the computer.
2. This behavior will be consistent under all circumstances

Automatic License Repositioning

If you have used a previous version of CrypKey, and your customers have licenses that have been redirected to a User profile directory by Vista, these licenses will only be available to the user who created them.

The first time an application with CrypKey 7.1 is run, CrypKey will detect this situation, and automatically move the license to a directory under “All users”, so that the license will be available to any user on the computer.

About Your License

Important

If you are currently using an older version of CrypKey, do not transfer your license to this new version. Please send the Site Codes from the Site Key Generator (SKW.EXE) and StealthUI application to: Authorize@CrypKey.com and we will issue you new licenses.

CrypKey V6 Users will also require updated developer keys, issued from CrypKey (Canada) Inc., to use CrypKey SDK 7.

Testing CrypKey SDK 7

Once you have purchased CrypKey SDK 7, you have the ability to evaluate the full functionality of the system. Test the CrypKey SDK 7 system with the example keys provided to you, to ensure that CrypKey will meet your needs.

This user manual contains example developer keys (see below) for use during your testing phase. These temporary keys are not intended for distribution with your product. Since all Site Key Generators can create keys for any program that uses the temporary keys, the only keys you should distribute with your product are those specifically created for the product.

Warning

If you distribute your product with these temporary User and Master Keys, recipients who use CrypKey can unlock your product.

Example Keys

- User Key: D050 815C D1A2 A79D B1
- Master Key (32-bit):

```
6a6167abf35cb0253b91761971cfcecf79b6935d6e77905f5b29a45badd3a4f213
15cee18798f96ef1861455c0f18643746f1183580caa6a3f5ef6d8e3dd7e15da0f3a
b21414b8fbe549d385552f816406b15040070d636ab2153d715e71c6c2eeda08f5
8503f64ce9183759075949ff249013d045b55907aa6924c5d707546
```

CrypKey SDK 7 offers two types of developer keys — those related to the Example file and those locking CrypKey to your product. Think of developer keys as alphanumeric strings of text. You require the following four keys to run CrypKey SDK 7:

- **User Key:** created from the CrypKey password you specify.
- **Master Key:** created from the file name you give to CrypKey and other pertinent information. We recommend that the filename that you provide to CrypKey to license your software has the extension *.lic*
- **Site Code** (for the Site Key Generator, generated from the computer you have installed CrypKey SDK 7 on): required in order to obtain your Site Key.
- **Site Key** (for the Site Key Generator): enables you to authorize your product using your Site Key Generator.

Authorized License

Once you have fully tested your product with the CrypKey example developer keys, you will need to obtain your company's developer keys. Please note that if you are ready to ship CrypKey with your product, you will need the following developer keys and numbers:

User Key: This is an encrypted form of the developer/product-specific password you supply to CrypKey. Use this key for the `InitCrypkey()` function.

Master Key: This is an encrypted form of information specific to the product being protected and is provided to you by CrypKey. We recommend you choose a license filename with the *extension .lic*. Use this key for the `InitCrypkey()` function.

Site Key: This key is required to unlock any CrypKey-protected software. Since the Site Key Generator is protected, you must send us the Site Code so that we can authorize your Site Key Generator. You can find the Site Code to the Site Key Generator in the license button of the Site Key Generator. The Stealth application also needs to be unlocked if you would like to use this feature.

Company Number: This value is specific to your company and is used for the `CKChallenge32()` function.

Password Number: This value is specific to your password and is used for the `CKChallenge32()` function.

Although you can develop your application using the temporary license provided, you eventually need the developer keys for final testing and shipping of your product. These keys are issued only after we receive your payment. If you are working under an aggressive development schedule (who isn't these days?), we ask that you time your payment accordingly.

REGISTER WITH CRYPTKEY (CANADA) INC. TO GET YOUR DEVELOPER KEYS

Procedure

1. Finish testing your product with CrypKey SDK 7, using the example developer keys.
2. Confirm you have paid for CrypKey SDK 7. If you selected BillNet30 or Telegraphic Transfer, please ensure your payment has been sent. We can authorize you to use CrypKey SDK 7 on your product only after the payment has been received.
3. Send us the following information via email, fax, or telephone:
 - a. Contant Name – person at your company CrypKey can contact if we have any questions or concerns;
 - b. Company name your product is registered under;
 - c. your Customer Service Number (CSN), assigned to you by CrypKey, found on your sales receipt;
 - d. Contact phone number;
 - e. Name/Title of your product;
 - f. the file name of your product (maximum 8.3 characters; we recommend only registering filenames that are alpha/numeric with .lic extensions);
 - g. the password for your product (maximum 12 alphanumeric characters);
 - h. your Site Code from your Site Key Generator (see *Chapter 6: SKG, Authorizing Site Key Generator*); and
 - i. Your Site Code from Stealth if you would like to use this feature.

USING THE TEMPORARY LICENSE FOR DEVELOPMENT

For the purposes of development and testing, you can temporarily use the following example keys in your source code:

User Key: D050 815C D1A2 A79D B1

Master Key:

```
6a6167abf35cb0253b91761971cfcecf79b6935d6e77905f5b29a45badd3a4f213
15cee18798f96ef1861455c0f18643746f1183580caa6a3f5ef6d8e3dd7e15da0f3a
b21414b8f549d385552f816406b15040070d636ab2153d715e71c6c2eeda08f5
8503f64ce9183759075949ff249013d045b55907aa6924c5d707546
```

To use these keys, you must (temporarily) either place a file called *example.exe* in the same directory as your application or rename your executable to *example.exe*.

Under this temporary arrangement, you are checking and modifying the license for *example.exe*. This means you can use *example.exe* to test your program's license operations (see *Chapter 3: Install* for more details).

How to Implement Licensing

CrypKey SDK 7 can best be thought of as a Software Licensing API (SLAPI). The following steps describe how CrypKey is used in the development process:

1. You (the developer) purchase the CrypKey SDK 7 software licensing system and program your application with the library security functions. You must embed the primary CrypKey functions in your source code using CrypKey SDK 7, which handles the initialization, license verification, Site Code generation, and termination features.

You may also choose to organize the features of your application in terms of specific "levels" and "options", which are automatically facilitated within the CrypKey system. In fact, we recommend you do this if it is appropriate for the application, since this allows you to use the full feature set that CrypKey provides for licensing.

CrypKey also has a product available called CrypKey Instant, which embeds the CrypKey licensing protection system within your application executable without requiring any changes to the source code. This product offers an alternative method of copyright protection to developers and distributors. Contact Sales@CrypKey.com for more information.

2. **You experiment with CrypKey SDK 7 and test the application using the temporary license provided (see *Testing CrypKey SDK 7 on page 8*). Once you have finished testing your product using the example developer keys, and are completely satisfied with the performance of CrypKey SDK 7, email us the required information (see *Authorized License on page 9*). All CrypKey SDK 7 features are available to you during the testing phase.**

CrypKey will email you developer keys to activate your Site Key Generator license. CrypKey requires the product executable name, password, Site Key Generator Site Code, and customer service identification number in order to create the developer keys that enable you to license your product(s) to customers (using the Site Key Generator).

Note: Please note that you must use the example developer keys until you are finished testing. Once the developer keys for your product(s) have been issued, no refunds are issued.

3. **You distribute the application that you have developed to potential customers.** You can (optionally) give your customer a trial period based on number of days or runs. The customer can then experiment with the product to determine which functions and options are desired.

Once the trial period expires, the customer needs a Site Key to unlock the product.

4. **When your customer is satisfied with the application, he or she obtains a Site Code from the application and submits the code to you (typically by email), along with the license permissions the customer wants to purchase.** The customer provides payment and the Site Code to you. The Site Code is based on a special internal hardware signature that is unique to the personal computer on which the application is installed. The Site Code appears as an encrypted alphanumeric number. CrypKey SDK 7 allows license restrictions to be specified in terms of an unlimited license, or a limited number of days or runs. “Runs” are implemented in the application in whatever manner you designate. The number of program copies or concurrent network users is also specified within the license.
5. **You enter the customer’s Site Code into the Site Key Generator and set the license configuration.** The Site Key Generator automatically verifies that the Site Code given is authentic. A Site Key, which provides specified access to your application, is then generated and given to the customer when you receive payment. The specific levels, options, and permissions are generally referred to as the “license” in the CrypKey system. This information is embedded and encrypted within the Site Key. When the customer enters the Site Key, it is decrypted and activates and configures the customer’s license. The Site Key you generate works for your CrypKey-protected application.
6. **Your customer uses the Site Key provided by you to unlock the application under the terms specified.** Site Codes and Site Keys can be transferred by email, telephone, or other means because they are alphanumeric strings of manageable length. The process of issuing a Site Key for a customer’s specific Site Code is referred to as “authorization” in the CrypKey system. CrypKey SDK 7 automatically tracks restrictions in terms of days or runs and invalidates the license when the time period expires or the number of runs is used. The CrypKey SDK 7 functions inside the application verify that the Site Code/Site Key combination is valid before the customer is granted access. Also, the license is unique to the personal computer on which the application runs. If the customer copies the program and its license files to another computer, the license becomes invalid on that machine and does not allow access. This is possible because CrypKey is capable of identifying the exact computer the license was issued for without the use of an external hardware (dongle) or floppy disk key.
7. **You achieve fame and fortune as customers everywhere are pleased with the product and the clean, non-intrusive licensing system you have provided.** Steps 4 through 6 can be repeated if the customer’s needs change and the customer wants to buy licenses for different levels or options for the application.

System Requirements

The system requirements of CrypKey SDK 7 are:

- 133 MHz 32 MB 650 MB
- 1 MB free hard drive space

- Windows 2000, 2003, NT, XP, or Vista.

CrypKey SDK 7 supports the following development platforms:

- Microsoft C++ 6.0, 7.0, and 8.0 (directly, using static libraries)
- Microsoft Visual C++ 5.x and 6.x, and C++.Net (directly, using static libraries)
- Borland C++ 5.x or more recent (directly, using a static library)
- any development system capable of reading and writing a 32-bit file or calling a DLL. This includes, but is not limited to, Microsoft Visual Basic, Microsoft FoxPro, Microsoft Access, CA Clipper, Borland Delphi, Borland Visual dBASE, Borland Paradox, and Powersoft PowerBuilder.
- .Net — VB.Net, C#.Net.

Technical Support

CrypKey SDK 7 includes 12 months of technical support, at the level of support that you purchased for the software, starting from your date of purchase. When requesting assistance from CrypKey Technical support, please ensure you have first done the following:

- read the manual (most problems can be solved using the information contained in this document)
- work through the demonstrations in Chapter 5, which provide a basic understanding of CrypKey SDK 7
- test the sample program within the scenarios you are investigating (again, most questions can be answered using this method; if problems persist, try the same situation with *example.exe* which is included in the install directory/demos).

Frequently Asked Questions

For a list of up-to-date Frequently Asked Questions (FAQs), please visit the CrypKey support website at <http://www.crypkey.com/faq.asp>.

CrypKey Contact Information

CrypKey (Canada) Inc.

Mailing Address: CrypKey (Canada) Inc.
The Devenish Heritage Building
908 - 17th Avenue SW
Suite 200
Calgary, Alberta
T2T 0A3 Canada

Phone: 1-403-258-6274
Fax Line: 1-403-264-8838
Support email Support@CrypKey.com
Sales email Sales@CrypKey.com
Website <http://www.CrypKey.com>

Chapter 2: Upgrading

*This chapter provides details on how to upgrade
CrypKey SDK on your system.*

If you are already using CrypKey SDK, this chapter contains important information you need to know in order to upgrade your product.

In this Chapter

In this chapter you will find information on:

- activating and configuring CrypKey SDK 7
- re-protecting your application
- upgrading files
- upgrading the CrypKey License service.

CrypKey Instant vs. CrypKey SDK

The upgrade process differs depending on which CrypKey product you use to protect your application. This document concerns the upgrade process for applications protected with CrypKey SDK. For information on upgrading to CrypKey Instant, please see the CrypKey Instant manual.

CrypKey SDK 7 Installation Folder

The default installation directory will be similar to: C:\Program Files\CrypKey SDK\Build 7xxx. You can select a different directory during install if you choose. The last four digits '7xxx' may be different as they represent the build number of CrypKey SDK 7.

Activating and Configuring CrypKey SDK 7

Before you can use the Stealth utilities or the Site Key Generator, you must:

1. Activate a license for Stealth;
2. Activate a license for the Site Key Generator; and
3. Transfer your old configuration file (skw.ini) to the new install.

Activate a License for CrypKey Stealth

CrypKey will send you a new license for this version of CrypKey SDK 7.

1. From the Start menu, select **CrypKey StealthUi Utility**:

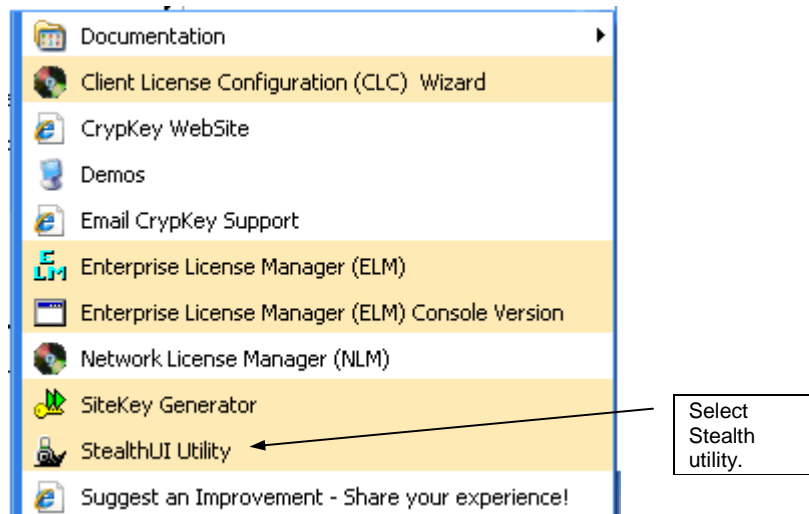


Figure 1 Select Stealth Utility

2. The Stealth license screen opens:

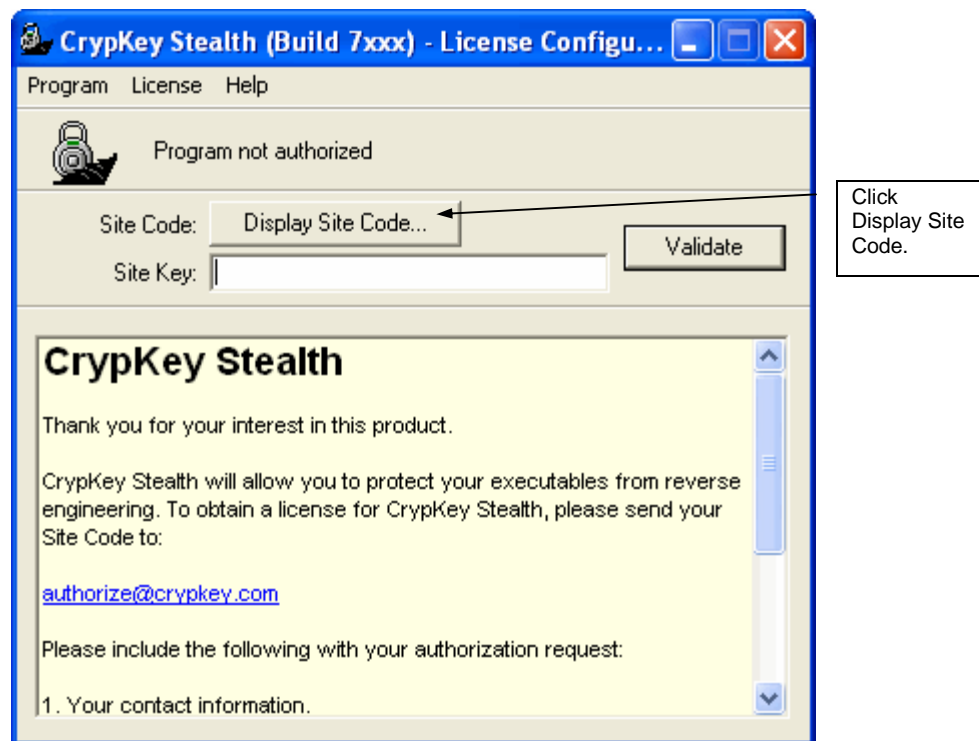


Figure 2 Stealth – Unauthorized License

3. Click on **Site Code** button to display the Site Code.

Note: When the message below is displayed, click **View License Agreement**. After reading the license agreement, click **Yes**:

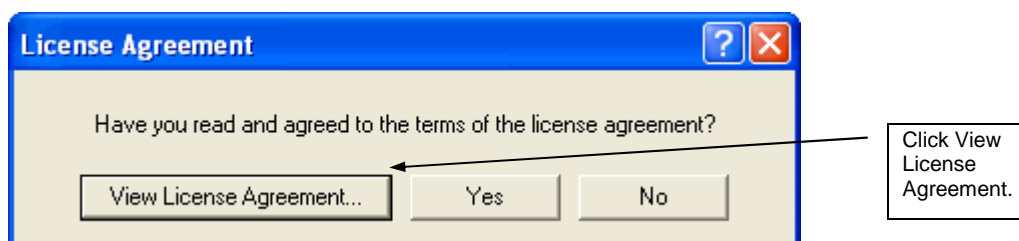


Figure 3 Stealth – License Agreement

4. If you have Outlook, click on the email link in the Stealth screen — an Authorization Request email opens for you:

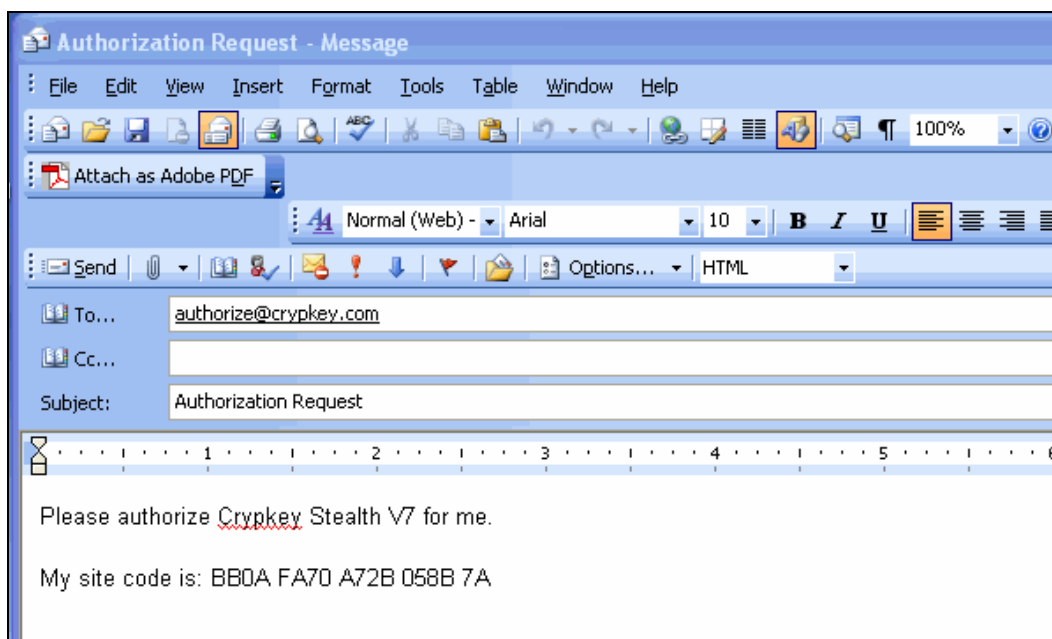


Figure 4 Stealth License Authorization email

5. If you do not have Outlook installed, you can manually send the Site Code by email to Authorize@CrypKey.com, or phone us to request a license.

Note: Please provide us with your contact information, company name, and customer service number, to obtain authorization.

6. Once you receive a reply from CrypKey, enter the Site Key you receive, into the Site Key field, and press the **Validate** button. This completes the licensing process and you can now use CrypKey Stealth.

Activate a License for the CrypKey Site Key Generator

1. See Authorizing Site Key Generator on page 57 for details.

Note: This manual describes activating SKG (*skw.exe*) for Normal mode operation; for steps in configuring SKG for Drive Share mode (to be used across a network), see *Configuring Site Key Generator for Network Use.rtf* in *C:\Program Files\CrypKey SDK\Build 7xxx\Documentation\Supporting Documentation*.

Transfer your old configuration file to the new install

1. Once SKG is authorized, copy the *skw.ini* file from your previous version of CrypKey, to the new location. e.g. C:\Program Files\CrypKey SDK\Build 7xxx.
2. You are ready to use CrypKey SDK 7 to upgrade your program.

Re-protect your application

At this point you must upgrade your application and install using the new CrypKey SDK 7 components.

Note: This manual focuses on upgrading your application to work in the **Normal** mode of operation. If your goal is to incorporate CrypKey Enterprise License Management (ELM), please refer to the *ELM User Manual* (located in C:\Program Files\CrypKey SDK\Build 7xxx\Documentation\Supporting Documentation\ELM.rtf) after you have completed this phase of the upgrade.

Re-Link and Recompile your application

1. Identify the CrypKey Library you have linked to in your product.
2. Replace the associated library, object, or assembly in your product with the updated version from the CrypKey 7 Installation folder. Refer to the CrypKey SDK 7 Quick Start Guide.pdf (located in C:\Program Files\CrypKey SDK\Build 7xxx\Documentation) for information on which libraries are available. Also refer to the sample projects in C:\Program Files\CrypKey SDK\Build 7xxx\Examples\Sample Code.

Stealth your new Executable (optional)

If you wish to take advantage of CrypKey Stealth technology (to prevent reverse engineering), use the *StealthUI.exe* application to protect your executable (see *Chapter 8: Protection*).

Distribution

This section discusses prerequisites for distributing your CrypKey SDK 7-protected product.

Upgrade the files

Replace the files in your installation folder with the newest ones from CrypKey. HDSN1.DLL is required no matter what OS. It is what controls the HDSN recognition.

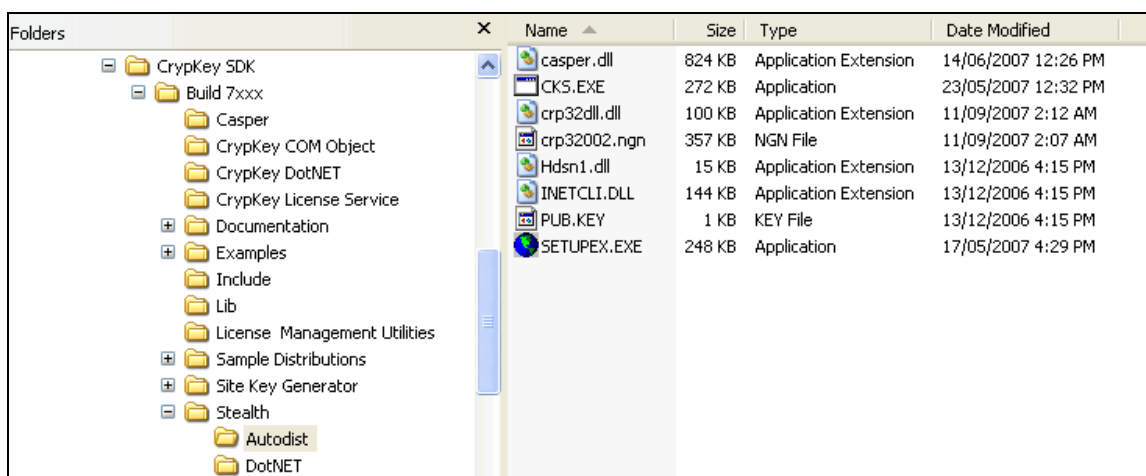


Figure 5 Replace files in your Installation folder

You will also need to include any CrypKey assemblies or dll's that your application links to.

Now you should be ready to distribute your application.

INSTALLATION CONSIDERATIONS – UPGRADE CRYPKEY LICENSE SERVER

Your install must reboot the user's computer after upgrading to the current version of the CrypKey License Service. It is important to upgrade the CrypKey License Service as part of your installation, otherwise CrypKey 7 will not function correctly. See below for details.

1. After copying the above files to the installation folder, run *SetupEx.exe* to upgrade the CrypKey License Service.

Note: The user must have administrator privileges to execute *SetupEx.exe*.

2. After the service has run, prompt the user to reboot the computer.

Note: You must reboot for the changes to take effect.

IMPLEMENTATION CONSIDERATIONS – .NET

Previous versions differed in the implementation and distribution. This information is provided for your reference. You can tell the build you are using from the CrypKey install directory. e.g. C:\CrypKey.6525, or from the .cnk text file that is generated by your product.

PRE 6504 IMPLEMENTATION

Distribution Files: crypkeynet.dll crp32001.ngn

C# implementation `CrypKeyNET6.CrypKey ck = new CrypKeyNET6.CrypKey();`

VB.NET implementation `Public ck As New CrypKeyNET6.CrypKey`

Post 6504 IMPLEMENTATION

Distribution Files crypkeynet2.dll crypkeynet.dll crp32dll.dll (note: crp32001.ngn is not required)

C# implementation CrypKey.CrypKeyNET2 ck = new CrypKey.CrypKeyNET2();

VB.NET implementation Public ck As New CrypKey.CrypKeyNET2

Note: CrypKeyNET.dll uses Crp32dll.dll. If an outdated copy of crp32dll.dll is in the system directory, CrypKeyNET.dll will use it rather than a current version in the same directory as CrypKeyNET.dll. If the DLLs are V7/V6 mismatched, it will crash.

Using an Existing License

A CrypKey SDK 7 application can use a version 6 license by one of the following methods:

1. Assume a license
2. Transfer a license.

Note: Alternatively you may wish to Issue a new license to the application.

Assume a License

This method works by simply installing your upgraded application, along with the new Crypkey files, into the application folder containing the version 6 license. When the version 7 application is run, it will immediately pick up the pre-existing license.

Transfer a License

The options for transferring a license are:

1. Use the DirectTransfer function in the version 6 application to transfer a license to a folder containing the version 7 application.
2. Use the AcquireLicense function in the version 7 application to pull a fixed license from a folder containing a version 6 application.

The end user process for the execution of these functions will depend on whether you have included this functionality in your applications, and how it is applied.

Issue a New License

To issue a new license, go through the process of obtaining a Site Code from the end user, and then send a new Site Key to that end user.

Chapter 3: Installation

This chapter provides details on how to install CrypKey SDK 7 and its associated CrypKey License Service on your system.

The CrypKey License service must be installed to use a CrypKey-protected application in Normal mode.

In this chapter you will find step-by-step instructions on how to:

- install CrypKey SDK 7
- test CrypKey SDK 7 (using a demo program).

Install CrypKey SDK 7

The installation procedure applies to both new and upgraded software. If you are upgrading, see *Chapter 2: Upgrading* for additional information.

Procedure for Installing CrypKey SDK 7

1. When you have completed the download of the installation files to your computer, double-click the executable file *CrypKeySDKV7xxx* (or whatever the latest version may be).
2. The system displays a sequence of standard installation windows.

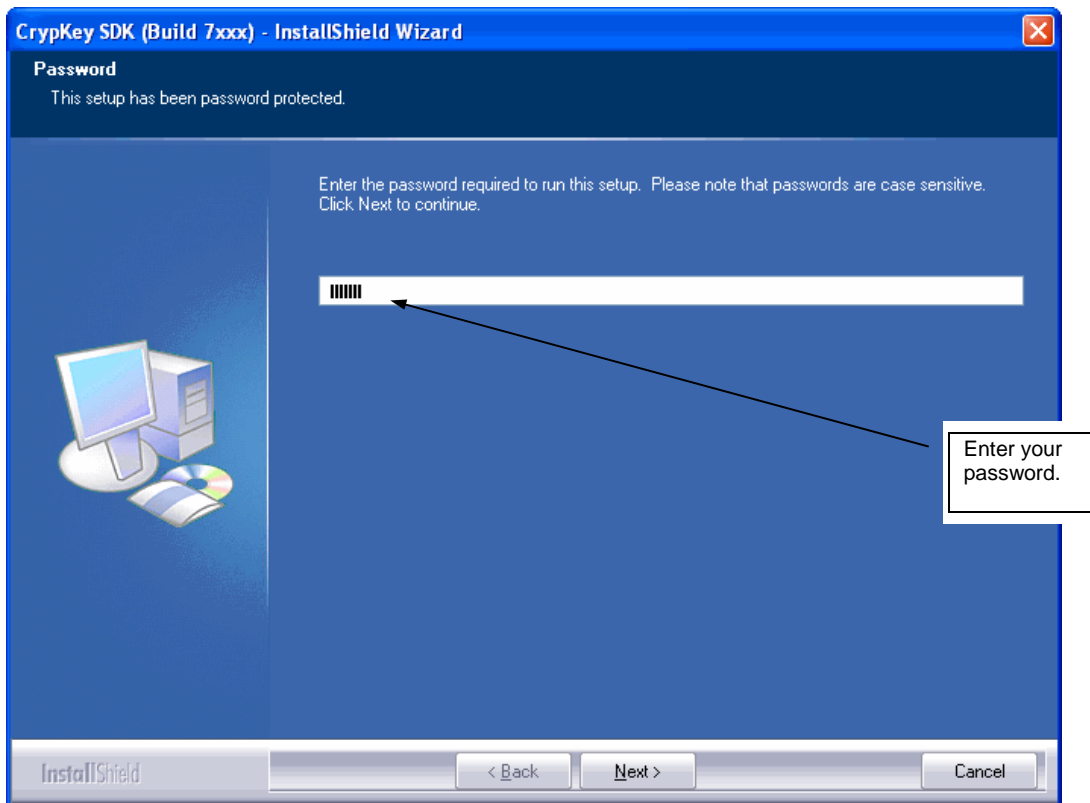


Figure 6 Install screen 1 – Enter Password

3. Enter the installation password supplied by CrypKey.

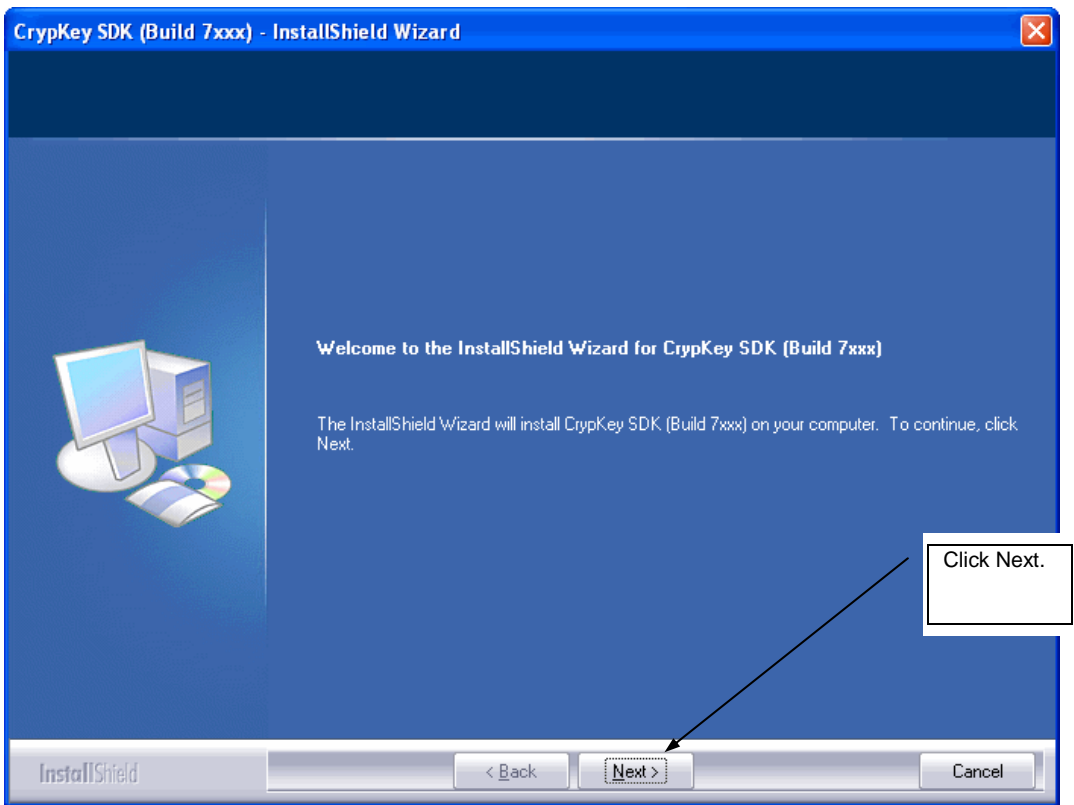


Figure 7 Install screen 2 – Welcome to Wizard

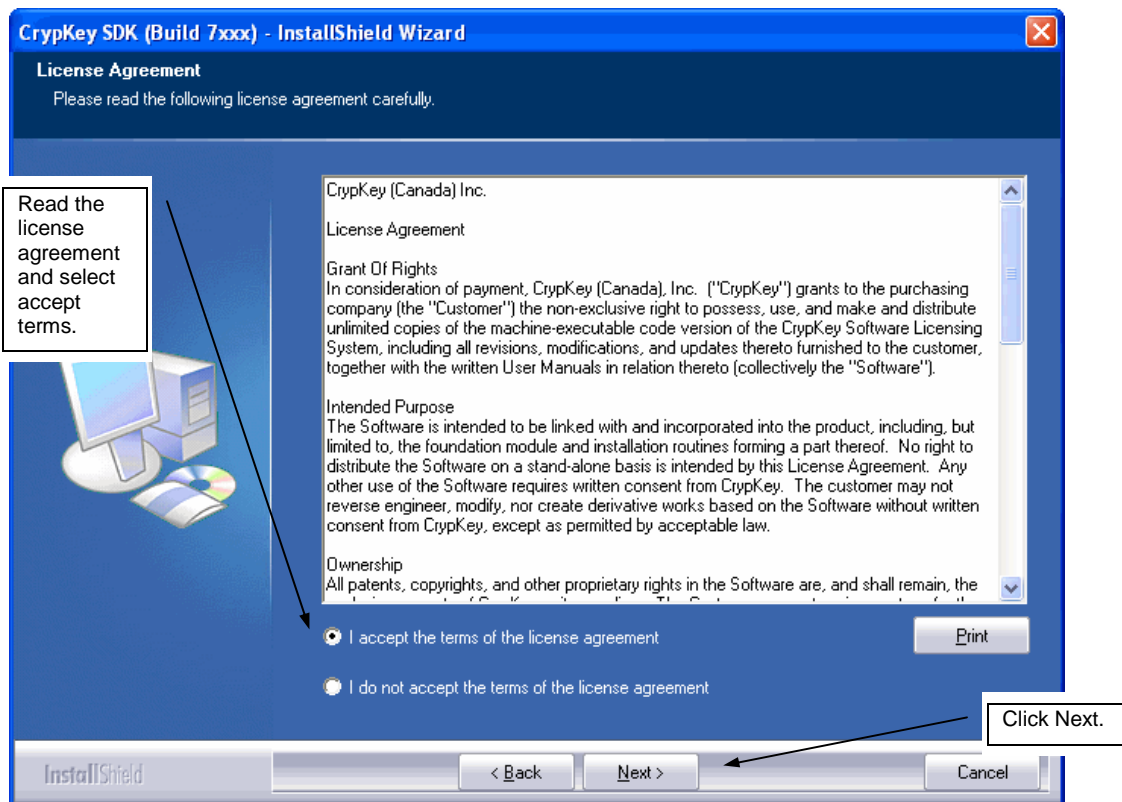


Figure 8 Install screen 3 – License Agreement

4. Read the license agreement.
5. Select “I accept the terms”.
6. Click **Next**.

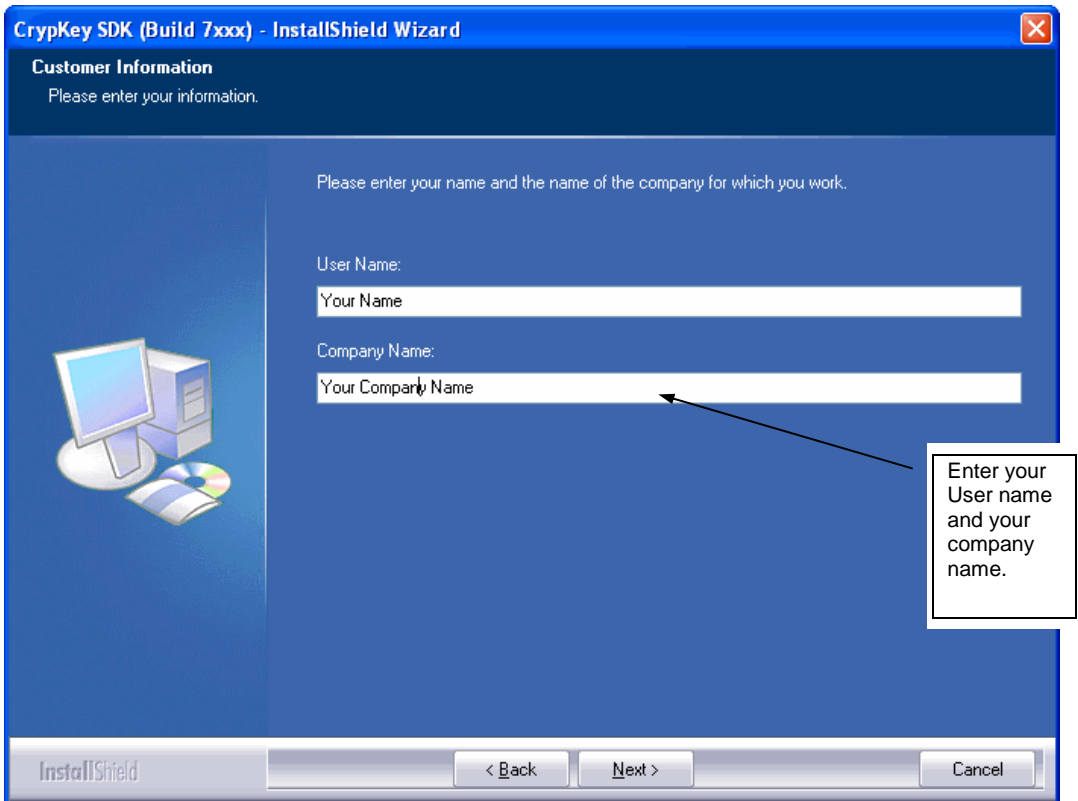


Figure 9 Install screen 4 – User/Company Information

7. Type in your User name and Company name.

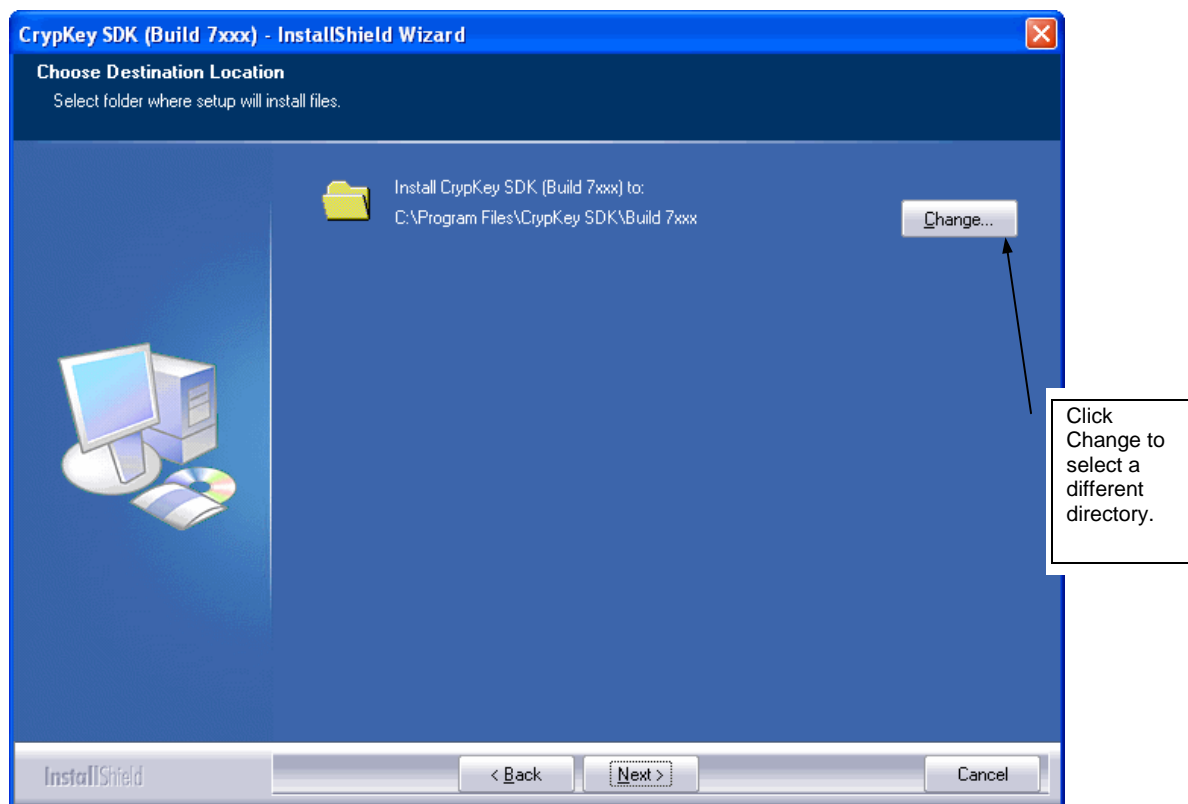


Figure 10 Install screen 5 – Directory location

8. Accept the default directory provided by the installation program, or specify which directory to install CrypKey SDK 7 in. If the directory does not already exist, the installation program creates it for you.

Note: If you select another install directory, note that the directory paths for finding files and documentation given in the rest of this manual will also be different.

9. Click **Next**.

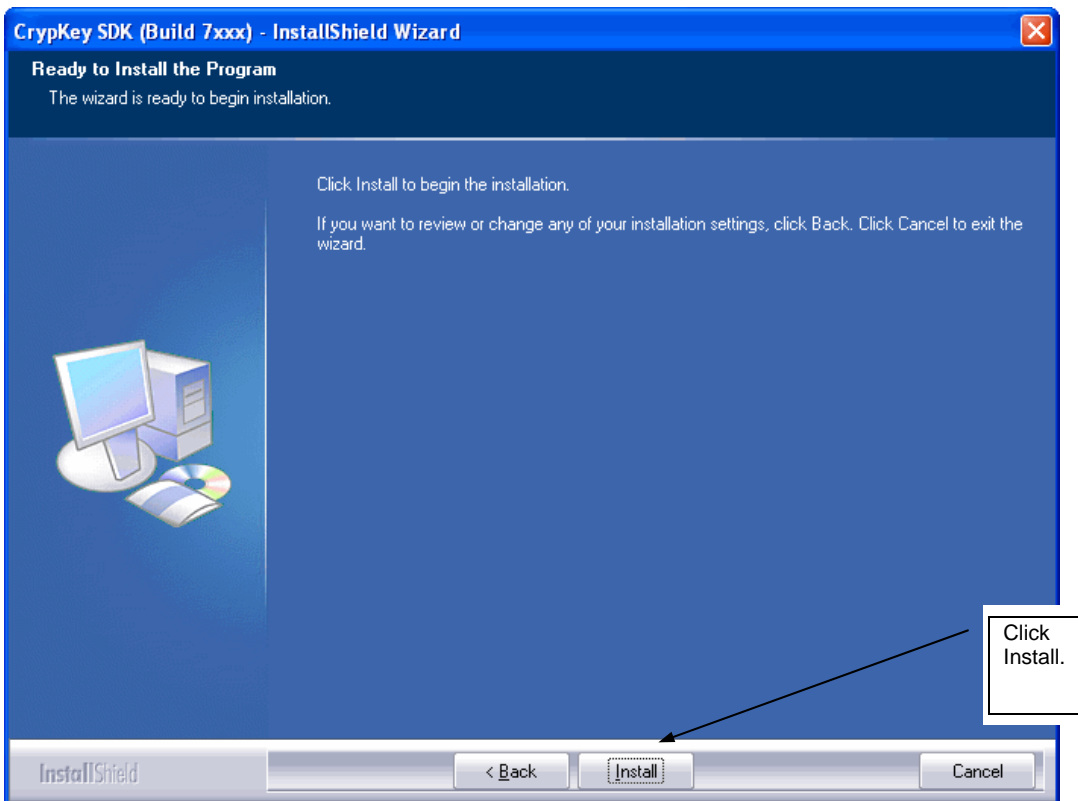


Figure 11 Install screen 6 – Start Install

10. Click **Install** (or click **Back** to change a previous screen's information).

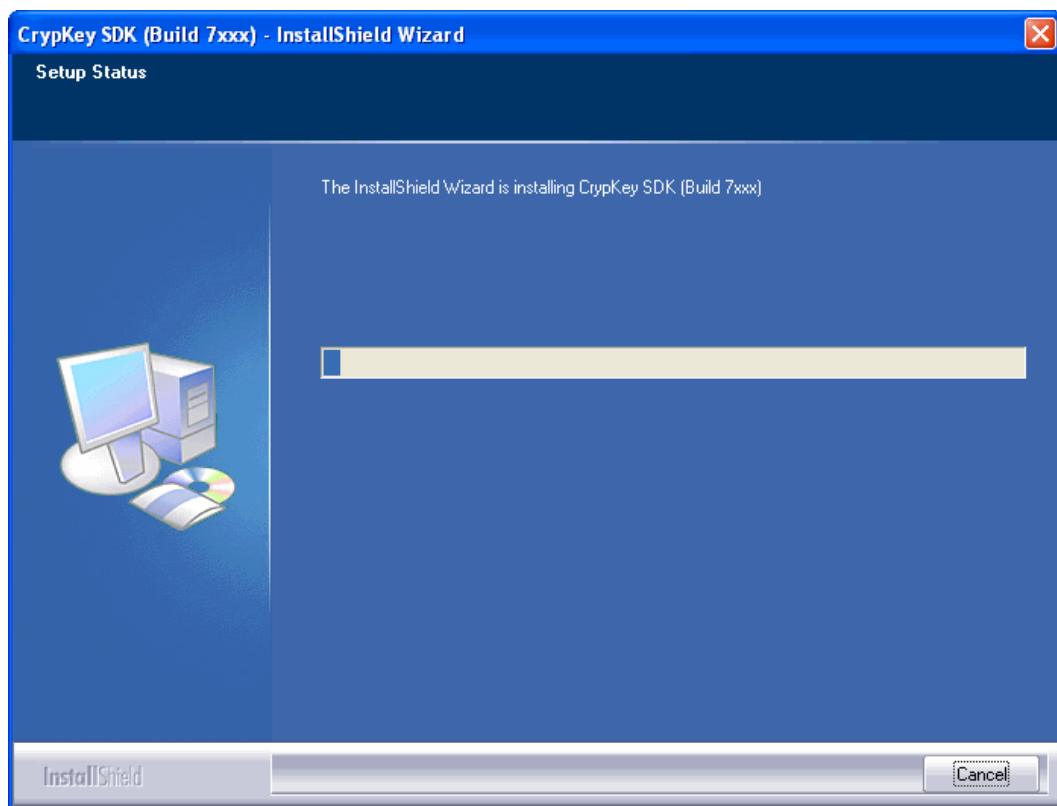


Figure 12 Install screen 7 – Installation in progress

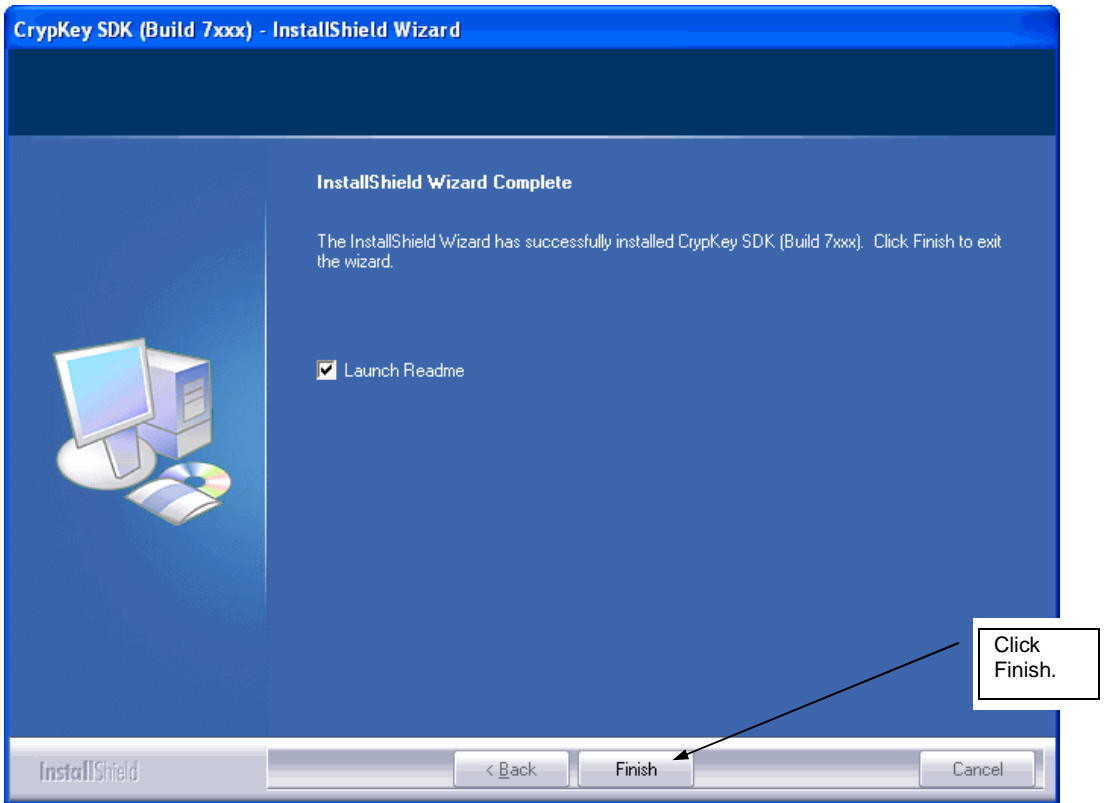


Figure 13 Install screen 8 – Installation complete

11. You may open the Readme file now or choose to view it later.

We strongly recommend that you read the Readme file immediately, as it may contain last-minute details that were not available in time to be included in this manual.

Readme Files

The *Readme!.chm* file is located in your CrypKey SDK\Build 7xxx\Documentation directory. They contain important last-minute information about CrypKey SDK 7, SKG, Stealth, etc. that was not available in time to be included in the manual. You should read it after installing CrypKey on your system.

CrypKey SDK 7 DIRECTORIES

The contents of each installed directory follows:

CrypKey COM Object

This directory contains the CrypKey COM object. This is the easiest library to integrate into VB and Visual C. This directory also includes the documentation and VB and VC sample programs.

CrypKey DotNET

This directory contains the CrypKey .NET assembly. This is the easiest library to integrate into the .NET languages (VB.NET, C#.NET, and C++.NET). This directory also includes the documentation and C#.NET and VB.NET sample programs.

Examples

This directory contains the compiled samples and sample code.

License Management Utilities

This directory contains the Enterprise utilities.

Include

This directory contains the include files you would use if using C or C++. Currently, there is only `cryptkey.h`.

Lib

This contains a number of different versions of the CrypKey library.

Documentation

This directory contains documentation about CrypKey SDK 7 and its utilities and features.

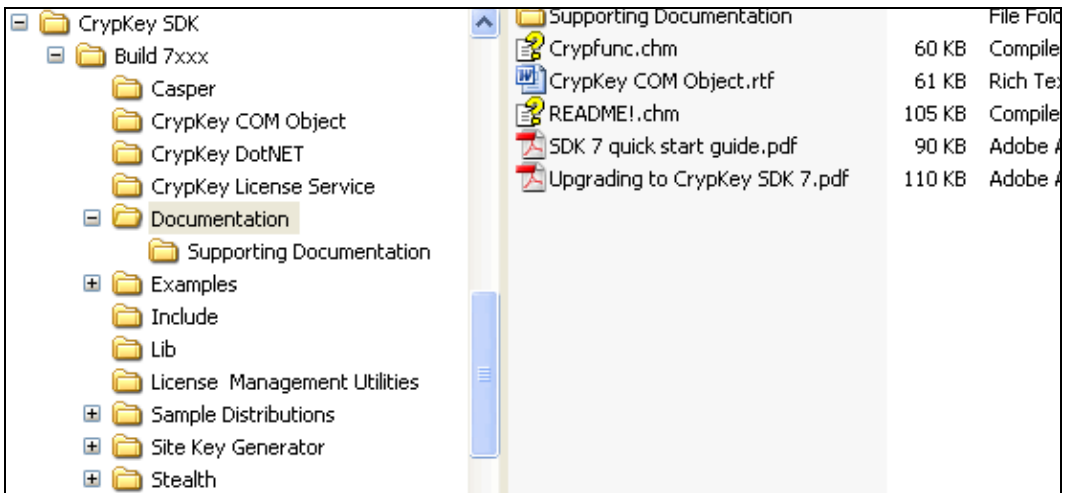


Figure 14 Documentation directory

The Supporting Documentation directory contains the following documents:

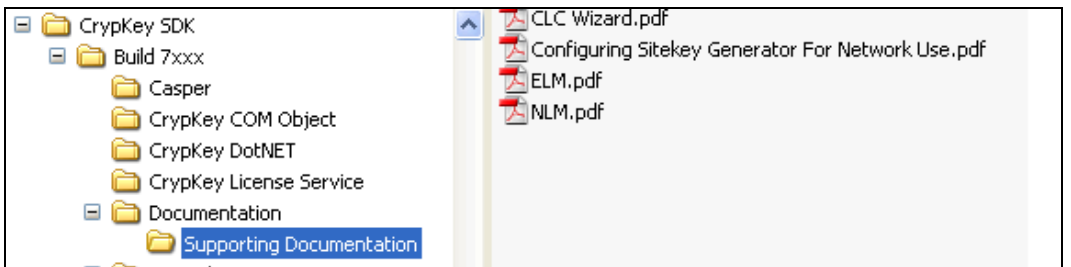


Figure 15 Supporting Documentation directory

Documentation:

- Crypfunc.chm – contains information about CrypKey SDK 7 function calls.
- CrypKey COM Object – tools that have proven useful in configuring protection for applications and issuing licenses.
- README!.chm – contains software release notes about the available documentation; operating modes for CrypKey SDK 7; the test suite; server and client configuration, and troubleshooting.
- CrypKey SDK 7 Quick Start Guide – quick reference for getting started with CrypKey SDK 7.
- Upgrading to CrypKey SDK 7.pdf – how to upgrade from your current version to CrypKey SDK 7; also included in this manual in Ch. 2.

- CrypKey SDK .doc – This manual. You may wish to use some parts in your software documentation for your customers.
- .NET IP Protection white paper – see online at <http://www.crypkey.com/whitepapers.asp>.
- CLC Wizard.rtf – Client Licence Configuration.
- Configuring Site Key Generator for Network Use.rtf – how to configure your Site Key Generator for use on a network.
- ELM.rtf – Enterprise License Manager.
- NLM.rtf - Network License Manager.

CrypKey License Service

This directory contains the service needed for all CrypKey platform usage, as well as for using CrypKey Enterprise Network Licensing.

Five server types are supported:

- Windows NT, 2000, XP, 2003 and Vista

Examples

This directory contains complied samples and sample code from different platforms. Some samples are contributed by kindhearted CrypKey customers (and we always welcome more). These samples are a good starting point and can always be improved upon.

Note: The test demo program, *example.exe*, is in this directory.

Sample Distributions

This directory contains sample distributions for Normal mode, Drive Share mode, and Client/Server mode.

Site Key Generator

This directory houses the CrypKey Site Key Generator. For those updating from previous versions, we think you will like the improvements.

Stealth

This directory contains CrypKey Stealth™:

- STEALTHUI.EXE — user interface version

- STEALTHCMD.EXE – command line version.

CrypKey License Service Installation

This section pertains to native Win32 programs running on the NT File System (NTFS). The CrypKey License Service must be installed on Windows NT/2000/XP/2003/Vista operating systems.

The CrypKey License Service manages licenses. It is required for either standalone or network installations.

To run a CrypKey-protected Win32 program from a local drive on any of the above platforms, you must install the CrypKey License Service on that machine. The CrypKey License Service is required because the Operating Systems do not allow programs to directly access the computer hardware, which CrypKey must do. For the installation procedures, see *Installing the CrypKey License Service* on page 103.

If you are currently running CrypKey on a Windows personal computer, you already have the CrypKey License Service installed. If you want to use the License Service with the installation of your application, you can find it in C:\Program Files\CrypKey SDK\Build 7xxx\CrypKey License Service.

Stealth can install the License Service automatically for you. For more details, see Chapter 6: Protecting Your Software with Stealth.

Note: CrypKey does not support custom installations for CrypKey License Service.

Libraries and Distribution Files

Following are lists of 32-bit libraries from which you must distribute the appropriate files with the product that you provide to your customer. For convenience, both .LIB and .DLL files are listed here, although .LIB files are not distributed with your application. The .DLL files that must be distributed are indicated below; others are for distribution as required by your application. These libraries reside in your C:\Program Files\CrypKey SDK\Build 7xxx\Lib directory.

For more details on linking your application to libraries, see *Chapter 14: Function Reference, Linking* on page 89.

For more details on libraries to be distributed for various platforms, see *Chapter 14: Function Reference.WIN32 Libraries*

The following libraries are found in ~\CrypKey\Lib\.

Refer to the following table for files required to be distributed.

Table 1: Files required for Distribution with 32-bit Applications

File	Notes
CRP32002.NGN	This file must be distributed with your program in the same directory as your 32-bit application, or in the directory that holds your license.

Files required for various purposes:

CRP32M60.LIB — Microsoft VC++ 6.0 Win32 library (multiple thread Static RT Libraries)

CRP32M70.LIB — Microsoft VC++ 7.0 Win32 library (multiple thread Static RT Libraries)

CRP32M80.LIB — Microsoft VC++ 8.0 Win32 library (multiple thread Static RT Libraries)

CRP32D60.LIB — Microsoft VC++ 6.0 Win32 library (multiple thread Dynamic RT Libraries)

CRP32D70.LIB — Microsoft VC++ 7.0 Win32 library (multiple thread Dynamic RT Libraries)

CRP32D80.LIB — Microsoft VC++ 8.0 Win32 library (multiple thread Dynamic RT Libraries)

CRP32S60.LIB — Microsoft VC++ 6.0 Win32 library (single thread RT Libraries)

CRP32S70.LIB — Microsoft VC++ 7.0 Win32 library (single thread Static RT Libraries)

CRP32DLL.DLL — Win32 DLL (multiple thread RT libraries)

CRP32DLL.LIB — Microsoft VC++ 4.2 import lib for CRP32DLL.DLL

CRP32BMT.LIB — Borland C++ 4.51 Win32 library (multiple thread)

CRP64M80.LIB — Microsoft VS8 x64 Win64 library (multiple thread static RT libraries)

CRP64D80.LIB — Microsoft VS8 x64 Win64 library (multiple thread dynamic RT libraries)

CRP64DLL.DLL — Unified Windows x64 DLL (multiple thread dynamic RT libraries)

The following files are found in other directories:

- ~\CrypKey\CrypKey COM Object:

CRYPKEYCOM.DLL — COM object for all compilers that can use COM objects.

- ~\CrypKey.60\CrypKey DotNET:

CRYPKEYNET.DLL — .NET assembly for all compilers that can use .NET assemblies.

Connectivity to Microsoft Access

The rules enabling CrypKey to work with Microsoft Access are the same for both Windows 97 and Windows 2000:

1. This directory must be on the path or explicitly named.

You can achieve this in one of three ways:

- a. By putting the DLLs in a windows or system directory.

Note: It is always better to keep your files out of common directories.

You can avoid using the sys or win directories by choosing one of the other methods.

- b. By adding your directory to the path and rebooting.
- c. By always putting your files in the same directory, e.g. C:\Dan\Danstuff, and changing the declares in VB accordingly. For example from "Crp32dll.dll" to "C:\Dan\Danstuff\Cryp32ddll.dll"

Each method has merits and drawbacks, so it is up to you to choose the one to use.

Chapter 4: Getting Started

CrypKey SDK 7 offers new utilities to make software licensing easier and more powerful.

CrypKey SDK 7 (CrypKey Enterprise) allows flexible and on demand licensing of software over a network, without resorting to the use of a drive share mode, as with previous versions. This means you can:

- limit the number of users and computers using your software; and
- transfer licenses based on individual users and computers over multiple domains on the same network.

New in CrypKey SDK 7

New features in CrypKey SDK 7 offer enhanced flexibility:

- With little or no programming, any CrypKey-protected program can easily act as either a License Server or a License Client
- Automatic restart of Servers on reboot
- Supports an unlimited number of redundant servers
- Transfer of licenses across the network without any special directory read-write shares.
- Specify the number of floating licenses to transfer.
- Add licenses to a server that has existing licenses. This means you can borrow licenses from a “license pool” and return them at a later time.

CrypKey SDK 7 Utilities

CrypKey SDK 7 includes three new utilities in one streamlined product to make configuring server licenses easier and quicker:

- Enterprise License Manager (ELM)
- Network License Manager (NLM)
- Client License Configuration (CLC) wizard

These utilities provide your customer with the ability to configure how their licenses are used on a network. Use your own discretion to decide whether to provide these utilities to the customer: be aware that there is a potential for a license to become disabled if the user is not familiar with the purpose and operation of the new utilities. Please refer to the user manuals for these utilities for more detailed information.

ENTERPRISE LICENSE MANAGER

The ELM offers the ability to transfer licenses and view license usage on a network. It can be used to modify network license configurations.

NETWORK LICENSE MANAGER

The NLM provides the user with the ability to create network license configurations and manage network licenses. The NLM can be launched from the Control Panel, and provides a button to launch the ELM.

CLIENT LICENSE CONFIGURATION

The CLC wizard provides the user with the ability to generate client configuration files.

Installing the Utilities

For these applications to work correctly together, ensure that the folder containing the network license contains the *ckconfig.dll* and the Enterprise License Manager executable and the .ced file for your application. The *ClientLicenseConfig.exe* is a stand-alone application that can be saved to the license folder on the client machine.

The ClientLicenseConfiguration application can be installed on the client computers if you want the end user to have the ability to control which servers or domains they can access for a license.

Who Should Use the Utilities

CrypKey recommends that the network administrator or IT person be given the responsibility of using these utilities to create the client and server configuration files. The client configuration file (*cec.ini*) can then be distributed to the client computers.

Note: This document assumes the reader has the necessary skills and knowledge to understand the information within this guide.

Here is a summary of the steps involved in implementing CrypKey 7:

1. Determine the mode of operation for your application.
2. Determine which library to link to.
3. Implement CrypKey functions in your application.
4. Stealth protect your application.
5. Determine which CrypKey files you need to distribute.
6. Configure distribution files.
7. Modify your install to distribute CrypKey files.
8. Distribute your application.
9. License the application on your customer's computer.

Step 1. Determine the mode of operation for your application

To begin, you must decide how your application will be used. This information will be used to determine how to configure and distribute files:

- If your application is intended to run on individual computers, then you want to run CrypKey in Normal mode.
- If your application is located and launched from a shared folder on the network, you need to use drive share mode.
- If your application is installed and run from the client computer, you need to use Client/Server License mode.

Note: CrypKey 7 Client/Server License mode offers superior performance over drive share mode, and does not require any specific client user permissions. It is the recommended mode of operation for network applications. Ideally you should make whatever changes you need to allow your application to launch from the client, so that you can take advantage of the CrypKey 7 true client/server operation.

Step 2. Determine Which Library to Link to

Please refer to the *CrypKey SDK 7 User Manual* for more information about specific libraries.

Table 2: Link to Library

Compiler	Library
Visual Studio 6 C++	crp32d60.lib dynamic multithreaded crp32s60.lib single thread static crp32m60.lib multithread static

Visual Studio 7 (2003) C++	crp32d70.lib dynamic multithreaded crp32s70.lib single thread static crp32m70.lib multithread static
Visual Studio 8 (2005) C++	crp32d80.lib dynamic multithreaded crp32s80.lib single thread static crp32m80.lib multithread static
Visual Basic	CrypKeyCom.dll * Crp32dll.dll
C#	CrypKeyNet2.dll * CrypKeyCom.dll Crp32dll.dll
VB.NET	CrypKeyNet2.dll * CrypKeyCom.dll Crp32dll.dll
ASP.NET	CrypKeyNet2.dll * CrypKeyCom.dll Crp32dll.dll
Borland C++ Builder	crp32bmt.lib CrypKeyCom.dll Crp32dll.dll
Delphi	CrypKeyCom.dll Crp32dll.dll
VBA (Access,Word,Excel)	CrypKeyCom.dll Crp32dll.dll
Powerbuilder, WinDev, Jade, FoxPro, etc	CrypKeyCom.dll will work with any compiler capable of referencing COM dlls. Crp32dll.dll will work with any compiler capable of implicit or explicit runtime linking to standard dlls.

* Preferred libraries for ease of implementation.

Step 3. Implement CrypKey functions in your application

Your application should check for a valid license when it loads. The code to do this is often placed in some sort of splash screen to keep the user occupied while the license check completes. There are many ways to integrate CrypKey into your application to meet your various requirements.

Please take a look at the samples provided with your installation. In many cases you can copy, paste, and modify the code from our example project to get up and running.

You can use the example license to do your testing, but will need to contact CrypKey to get your own developer keys on completion of testing with the example files. The Master Key, User Key, and license filename will be passed as parameters into the InitCrypKey function. During the implementation and testing, you will need to configure the Site Key Generator so that you can issue licenses to the application.

Step 4. **Stealth protect your application**

Use the *StealthUI.exe* utility to protect your executable from reverse engineering. Do this BEFORE digitally signing your executables.

.NET executables, as well as exe/dll combinations are also supported.

If you encounter any errors during this process, please contact Support@CrypKey.com.

Note: CrypKey cannot protect a .NET dll independently. If you would like to protect the .NET dll it must be combined with an exe. The exe and dll(s) will be protected using StealthUI and result in a new exe that would be distributed in place of the original exe and dll. The process to protect an exe/dll combination is as follows:

Procedure to Protect exe/dll Combination

1. Edit the C:\Program Files\CrypKey SDK\Build 7xxx\Stealth\ input.xml file to contain the name of your exe and dlls:

```
<DotNetJoiner>

    <base>DotNetBase.exe</base>
    <host>DotNetHost.exe</host>
    <input>main.exe</input>
    <input>mod1.dll</input>
    <input>mod2.dll</input>
    <input>mod3.dll</input>
    <output>input.exe</output>
</DotNetJoiner>
```

2. From the command line or .bat file launch the initial conversion process:


```
dotnetjoiner.exe input.xml
```

This will create the *input.exe* file as specified in the input.xml above. The new file is now a C++ exe containing your merged exe and dlls. Now it is ready to be protected.

3. Start the stealthui.exe program in C:\Program Files\CrypKey SDK\Build 7xxx\STEALTH\
4. Specify C:\Program Files\CrypKey SDK\Build 7xxx\STEALTH\DotNet\input.exe as the file to compress.
5. Specify the name of the location of the output file.
6. Begin compression.

Once the compression (stealth process) is done, you should be able to run the output exe alone.

Note: Since the dll is now merged into the output exe, you do not need to distribute the dll or original exe. If you do put the dll in the directory with the output.exe, you will get an error.

Step 5. Determine which CrypKey files you need to distribute:

Table 3: OS Specific Files

NT/XP/W2K/2003/Vista
Setupec.exe
CKS.exe
Hdsn1.dll
CRP32002.NGN

Table 4: CrypKey Libraries

If compiled with...	Then distribute....
crp32dll.dll	crp32dll.dll crp32002.ngn
crypkeycom.dll	crypkeycom.dll crp32002.ngn
crypkeynet2.dll	crypkeynet2.dll crp32dll.dll crp32002.ngn

If compiled with...	Then distribute....
Any other lib	crp32002.ngn

Table 5: Mode Specific Files

Mode specific	
Normal	No additional files required.
Drive share	No additional files required.
Client/Server	<p>Additional files required on client:</p> <p>cec.ini ClientLicenseConfiguration.exe (optional)</p> <p>Additional files required on server:</p> <p>.ces.ini enterprise license manager.exe .ced file</p> <p>CRP32002.NGN</p> <p>ckconfig.dll</p>

Note: Optional files are only required on client to support server mode transfers (see *ELM User Manual* for more information). For more information about the optional utilities and their purpose, please refer to the section *CrypKey SDK 7 Utilities* at the beginning of this manual.

Step 6. Configure Distribution files

The only time you need to configure distribution files is if you are supporting CrypKey 7 Network License mode.

The client needs a file called *.cec.ini*, where the status = enabled. The server needs a file call *.ces.ini*, where the status = enabled. Note: these files are preceded by the name of your license, e.g. *license.cec.ini*.

These files can be configured in one of two ways:

- a. Use the Enterprise License Manager (See *ELM User Manual* for more information); OR
- b. Use the Network License Manager (NLM) to create a server configuration for a license, and the Client License Configuration (CLC) wizard to create a client license configuration;

Note: We do not recommend manually editing the ini files.

Here is a sample of what the file could look like:

Client: example.cec.ini

[INCLUDED_DOMAINS]

E1=PRIMARY

[INCLUDED_MACHINES]

E1=CRYPKEY04

[GENERAL]

STATUS=Enabled

Server: example.ces.ini

[INCLUDED_DOMAINS]

E1=ALL

[INCLUDED_MACHINES]

E1=ALL

[INCLUDED_USERS]

E1=ALL

[GENERAL]

STATUS=Enabled

Step 7. Modify your install to distribute CrypKey files

You must configure your installation to do the following, based on the mode of operation you have selected.

Table 6: Mode Specific Tasks

Mode	Task
Normal	<ul style="list-style-type: none"> - all files installed into local application folder - install must include <i>SetupEx.exe</i> and <i>CKS.exe</i> and must launch <i>SetupEx.exe</i> to install the CrypKey License Service. The service can only be installed by an administrator.
Drive Share	<ul style="list-style-type: none"> - all files installed into drive share folder on server - install must include <i>SetupEx.exe</i> and <i>CKS.exe</i> and must launch locally on the server <i>SetupEx.exe</i> to install the CrypKey License Service. The service can only be installed by an administrator.
CrypKey 7 Network License (client/server)	<p>Client Side</p> <p>protected application is installed, along with:</p> <ul style="list-style-type: none"> - OS Specific Files - CrypKey Libraries - Mode Specific Files <p><i>Setup.exe</i> and <i>CKS.exe</i> NOT REQUIRED.</p> <p>Server Side:</p> <p>protected application is installed, along with:</p> <ul style="list-style-type: none"> - OS Specific Files - CrypKey Libraries - Mode Specific Files <p>Run <i>SetupEx.exe</i> locally from the server, to install the CrypKey License Service.</p> <ul style="list-style-type: none"> - Ckconfig.dll

Note: The protected application does not have to be installed on the server. It is only suggested here as it will logically contain the interface to manage your license requirements. If you do not install your application on the server, you need some other method of authorizing a license at the server. You can do this by creating a separate license manager to provide the network administrator with an interface to the licensing functions. Essentially, a license manager is an application that provides the user with the necessary interface components to enter a Site Key and initialize a license.

Regardless of how you manage this step, you should now be at the point where your customer can install your application.

Step 8. Distribute your application

Send your protected software out to your clients in whatever distribution method you prefer.

Step 9. License the application on your customer's computer

If your application starts up with a trial license, the customer is ready to start using your program. Optionally, you may want to issue them a license using the Site Key Generator.

If you need further assistance or have more questions after reading this document, please contact CrypKey Support at Support@CrypKey.com.

Chapter 5: Demo Programs

Demo programs allow you to test and evaluate the CrypKey system in operation.

Test the full functionality of CrypKey SDK 7 using our demonstration program *wexample.exe*. This is a Win32-based CrypKey-protected program that demonstrates all of the basic features of the CrypKey system. The complete source code is included in the CrypKey SDK 7 installation (see Sample Code).

After installation, your Windows Program Manager contains a new program group with several new icons, as shown:

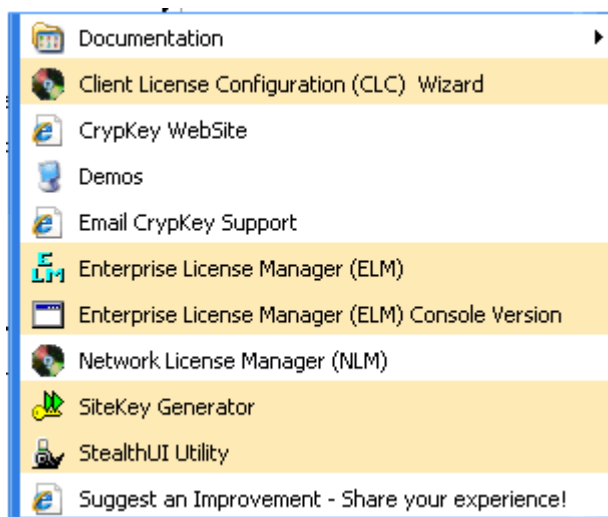


Figure 16 CrypKey SDK 7 Program Group

As in the normal authorization process for your products, the Site Key Generator is an essential element of the EXAMPLE demonstration. For more details on how to use the Site Key Generator, see *Chapter 6: SKG*.

Procedure

1. Using the **Start** button, display the CrypKey program group (see Figure 16).

- Click the **Demos** option, then **Compiled Demos**. Your computer displays a list of files, partially shown below:

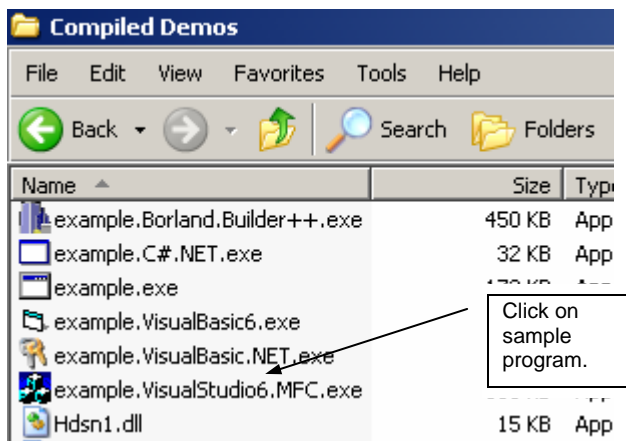


Figure 17 Sample Compiled Code

Note: Your own product does not need to be written in C++ as this particular example is. CrypKey SDK 7 will perform in the same way when paired with products written in the other supported languages shown.

- In the above window, double-click the `example.VisualStudio6.MFC.exe` to start the program.

4. Your computer displays the CrypKey TESTAPP window as shown:

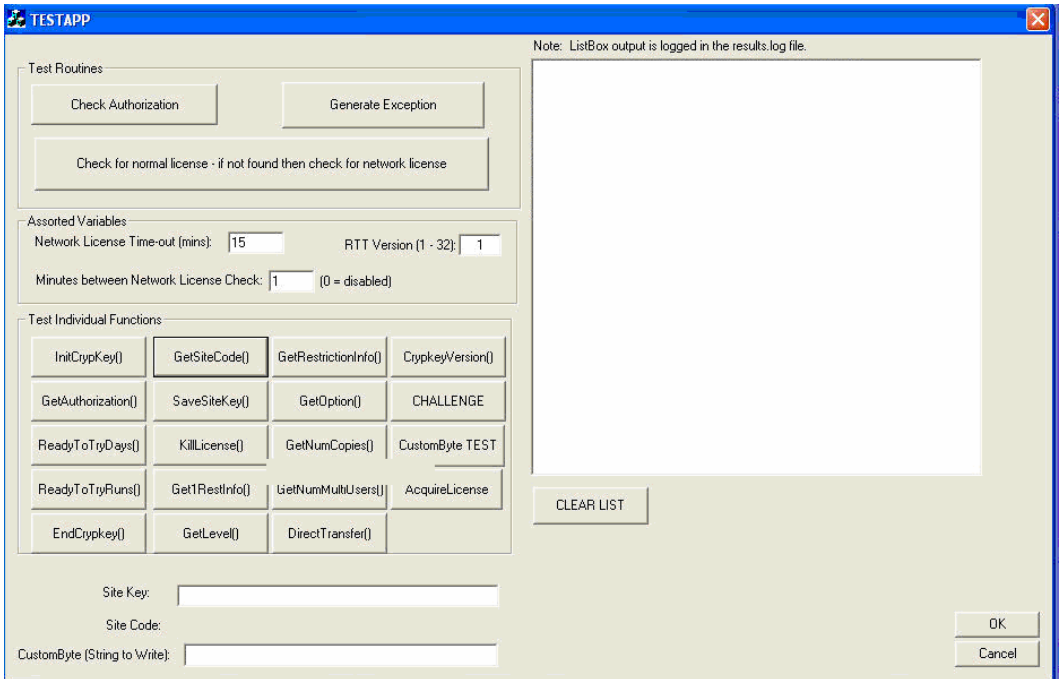


Figure 18 Test Application window

Note: The buttons in the window perform a number of CrypKey functions. You can use these buttons to troubleshoot the operational status of your CrypKey installation.

5. Click the following buttons in sequence: InitCrypKey(), GetAuthorization() and GetSiteCode(). As you click these buttons, the TESTAPP application window shows:

InitCrypkey() called ...
INIT OK

GetAuthorization() called...
Oplevel = 0
AUTHORIZATION NOT PRESENT

GetSiteCode() called ...
SITE CODE OK
5437 1C33 4595 1CC4 D5

The TESTAPP window appears as follows:

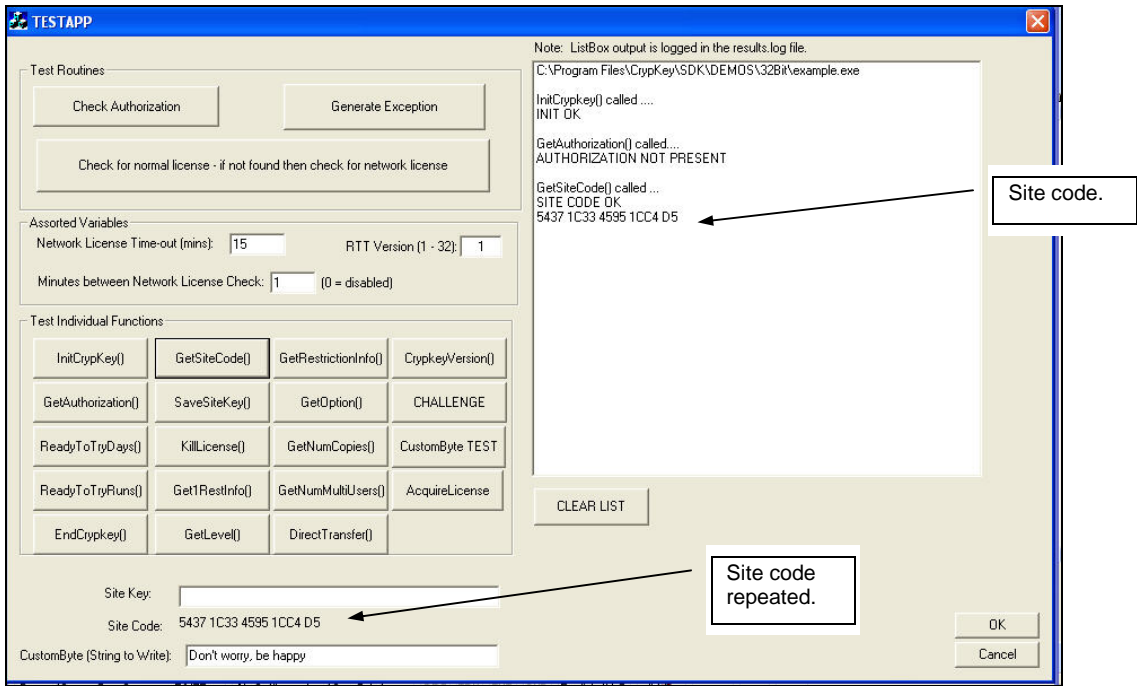



Figure 19 Test Application window with Site Code

In the above window, note that the Site Code, shown in the returned messages panel, is also displayed below the Site Key field. For explanations of how the Site Code and Site Key are generated, and their purpose, see *Chapter 6: SKG*.

6. Select the Site Code located below the Site Key field and press Ctrl+C to copy the Site Code to the clipboard.
7. Double-click the **Site Key Generator** icon  in the Start/Programs menu to start the *skw.exe* program. You can use the Alt+Tab keys to toggle between the two program windows that are now active (TESTAPP and the Site Key Generator).

Note: To place things in context . . . in the TESTAPP window, you are playing the role of your customer by opening an application that you as the vendor have protected using CrypKey. In the Site Key Generator window, you are acting as yourself by receiving the Site Code generated by the CrypKey program bundled with *example.VisualStudio6.MFC.exe*, and entering the code in order to generate a Site Key.

8. In the Site Key Generator window, press Ctrl+V to paste the Site Code (which you copied to the clipboard in step 6) into the Site code field.
9. Then click the **Check** button to verify the Site Code. If the code is valid, the window shows the following:

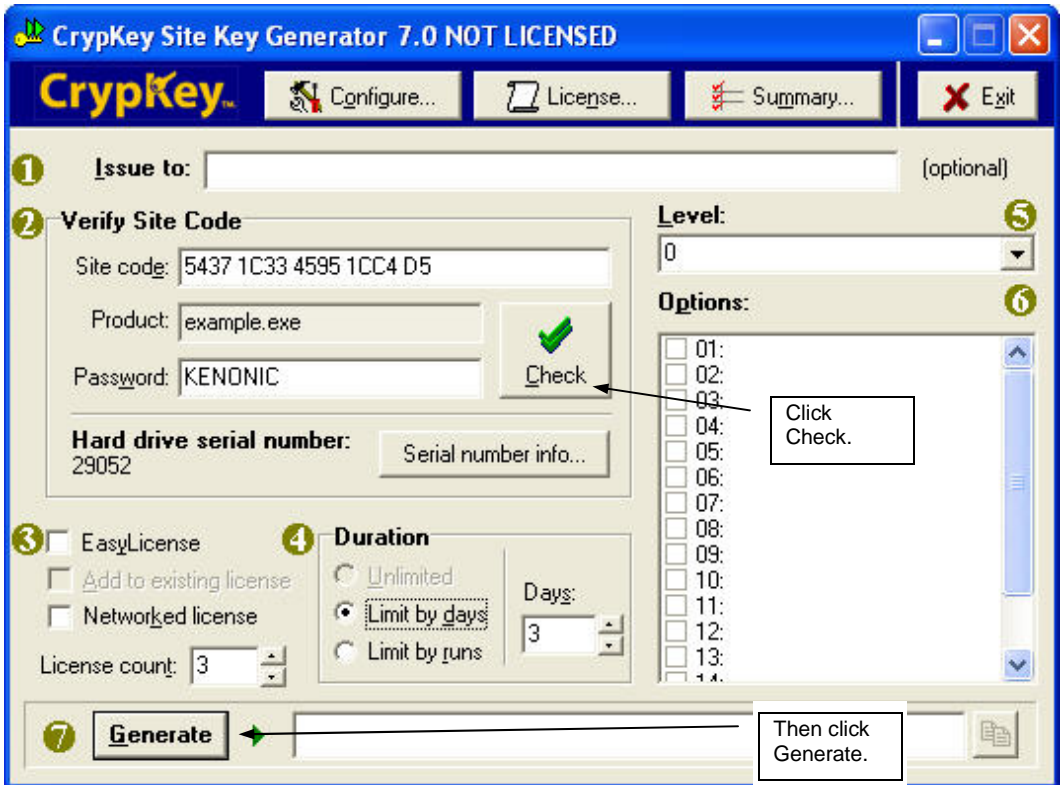


Figure 20 Validated Site Code

10. In the above window, enter the level, options, license type, and license restriction as desired. (These items are explained in detail in *Chapter 6: SKG*.)
11. Click the **Generate** button. The program generates a Site Key based on the Site Code and places it in the text box beside the **Generate** button.

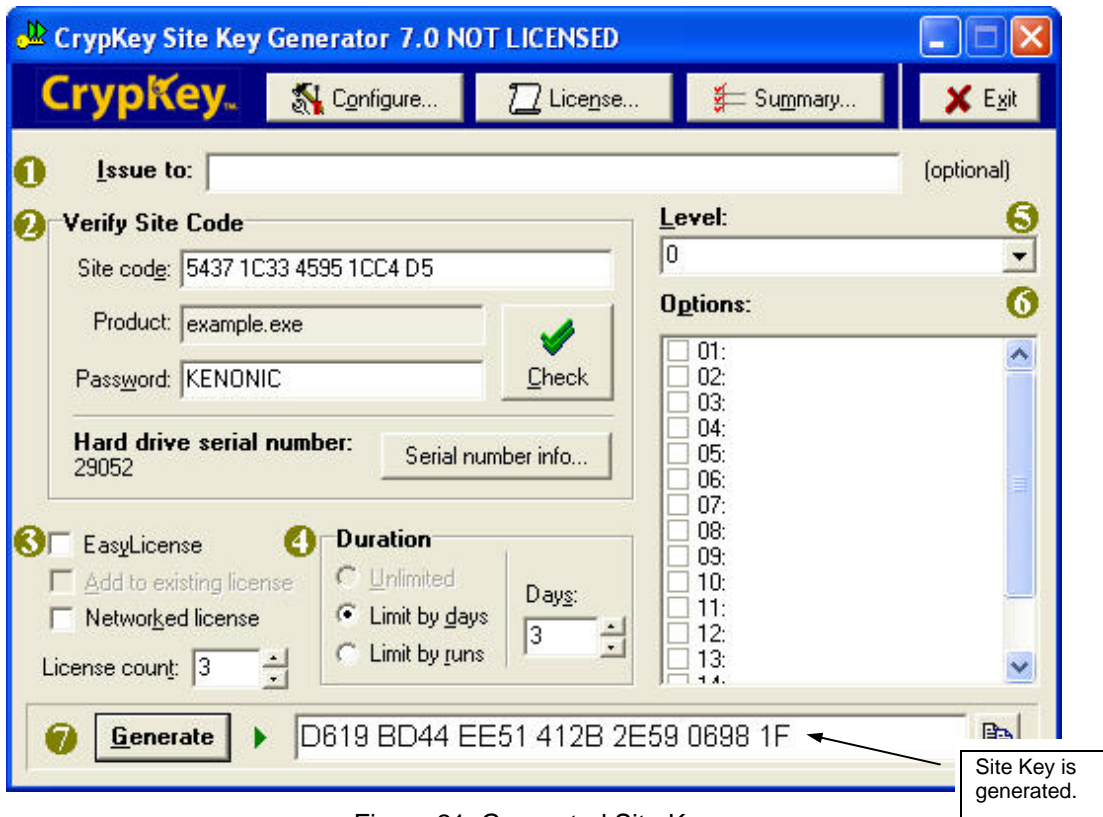


Figure 21 Generated Site Key

12. Select the generated Site Key in the above window and press Ctrl+C to copy the Site Key to the clipboard.
13. Toggling back to the TESTAPP window, place the cursor in the Site Key field and press [Ctrl]+[V] to paste the generated Site Key into the Site Key field.

Note: Again, to place things in context . . . you are playing the role of your customer by receiving the Site Key from the vendor and entering the key into example. Using the CrypKey function buttons in the TESTAPP window, you are now simulating product authorization.

14. Click the **SaveSiteKey()** button. The following return message is added to the preceding messages in the right panel of the TESTAPP window:

SaveSiteKey called ...
SITE KEY ACCEPTED

15. Click the **GetAuthorization()** button. The following return message is added to the preceding messages in the right panel of the TESTAPP window:

```
GetAuthorization() called ...
Oplevel = 0
AUTHORIZATION OK
```

The TESTAPP window appears as follows:

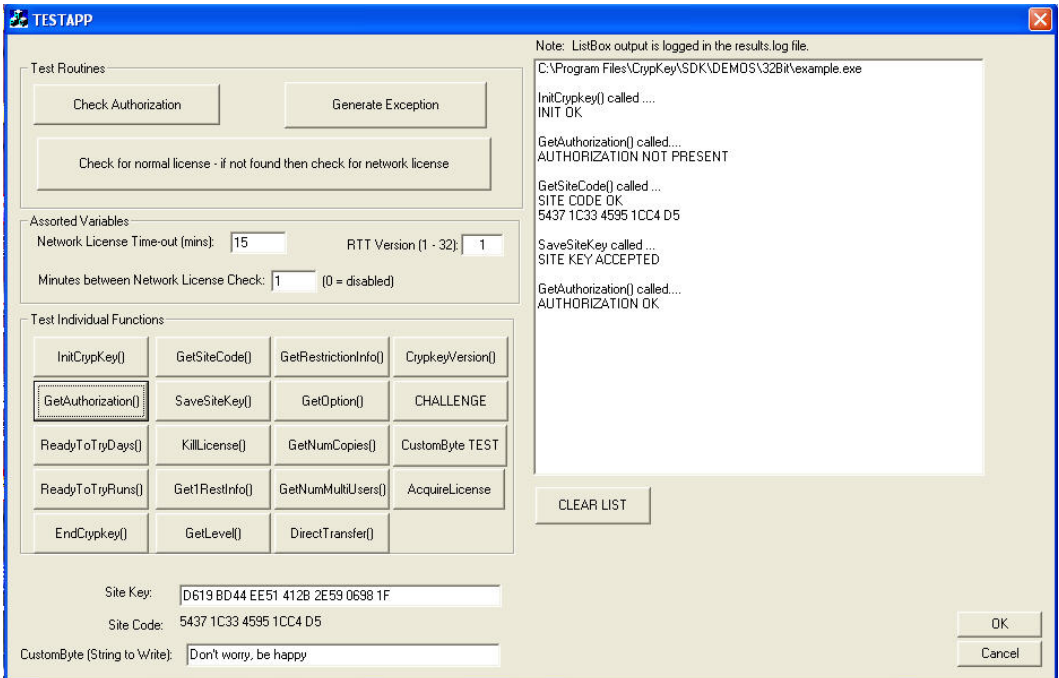


Figure 22 TESTAPP Window with Authorized Site Key

16. You have just granted the demonstration program, *wexample.exe*, the license restrictions you assigned using the Site Key Generator. In this demo, we didn't assign any levels or options. We assigned a three-day license. (Note: Site Key Generator, when in an unauthorized state, can only generate keys for *example.VisualStudio6.MFC.exe* for three runs or three days.) You can send this authorization to a customer located anywhere in the world by email or telephone.

Chapter 6: SKG

Use Site Key Generator to authorize your product for your customer's use.

The CrypKey Site Key Generator (SKG) creates the keys that unlock your product. With SKG, you can limit the number of days or runs that clients can use your product, or implement the unlimited usage option. You can also specify the number of program copies that can be used, and the number of concurrent network users allowed. In addition, you can review previous transactions and sort them by date, customer or product name, and also review archived log files.

Authorizing Site Key Generator

To use the Site Key Generator to authorize your CrypKey registered product, you must first authorize the Site Key Generator license.

Note: You may first use the unauthorized version of SKG for purposes of running the demo test program in *Chapter 5: Demo Programs*.

The Site Key Generator function handles the identification and general authorization of your clients, as well as the definition of your products and associated license terms, options, and levels.

In order to create Site Keys for your CrypKey registered product using your Site Key Generator, you must first provide us with the Site Code from your Site Key Generator (you can determine the Site Code by clicking License in the Site Key Generator dialog box: see below). Your Site Key Generator only needs authorization after you have finished testing with the example developer keys.

Procedure

Once you have finished testing your product with CrypKey SDK 7 (see *Chapter 3: Install*), you are ready to acquire authorization for Site Key Generator.

1. From the Start menu, select Site Key Generator (SKG) from the CrypKey SDK 7 program group:

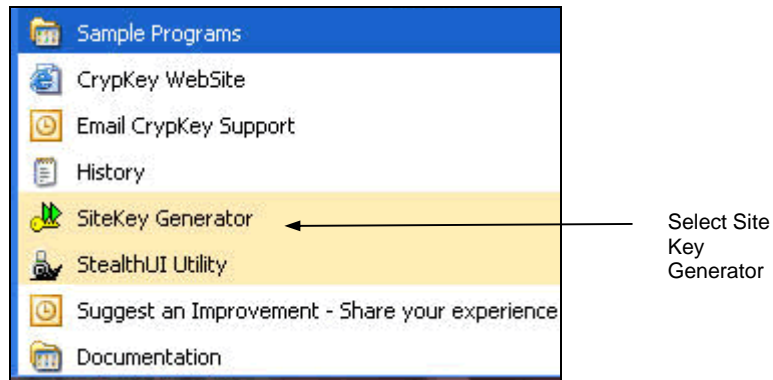


Figure 23 Select Site Key Generator

2. You will see the splash screen:

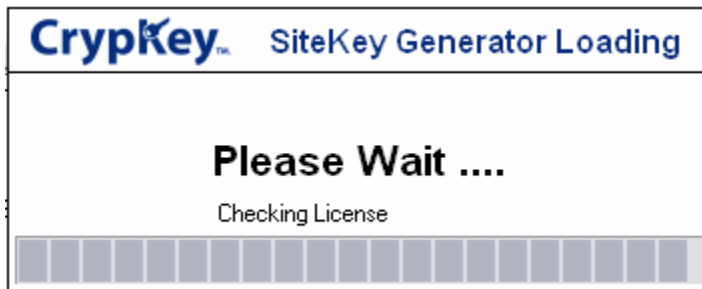


Figure 24 Site Key Generator splash screen

3. Next the SKG main menu appears:

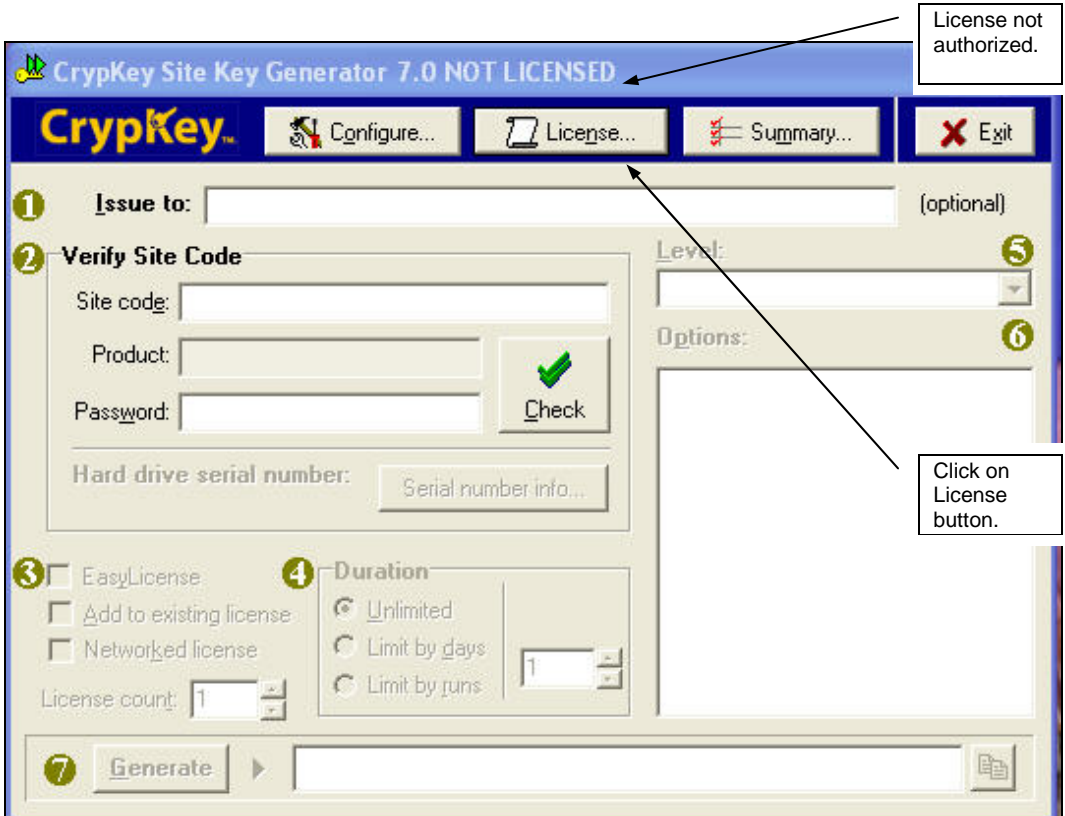


Figure 25 Site Key Generator main menu

4. The status of the Site Key Generator is shown in the title bar (Unauthorized). Click **License** in the Site Key Generator dialog box. Your Site Code appears in the License dialog box:

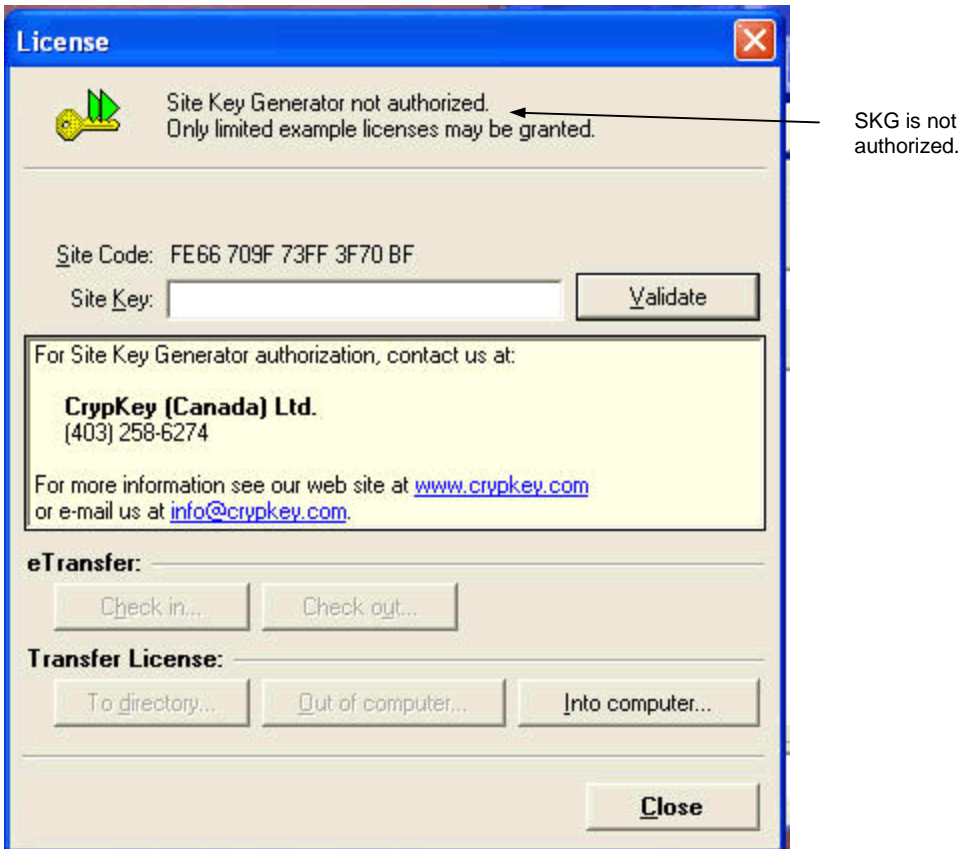


Figure 26 SKG License – Not Authorized

5. Email the Site Code to Authorize@CrypKey.com. Also specify whether you would like a network or standalone license for the Site Key Generator. Please include your name, company name, and CSN number. We will provide you with a Site Key.
6. Enter the Site Key (type it in or copy and paste from the email we send you).
7. Click **Validate** to activate the license:

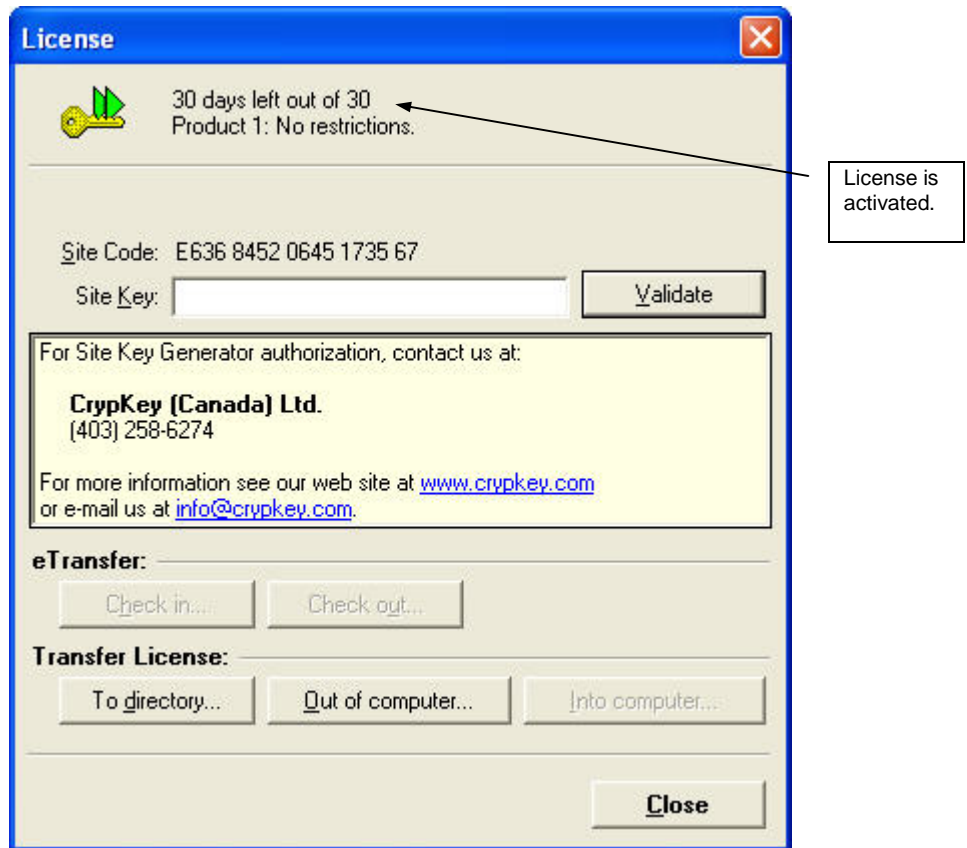


Figure 27 SKG License – Authorized

8. Close the License window. Site Key Generator is now authorized:

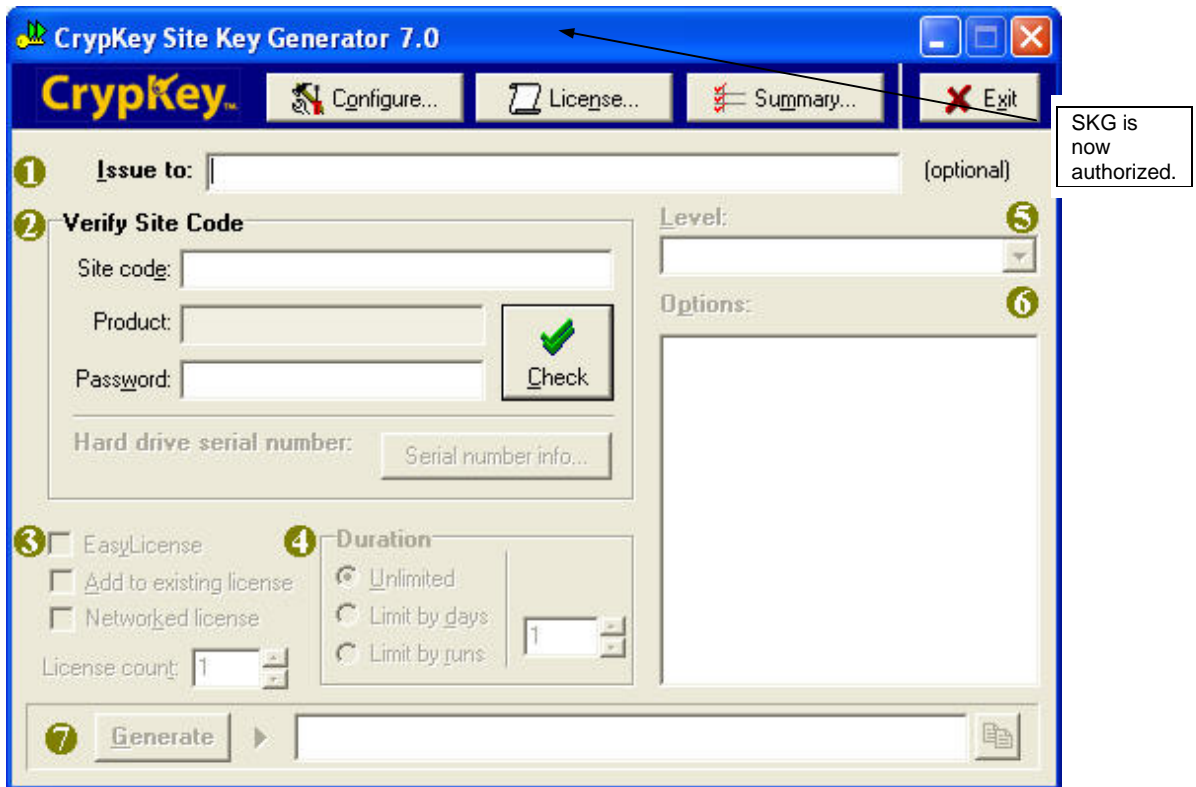


Figure 28 SKG – Authorized

Features of Site Key Generator

Levels and Options

CrypKey SDK 7 levels and options allow you to authorize different features of your product. For example, if your product consists of multiple components and you want your clients to pay per component, toggle on only those components your client wants at the time you issue the Site Key.

The combined total number of bits that can be used for CrypKey levels and options is 32. However, you control what portions of this total are assigned to levels and options. For example, you could implement 28 options (each option uses one bit) for your product, keeping in mind that only four bits would remain for implementing levels.

Within this framework, you set a range of levels and options for each product using the Site Key Generator (see *Configuring Products with Site Key Generator*

on page 67). When your customer requests a license authorization for a product, use the Site Key Generator to:

- select a level (only one of the available levels) and options specifying what access you are providing to the customer for that product;
- issue a Site Key to the customer, which incorporates the selected level and options.

The customer enters the Site Key into your protected application on his or her computer. The levels and options information encoded into the Site Key become a permanent part of the customer's license, which defines the customer's privileges for the application.

FEATURE LEVELS

With levels, you can choose one feature or style you want your clients to have. For example, you can divide your software into beginner, intermediate, and expert levels.

A level is an integer number that controls a variety of features in your product. For example, you may devise the following definitions:

- Level 0: no additional modules can be used
- Level 1: the graphic module can be used
- Level 2: the graphic and printing modules can be used.

Levels can also be used for version control. For example, you may decide that the first version (v1) of your software runs regardless of level, while v1.1 requires Level 1 and v1.3 requires Level 2. Level strategies relating to versions are numerous, but you can choose one level at a time. Configure levels into the Site Key Generator using the **Configure** button.

FEATURE OPTIONS

With options, you can toggle on or off one or more features of your product.

An option is a single bit used like an on/off switch. For example, if you want to offer clients a module that allows your product to perform printouts for an additional fee, you create an option similar to this:

- Option 1 (on): the module can be run
- Option 0 (off): the module cannot be run (a message appears noting that this option has not been purchased)

Use the Site Key Generator's **Configure** button.

Licensing Options

In addition to Levels and Options, the Site Key Generator offers flexible restrictions on licenses, described in this section: EasyLicense for the user's convenience, and CloneBuster technology to prevent hard-drive cloning, or "ghosting".

LICENSE DURATION SETTINGS

You can configure a license using one of three different duration types, all of which can be transferred via direct transfer or floppy disk:

1. **Runs:** allow clients to run your product for a specific number of runs (maximum: 32,768)
2. **Days:** allows clients to run your product for a specific number of days (maximum: 32,768)
3. **Unlimited:** allows clients to run your product without restrictions.

CLOCK MANIPULATION CHECKS

There are programs available on the market that are designed to try to deceive time-restricted software protection systems. These programs automate the process of changing a computer's clock in order to bypass time-restricted protection. These programs change the time registered by the protected program. For example, if the registered time never extends beyond your program's trial period, a user can in theory run your program indefinitely.

CrypKey safeguards against clock manipulations by monitoring your customers' computers for the inconsistencies generated by such an attack. CrypKey detects when time is falsely reported to your program and prevents your program from running. Time reported by CrypKey is that of the server, adjusted to the local time zone of the computer.

CrypKey may also detect clock manipulation if users adjust their computer clock by more than several hours, even for benign purposes such as testing. If your program is prevented from running in these situations, your customer can correct the situation by rebooting the computer.

EASYLICENSE

Some customers need the simplest possible form of copy protection when a particular application runs only on a particular computer. These customers do not want to deal with any associated issues of transfer, network license, trial period, or number of copies. Whether the software is offloaded and put back, deleted and restored later, the application still runs on the computer. EasyLicense meets these requirements.

EasyLicense features include:

- Clients can back up their licenses and restore them later.
- The license files can be moved without affecting the license itself.
- The license transfer ability is disabled to prevent abuse.
- The license types available for EasyLicense are an unlimited license and a days limited license.

EasyLicense works on any system that supports CrypKey's HDSN technology.

EASYLICENSE BACKUP AND RESTORE

Apart from EasyLicense, it is not possible to back up or restore CrypKey licenses. The reason for this is that the other types of CrypKey licenses (those with time or run restrictions) are transferable by the user and the availability of backup-restore would allow the user to duplicate these licenses. However, the EasyLicense is not transferable and, consequently, it can be backed up and restored. You can think of EasyLicense as a permanent authorization of a program for a particular hard drive.

Procedures

It is simple to back up and restore a valid EasyLicense.

To back up EasyLicense:

Copy the following files to a safe place (external media or another computer):

- yourfilename.key
- yourfilename.rst

To restore EasyLicense:

Install the software, if it is not already present, and copy the above two files back to the directory they were stored in. If the software is on the same hard drive as it was when the license was backed up, the license is restored.

CLONEBUSTER TECHNOLOGY

CrypKey SDK 7, with the new patent-pending CloneBuster technology, effectively solves the problem created by the upsurge in software products that clone hard drives. Many popular copy protection technologies that depend on placing some type of simple software footprint are now easily defeated by these products and

are no longer an effective option. Hardware dongles provide limited protection against HD cloning, but are expensive and somewhat unpopular.

Most modern IDE hard drives have burned-in hard drive serial numbers (HDSN). The HDSN is a permanent read-only attribute of a personal computer hard drive, rather than the easily changed volume serial number. The HDSN is very difficult to access, but the CrypKey SDK 7 with CloneBuster technology has this ability. Utilizing CloneBuster, CrypKey detects the HDSN, model number, and firmware identifier and uses this information in its proprietary licensing algorithms. This ensures that your CrypKey-protected applications are not fraudulently transferred to unauthorized computers.

TRANSACTION SUMMARY

The Site Key Generator includes a **Summary** button that you can use to view license activity. When you click this button, the system displays the following window:

Product	Date	Customer	Copies	Restriction	Level
ck_demo.exe	2002-11-28 9:49:55		3	None	6
example.exe	2002-12-2 13:4:47		3	3 Days	0
example.exe	2002-12-2 13:5:1		3	3 Days	0
example.exe	2002-12-6 18:3:42		3	3 Days	0
example.exe	2002-12-6 18:4:1		3	3 Days	0
example.exe	2002-12-7 20:37:18	HYPOTHETICAL CUSTOMER	3	3 Days	2

Figure 29 SKG – License Summary window

Configuring Products with Site Key Generator

Note: This section applies to new products only. To edit existing products, please see *Configuring Site Key Generator for* on page 77.

When you register your product with CrypKey, you must configure the Site Key Generator to recognize it. The following procedure allows Site Key Generator to authorize your protected application. It also gives you the ability to set initial license defaults for the product. (Any defaults that you set can be altered at time of license generation.)

Procedure

1. In the CrypKey Site Key Generator window, click the **Configure** button. The system displays the Configure window:

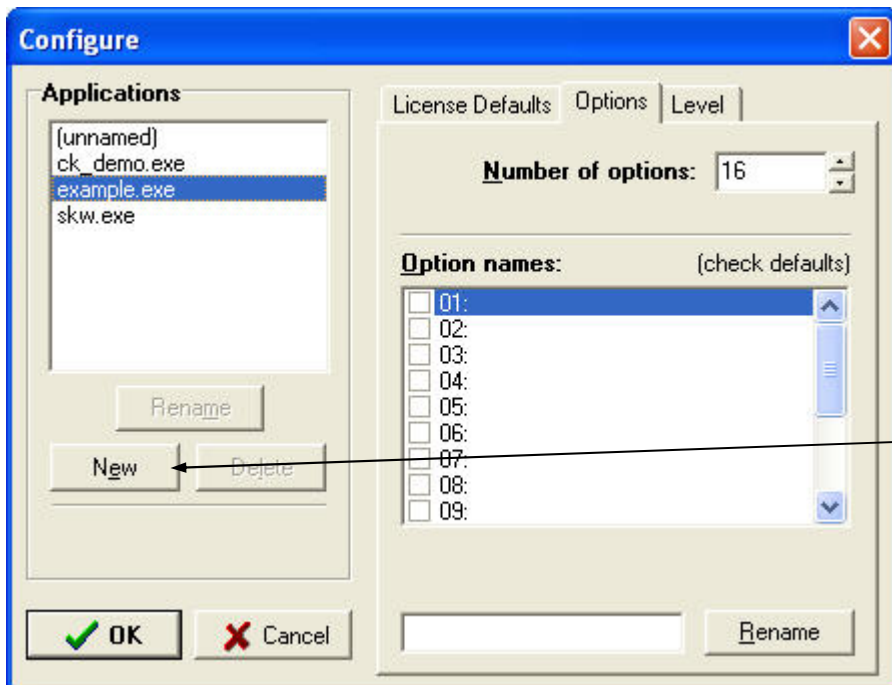


Figure 30 Configure a New Product

2. To configure a new product, click **New** in the above window. The system displays the Configure window with the New Application pop-up superimposed on it:

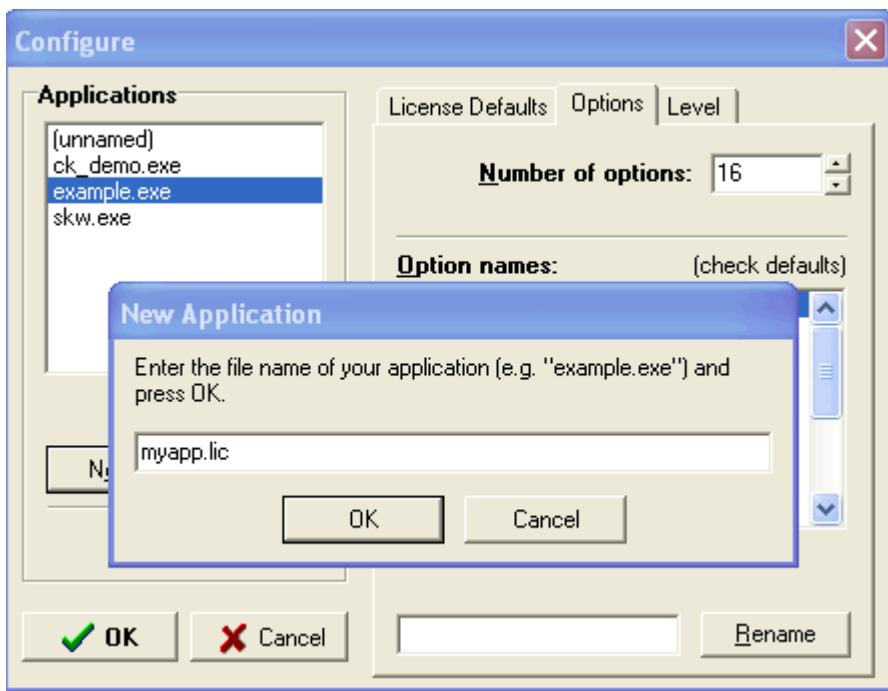


Figure 31 Site Key Generator with New Product Pop-up Window

3. Type the product's file name into the blank field and click **OK**.

Note: Enter the license filename that you registered with CrypKey to obtain your developer keys. For example, *myapp.lic*. You must use the 8.3 format as specified when registering.

4. The pop-up window disappears and the name you have typed is displayed in the Applications list, as shown:

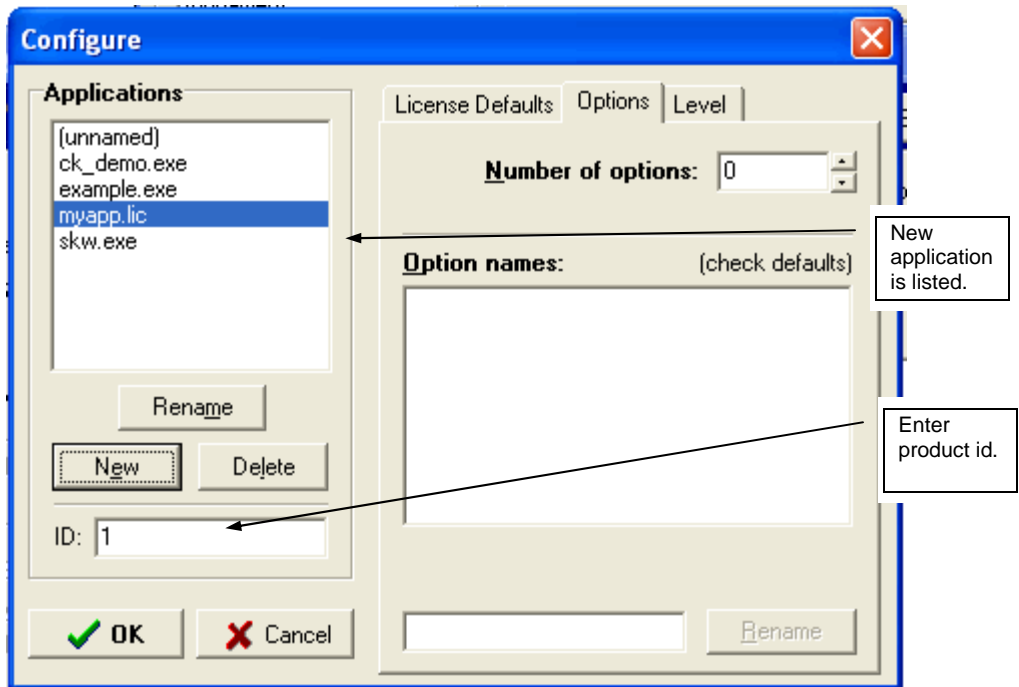


Figure 32 New Product listed in Configure window

5. You can rename or delete the new entry, or any existing entry, by clicking the name in the list, then clicking the **Rename** or **Delete** button.
6. In the ID field, enter the product number for your software — this information is in the developer key document sent to you by CrypKey for the product. For example, enter '1' if you are configuring your first product.
7. Enter the password you supplied to CrypKey to get your developer keys:

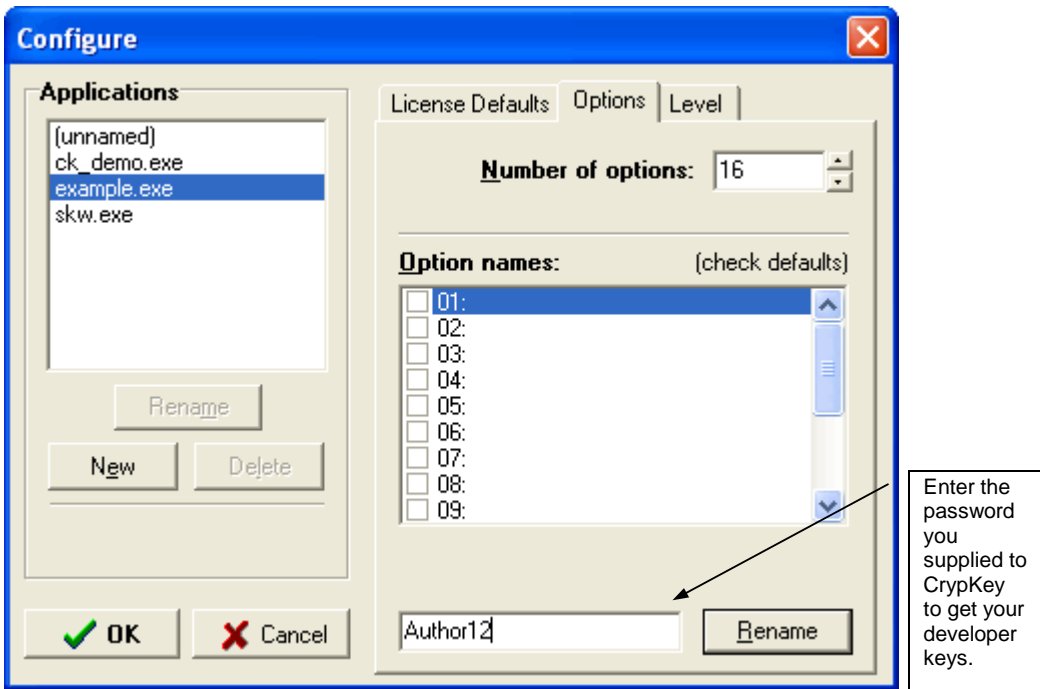


Figure 33 Configure New Product – Enter Password

8. That is all that's required initially: filename, ID, password.
9. The other configurations can be done when you create a Site Key for the product. Or, you can set a list of license defaults for the product. Options and Levels tabs are used if you are using options and levels within your application.
10. The checkbox choices are as follows. These are defaults for the specified product, but most options can be changed as needed for specific customers.
 - EasyLicense — used for the simplest form of copy protection.
 - Add to existing license — used if a customer with an existing license requests authorization. This option is only available if the customer requesting authorization currently has a valid license. If they do not have a valid license the option is not available. You can only see if the option is available after entering the end user's Site Code and clicking **Check**. For example, you would use this option to add days to an existing days limited license; therefore, if a customer with 10 days left on their license called in to get an extension of 5 days, and you added 5 days to the existing license, the new license would be valid for 15 days. This method insures that your customer does not mistakenly get additional days on their license extension.

- Networked license — used if the application is to be run by one or more users connected to a server.
- License count — maximum number of fixed or floating licenses that a customer can have for the specified product.
- Duration — amount of usage allowed under the license. This can be unlimited (the default) or can be expressed in number of days or runs. If you select days or runs, the number box to the right becomes active and you are able to enter the number of days or runs allowed for the license.
- Password — enter the password you specified when requesting your developer keys. You must enter that password into this field in order to be able to successfully authorize end users. (Note this *cannot* be changed per individual customer.)

11. Select the **Options** tab:

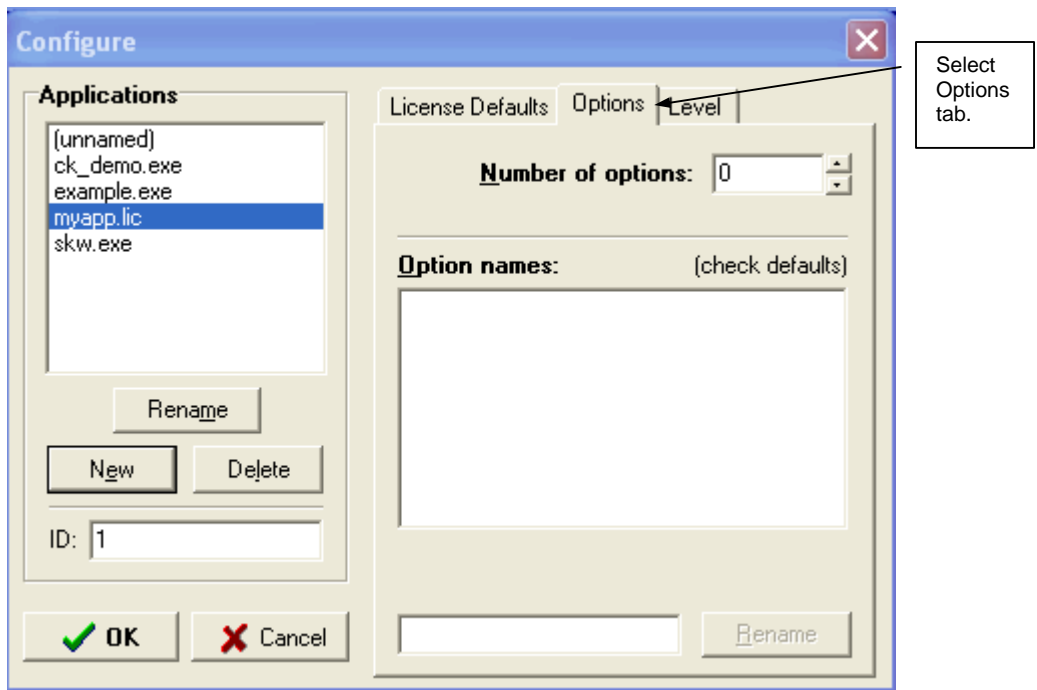


Figure 34 Site Key Generator – Options tab

12. In the **Options** window, click the combo box arrows to increase or decrease the number of options to be assigned to the product. As you increase the number, blank option entries appear in the Option names sub-window, as shown:

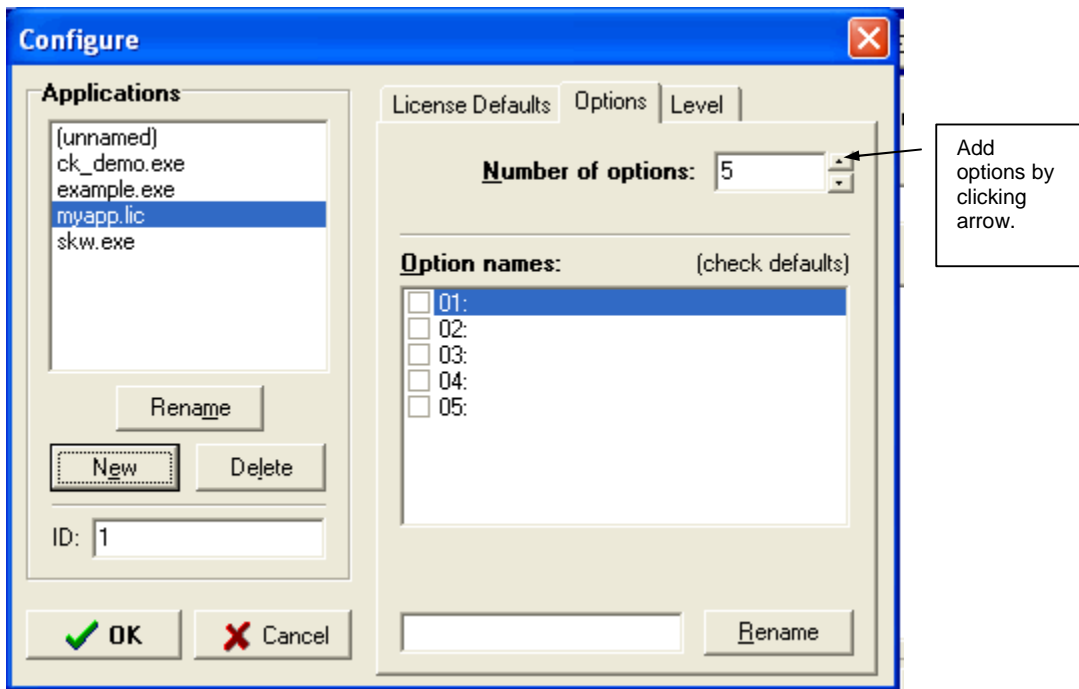


Figure 35 Site Key Generator – Add Options

13. To assign a name to an option number, check the option's checkbox, type the name in the text box at the bottom of the window, and click the **Rename** button:

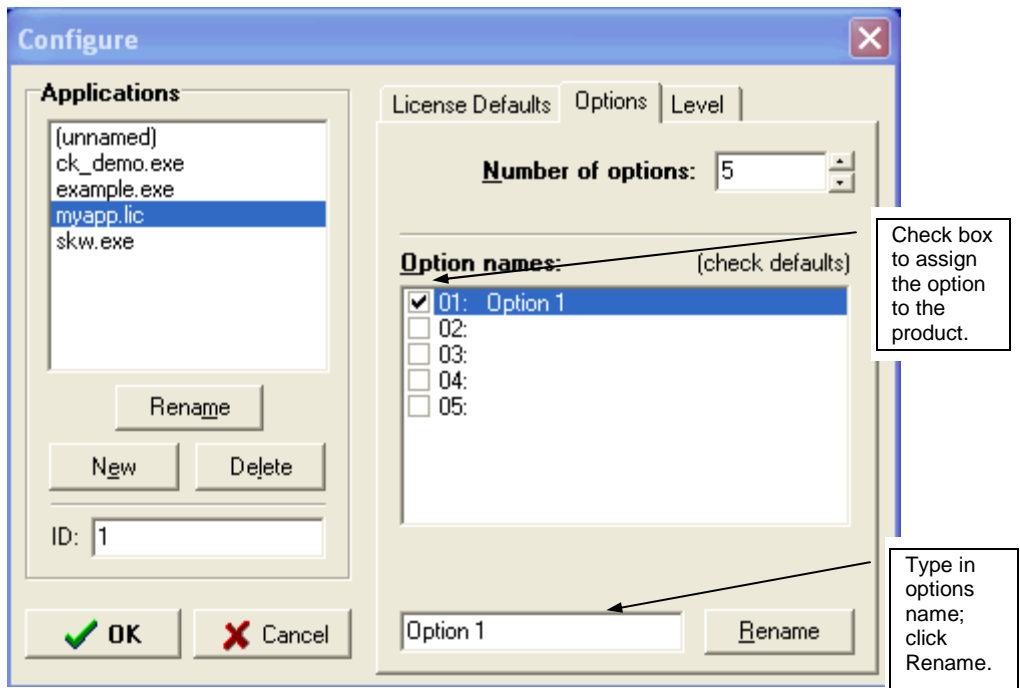


Figure 36 Site Key Generator – Rename Option

14. Click the checkboxes beside the options that you want as the defaults for the license.
15. Now select the **Level** tab:

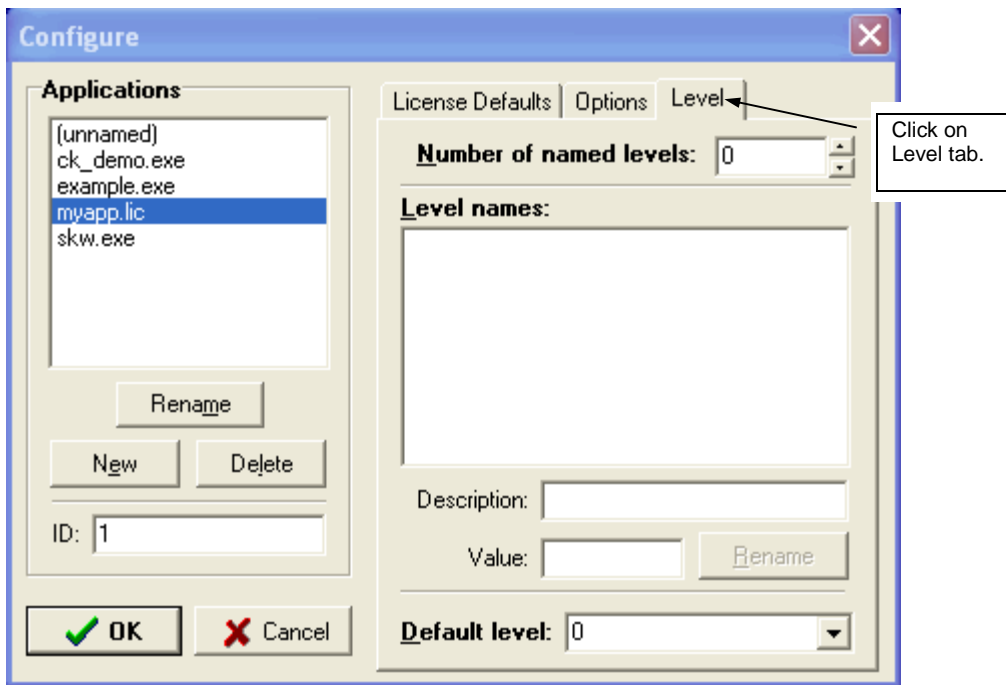


Figure 37 Site Key Generator – Level tab

16. In the above window, click the combo box arrows to increase or decrease the number of levels which will be the default for a license. As you increase the number, blank option entries appear in the Level names sub-window, as shown:

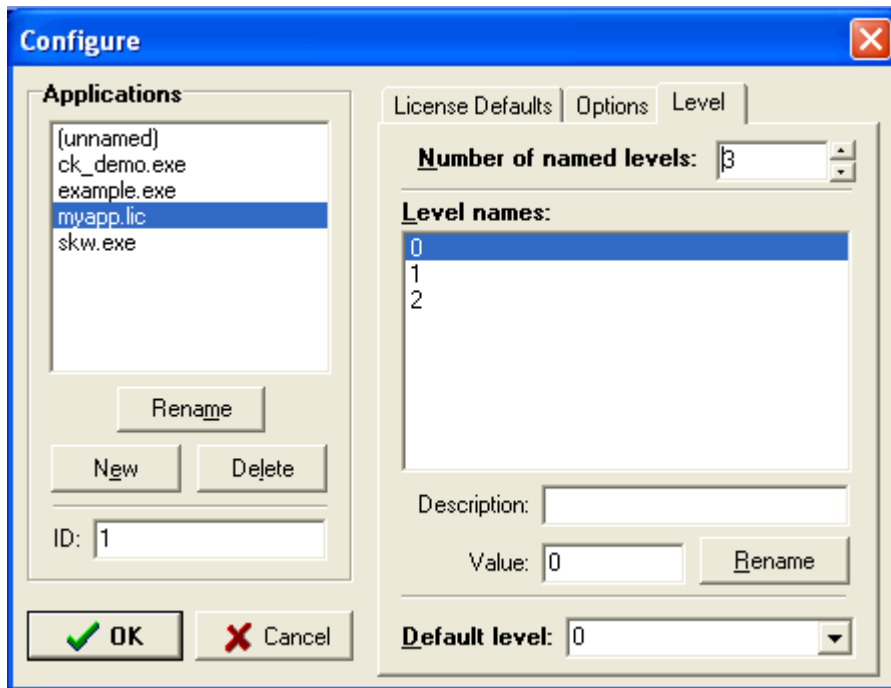


Figure 38 Site Key Generator – Assign Levels to a Product

17. To assign a name to a level, click on an entry number, type a name in the Description text box, and click the **Rename** button.

Note: Only one level can be assigned to an authorization.

18. You can verify the license terms and make any changes if needed. In the SKG main window, click on **Summary**. The License Summary for the sample program example.exe is displayed:

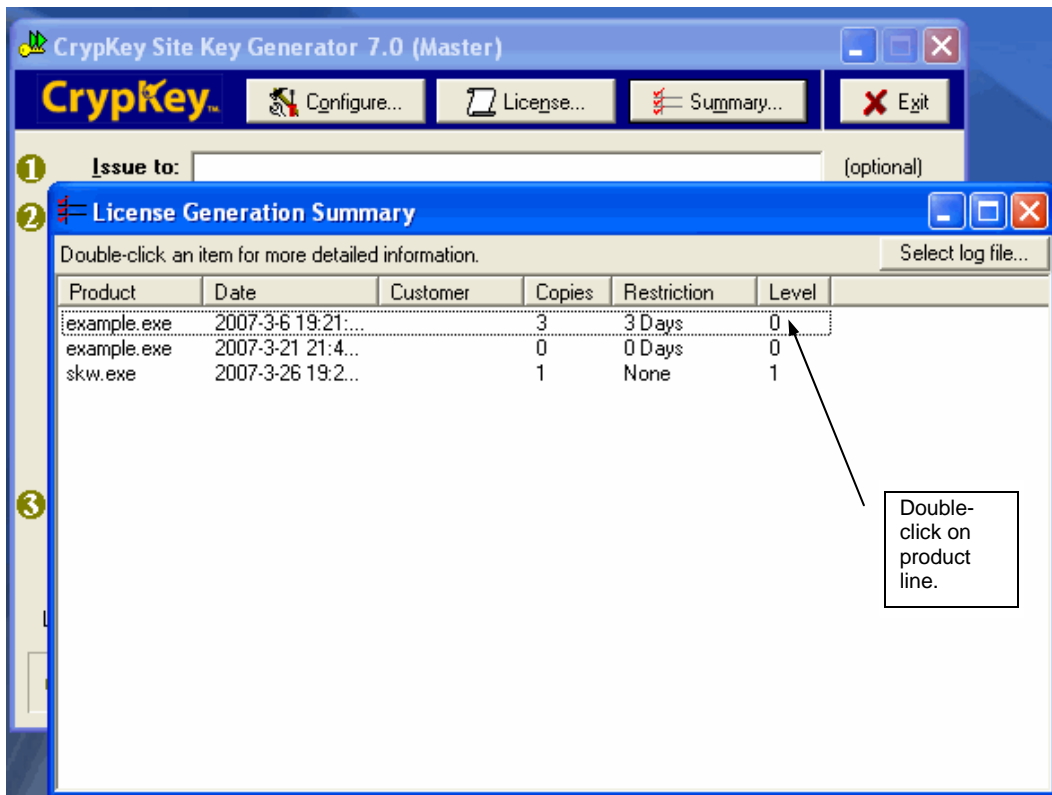


Figure 39 Site Key Generator – License Summary window

19. Double-click any product line to display the License Details window for it, as shown below:

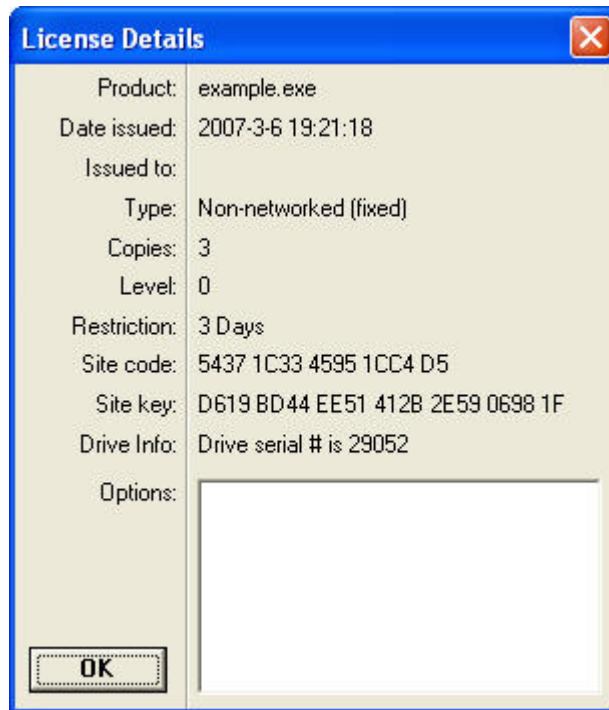


Figure 40 Product License Details

Configuring Site Key Generator for Network Use

The Site Key Generator (*skw.exe*) is designed to operate as a standalone application (Normal Mode operation), but can be run in drive share or client/server mode as well.

Procedure

To configure your client-server Site Key Generator for use on a network, you need to place the contents of the C:\CrypKey.7031\SiteKey Generator\SKW_LICENSE on the server. Place the C:\CrypKey.7031\SiteKey Generator\SKW_APP files a directory on the client machine. To do so, follow the steps below:

1. From the server machine, run the ELM.
2. Go to **Actions**.
3. Select a license and choose *skw.ced* in the license server directory.
4. Select **start server**, then go to **edit, edit configuration**, and choose who you would like to have permission (we recommend starting with primary and all to make sure everything works).

5. Go to manage server licenses.
6. Select your server and get it authorized for three (3) network users.
7. Send this request to Authorize@CrypKey.com with your customer service number and company name.
8. Place the Site Key in the authorization screen.
9. Then go to the *skw.exe* on the client machine and launch the *skw.exe*. This action should find the license on the server.
10. While the client has the *skw.exe* running check the view, network clients on the ELM. You will see who is attached.

Editing Products with SKG

This section applies to exiting products and customers.

Procedure

1. From the Site Key Generator main window, click **License**. The Configure window is displayed:
2. Select name of an existing product in the Applications list.

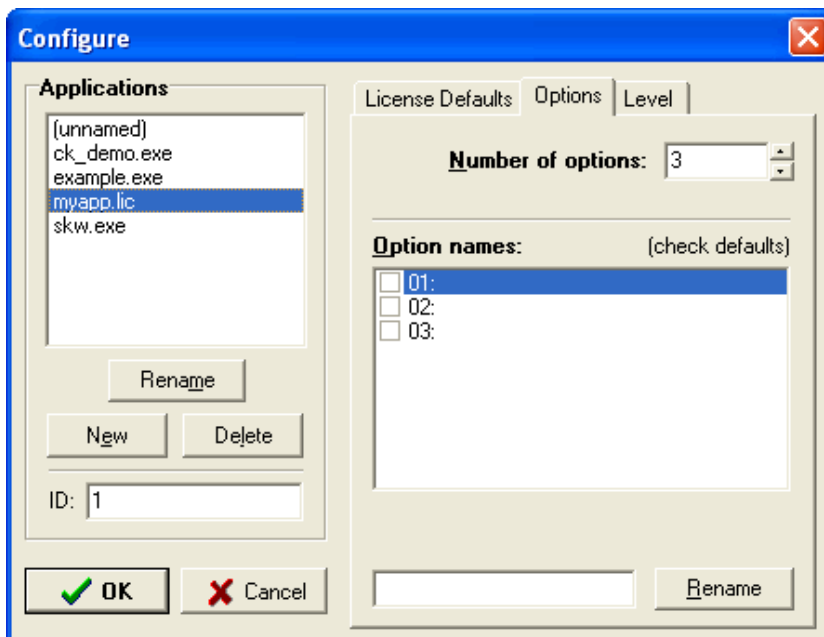


Figure 41 Configure Existing Product

3. To delete the product, click the **Delete** button.
4. To rename the product, click the **Rename** button. A dialog box pops-up:

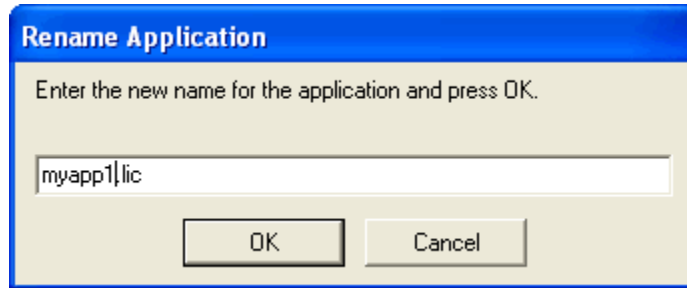


Figure 42 Rename Product


5. To make other changes, follow the applicable steps as described in *Configuring Products with Site Key Generator* starting on page 67.

Changing License Usage for your Customers

The Site Key Generator allows you to change the levels, options, and restrictions for a customer — when you have a valid Site Code from the customer's authorized copy of CrypKey-protected software. This option is not available if the Site Code supplied by your customer is from a Ready To Try period. If you need to extend a trial period, simply input the Site Code the end user has provided and enter the total number of days you would like them to continue using the license.

You can only change an existing license if you choose **add to existing license**. Otherwise you are simply giving the client a new authorization. For example, if you wanted to add 10 days to a 30 days authorization, you must select **add to existing license**, which would then give you 40 days. If you just choose 10 days without choosing add to existing you will have a 10 days license.

Procedure

1. Click the Site Key Generator icon  in the CrypKey program group to display the Site Key Generator main window.
2. Enter a valid Site Code into the Site Code field.
3. Click the **Add to Existing License** checkbox in the Site Key Generator dialog box, as shown:

Note: If the option is not available (greyed out), this means the entered Site Code is from a Ready To Try period or from an unauthorized piece of software.

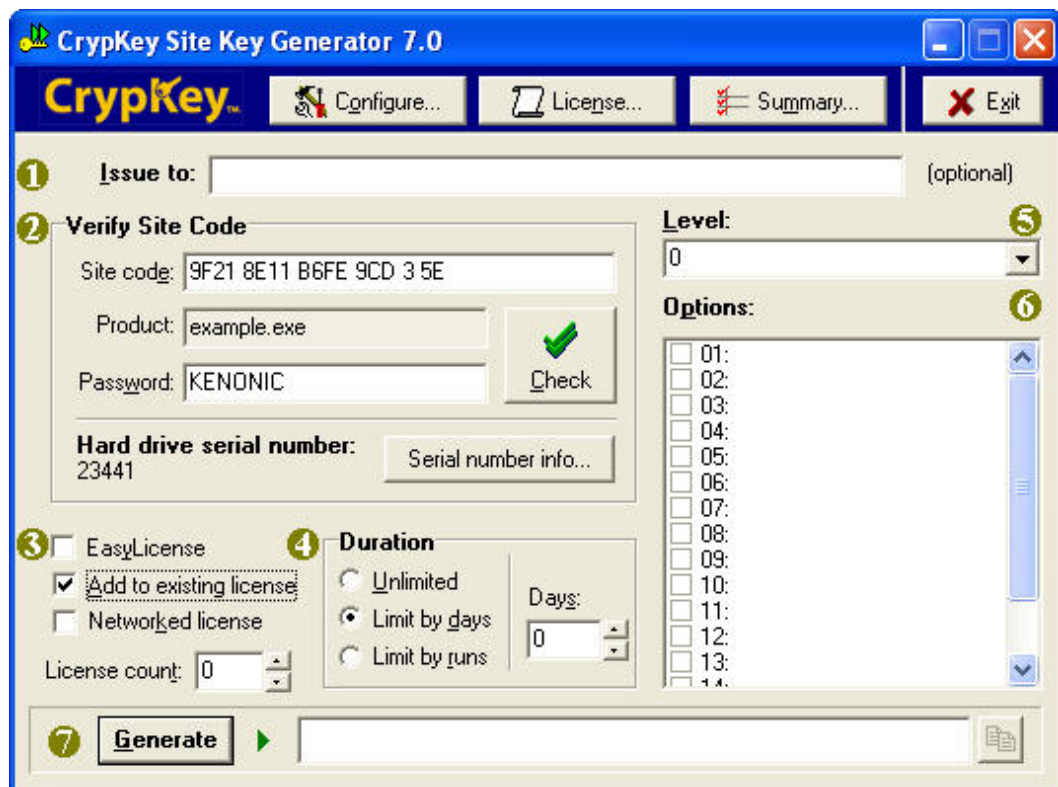


Figure 43 Add to Existing License

4. Specify levels, options, and duration of the license that you would like to add to the existing license. These changes are added to the license for the customer who provided the Site Code.
5. Click the **Generate** button to produce a new Site Key for the customer:

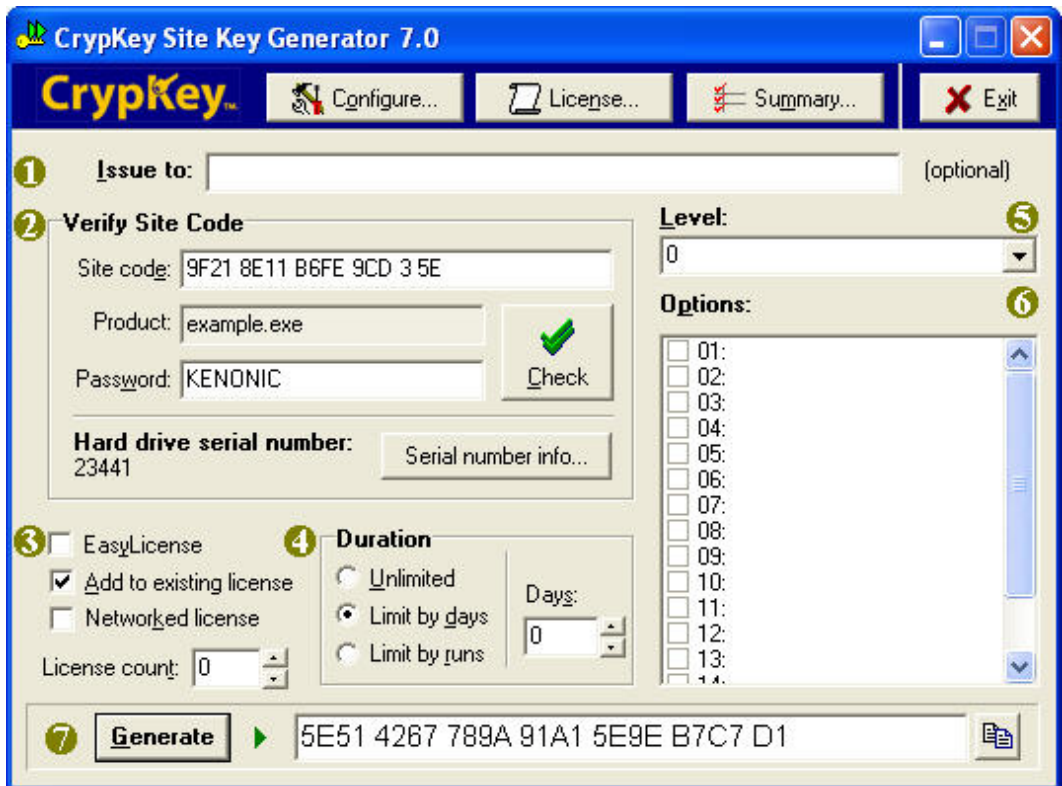


Figure 44 Generate New Site Key for Existing License

Generating Site Keys

The CrypKey Site Key Generator window lets you authorize customer use of your product. When a customer loads your CrypKey-protected application on his or her computer, the application generates a Site Code. The customer sends you the Site Code, which you enter into the Site Key Generator to produce a Site Key. You send this Site Key to the customer, who uses it to unlock your application under the terms you have defined in the Site Key Generator.

Procedure

1. Receive the customer's Site Code, generated from your application, which the customer has installed on their machine.
2. Start Site Key Generator (🌿) from the CrypKey menu. The system displays the CrypKey Site Key Generator window.

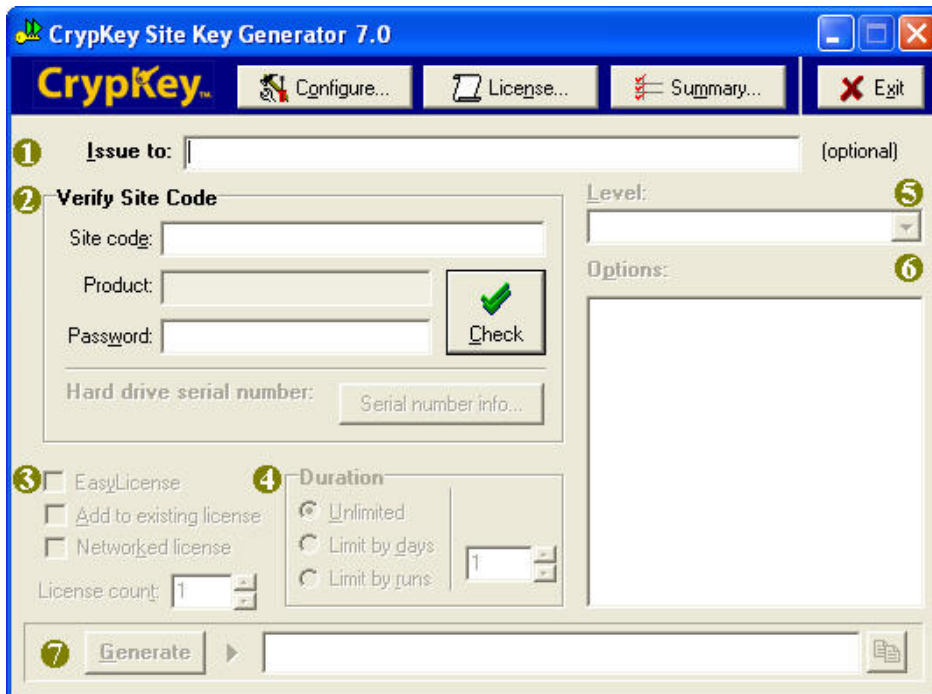


Figure 45 CrypKey Site Key Generator window

3. **Issue to:** Optionally, type in the customer name. CrypKey does not use this information, however it does get stored in the summary for your reference.

4. **Site Code** field: type in the Site Code provided by the customer.
5. Click the **Check** button. If the Site Key Generator is configured correctly, the filename and password will automatically appear in the appropriate fields:

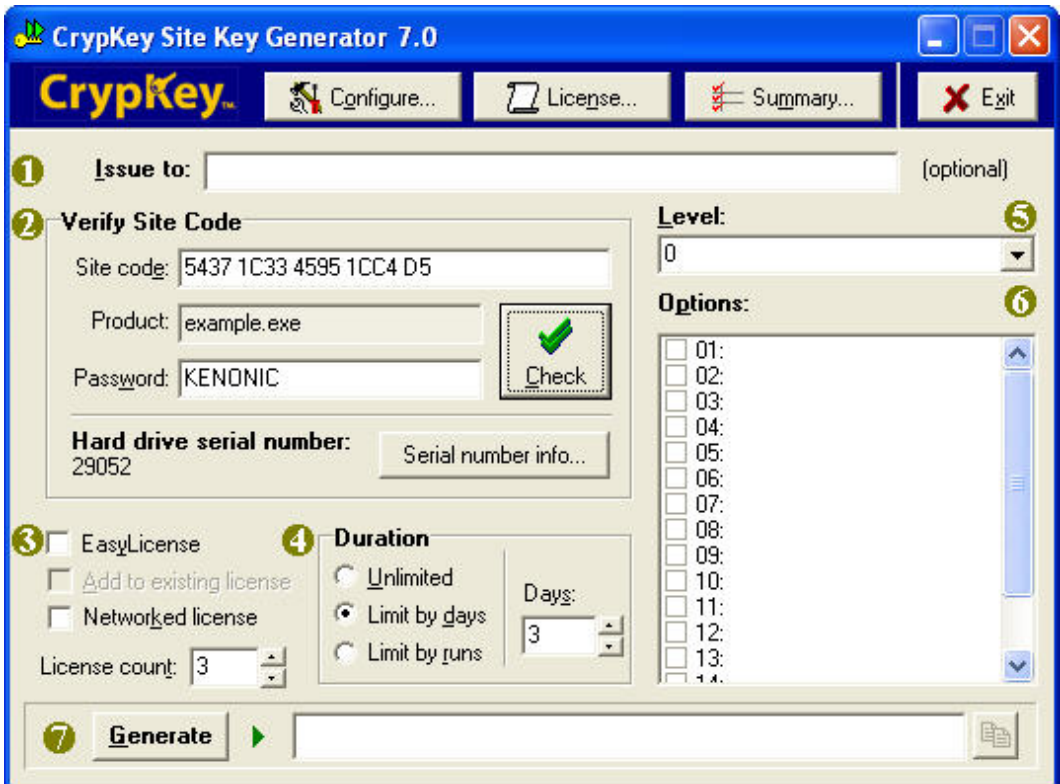


Figure 46 Site Key Generator – Check Customer Site Code

6. SKG keeps a record of the authorization information. It is a useful, for example, as verification if your customer asks you for re-authorization following a computer crash.
7. Choose your license restrictions.
 8. In the **Level** box, click the drop-down arrow to assign one of the levels that were pre-defined during the product configuration (if applicable).
 9. In the **Options** box, click the checkboxes beside the product options that you wish to activate for the customer (if applicable).
 10. Click the **Generate** button. The system generates a 26-character alphanumeric value and places it in the box beside the **Generate** button. This is the Site Key that you send to your customer. It contains all of the specifications you have entered concerning license terms, level, and options.

When your customer receives the Site Key and enters it into the Site Key field provided for the application, the customer is authorized to use that application under the license terms defined.

Distributor Authorizing License

This is an add-on option and must be purchased separately. The Distributor Authorizing License (DAL) allows you to authorize other Site Key Generator holders to authorize your product. CrypKey provides you with an authorization for a Master Site Key Generator that allows you to generate Site Keys for Slave Site Key Generators.

Note: The DAL is an add-on product to CrypKey SDK 7 or CrypKey Instant. Please contact Sales@CrypKey.com for more information on the DAL..

Chapter 7: Programming

The standard process for implementing CrypKey involves programming the SLAPI library functions into the source code of your application and providing the interface to your clients.

The compile and link procedure is the final stage of the process. There are several pieces of sample code available in your CrypKey SDK 7 Sample Code directory.

Basic Programming Steps

The following steps show a suggested algorithm for establishing the license at start-up. The steps are provided mainly to illustrate what the primary functions do. The steps you actually take may be very similar to the steps we describe, but your steps can be tailored for your application.

CrypKey SDK 7 operates very much the same as the previous version, with the exception that the *CRP32002.ngn* file is required in your application directory. If you don't include this file, errors in the -200 range will likely result.

For up-to-date error referencing, please refer to *crypkey.h* in the Include directory.

Establishing the License

Procedure

1. Call the `InitCrypkey()` function.

This function initializes the CrypKey system with the path of the file to check for authorization. Call it on opening the application or starting the program, to initialize CrypKey.

This function returns 0 if successful (CrypKey initialized) and returns a negative value if unsuccessful (CrypKey not initialized). If the `InitCrypkey()` function returns a negative value, display an error message and disable or limit the program as desired. You may find it helpful to use the `ExplainErr()` function with the failure code to retrieve the appropriate error message.

Note: The `InitCrypkey()` and `GetAuthorization()` functions must be called at least once before any other `CrypKey` functions are called. `InitCrypkey()` is the first function call used.

For details on the `InitCrypkey()` function, see the description in *Chapter 14: Function Reference*, `InitCrypkey()`.

2. Call the `GetAuthorization()` function.

The `GetAuthorization()` function checks whether or not the program is authorized to run and returns the level/options information if the program is authorized. The function is also used to decrement the count of runs used.

Call this function on opening the application or starting the program, to check whether or not the user is authorized to run the program, and call it whenever you need to check the level or option setting for the user's license. Call this function periodically to strengthen your security.

This function returns 0 if successful (program authorized), a negative value if unsuccessful (program not authorized), or a positive value (signifying that the program has a multi-user license and more users are requesting use of the program than the license allows).

If the `GetAuthorization()` function returns 0, proceed to step 3 only if the user wants to change the license permissions.

If the `GetAuthorization()` function returns a negative value, display an error message informing the user that the program is not authorized, optionally with an explanation (you may find it helpful to use the `ExplainErr()` function with the failure code to retrieve the appropriate error message). Continue as outlined in step 3 to prompt the user to authorize the program, possibly at a later time (i.e., from a menu item command).

If the `GetAuthorization()` function returns a positive value, which is a queue value indicating the number of users who must exit the application before a license becomes available, continue repeating calls to `GetAuthorization()` until the user either gets a license or tires of waiting and quits.

Note: The `InitCrypkey()` and `GetAuthorization()` functions must be called at least once before any other `CrypKey` functions are called, and `GetAuthorization()` must also be called repeatedly at regular intervals when network licensing is in effect.

See the `GetAuthorization()` function description in *Chapter 14: Function Reference*, `GetAuthorization()` for more details.

3. Call the `GetSiteCode()` function.

This function gets the Site Code for the program. Call it when you want to authorize a new license or change the current license permissions. It returns 0 if successful, meaning that the Site Code is returned, or returns -1 if CrypKey is not yet initialized.

You must display the Site Code to the user in order for the user to be able to provide it to you when he or she requests a Site Key. See *Chapter 6: SKG* for instructions on creating the Site Key. The user can shut down the program while waiting for the Site Key; the Site Code is not changed when the program starts up again.

See the `GetSiteCode()` function description in *Chapter 14: Function Reference*, `GetSiteCode()` for more details.

4. Call the `SaveSiteKey()` function.

The `SaveSiteKey()` function saves the Site Key information, which contains your product's licensing permissions for the customer's site. Encrypted license files are created from this function and saved on the computer's hard disk. Call the function after the user has entered the Site Key.

If the `SaveSiteKey()` function returns 0, a new license is created for the customer's site. If the `SaveSiteKey()` function returns a negative value, notify the user that the Site Key is invalid. You may find it helpful to use the `ExplainErr()` function with the failure code to retrieve the appropriate error message.

Note: If the Site Key is saved successfully, the Site Code changes. This is done to prevent the user from reusing an old Site Key. If you give the user a Site Key that allows 10 runs, you don't want the user to reuse that old key later for another 10 runs—even if recycling is good for the environment!

See the `SaveSiteKey()` function description in *Chapter 14: Function Reference*, `SaveSiteKey()` for more details.

5. Call the `EndCrypkey()` function.

The `EndCrypkey()` function ensures that the licensing control system terminates cleanly without errors. Call this function just before exiting the program. There is no return value.

Note: If `EndCrypkey()` is not called before the program terminates, various errors may be encountered the next time the program is started and the `InitCrypkey()` function is called.

See the `EndCrypkey()` function description in *Chapter 14: Function Reference*, `EndCrypkey()` for more details.

GUI Programming Tasks

In addition to checking to see if the program is authorized when your application is started, you must provide menu options that enable the user to authorize your product and perform authorization transfers.

Authorizing the Program

Procedure

To authorize the program:

1. Call `InitCrypKey()` – if a 0 is returned initialization is successful.
2. Call `GetAuthorization()` – if a 0 is returned authorization was successful. If a negative value is returned the program is not authorized.
3. Execute the `GetSiteCode()` function.
4. Check the return value and code the appropriate result (i.e., disable the program).
5. Open a window passing the Site Code obtained from the `GetSiteCode()` function to the user.
6. Allow the user to enter a Site Key in response to the Site Code.
7. Execute the `SaveSiteKey()` function passing the user provided Site Key.
8. Check the return value and code the appropriate result; Disable the program if it fails, check authorization information and enable your application's appropriate features if it succeeds.

Registering the Authorization Transfer on the Target Computer

Procedure

1. Open a window on the target machine requesting the directory path to which the license will be transferred.
2. The user enters or selects the default path in the window and presses an **OK** button.
3. Execute the `RegisterTransfer()` function.
4. Check the return value and code the appropriate result.

Transferring the Authorization Out of the Source Computer

Procedure

1. Open a window requesting the directory path.
2. The user enters or selects the default path in the window and presses an **OK** button.
3. Execute the `TransferIn()` function.
4. Check the return value and code the appropriate result.

Transferring the Authorization into the Target Computer

Procedure

1. Open a window requesting the directory path.
2. The user enters or selects the default path in the window and presses an **OK** button.
3. Execute the `TransferOut()` function.

Compiling and Linking

The following sections explain how to compile and link `CrypKey` into your C or C++ application.

CRYPKEY HEADER FILE

Procedure

Include `crpkey.h` in all modules that make calls to the `CrypKey` library. If you are using the `CRP32DLL.DLL` with a C or C++ module, you must include the following statement:

```
#define DLL
```

LINKING YOUR TO APPLICATION

Link the appropriate library to your application, depending on the compiler and platform. All libraries have been compiled using the Large memory model. `CrypKey` requires ample stack space — you should allow at least 10K.

The table below describes the available libraries. See the notes following the table.

Table 7: Libraries

Library	Compiler	Platform	Type
CRYPTKEYCOM.DLL	All compilers that can use COM objects	Win32	COM Object
CrypKeyNET.DLL	All compilers that can use .NET objects	.NET Framework	.NET assembly
CRP32D42.LIB	Microsoft VC++ up to 5.x	Win32	Multi-thread dynamic RT library
CRP32D60.LIB	Microsoft VC++ 6.0	Win32	Multi-thread dynamic RT library
CRP32D70.LIB	Microsoft VC++ 7.0	Win32	Multi-thread dynamic RT library
CRP32D80.LIB	Microsoft VC++ 8.0	Win32	Multi-thread dynamic RT library
CRP32M70.LIB	Microsoft VC++ 7.0	Win32	Multi-thread static RT library
CRP32M80.LIB	Microsoft VC++ 8.0	Win32	Multi-thread static RT library
CRP32M60.LIB	Microsoft VC++ 6.0	Win32	Multi-thread static RT library
CRP32S60.LIB	Microsoft VC++ 6.0	Win32	Single-thread static RT library
CRP32S70.LIB	Microsoft VC++ 7.0	Win32	Single-thread static RT library
CRP32DLL.DLL	Any Win32-based compiler	Win32	Multi-thread RT library
CRP32DLL.LIB	Microsoft VC++ 4.2	Win32	Import library for CRP32DLL.DLL
CRP32BMT.LIB	Borland C++ 4.51	Win32	Multi-thread static library
CRP32002.NGN	Any 32-bit	Win32	Same directory as

Library	Compiler	Platform	Type
	program on Windows NT, 2000 or XP, 2003 and Vista.		32-bit application or CK license
CRP64M80.LIB	Microsoft VS8 x64	Win64	multiple thread static RT libraries
CRP64D80.LIB	Microsoft VS8 x64	Win64	multiple thread dynamic RT libraries
CRP64DLL.DLL	Unified Windows x64 DLL	Win64	multiple thread dynamic RT libraries
CRP64DLL.LIB	Unified Windows x64 DLL	Win64	Used for static binding of the CRP64DLL.DLL

Note: If you are using Microsoft VC++ in a Win32 application that dynamically links to Windows C runtime library, you must use CRP32D42.LIB or CRP32D60.LIB.

CrypKey can also be used within non-C applications by applying the dynamic link library (DLL).

Library Notes

If you are getting unresolved externals or multiple defined externals, it's probable that you are linking to the wrong library.

Microsoft Visual C (MSVC) allows:

- single or multiple threads
- static or dynamically linked runtime libraries

There is a different C runtime library for each combination of these pairs and the compiler automatically links to the correct one for you.

You must also make a different CrypKey library for each combination and tell the compiler the correct one to link to.

Most MSVC sample projects default to multithread and dynamic runtime C Lib (CRT), which is a combination we don't support. We recommend the combination of multithread with static libraries, as fewer problems occur with this model.

CrypKey uses three of the possible four model combinations. The following table shows the library names and the combinations for which they are used:

Table 8: Library Linkages

Name	Type	Linkage
CRP32D70.LIB	Multi thread	Dynamic CRT
CRP32M70.LIB	Multi thread	Static
CRP32S70.LIB	Single thread	Static

To use development tools such as Visual Basic, Access, Delphi, and PowerBuilder with CrypKey:

1. Ensure that your application is able to make calls to dynamic link libraries. Refer to development system's manual for instructions on how to call and link DLL functions.
2. Declare/define your external CrypKey functions within your application.
3. Since you are unable to use the crypkey.h library file, which contains error handling variable declarations, you may use the integer value provided within your programming or declare your own variables within your application.
4. Ensure that you include crp32dll.dll. You must include this file with your installation package.
5. Call the **CKChallenge32()** function to ensure that the file CRP32DLL.DLL has not been replaced with an impostor. This function requires two additional secret numbers that are provided with your developer keys. You can use the numbers in the sample applications for testing purposes.

The secret numbers are the password number and the company number you receive with your developer keys. Following is an example set of these numbers.

The Master Key for EXAMPLE.EXE, company number 7956123, is:

```
6a6167abf35cb0253b91761971cfcecf79b6935d6e77905f5b29a45badd3a4f213
15cee18798f96ef1861455c0f18643746f1183580caa6a3f5ef6d8e3dd7e15da0f3a
b21414b8fbe549d385552f816406b15040070d636ab2153d715e71c6c2eeda08f5
8503f64ce9183759075949ff249013d045b55907aa6924c5d707546
```

The User Key is:

D050 815C D1A2 A79D B1

The password number is:

984534120

COM Object

The CrypKey COM Objects are tools that have proven useful in configuring protection for applications and issuing licenses. The essential information you need on this topic is provided here. Additional detail is available in the document named *CrypKeySDK COM Object.doc*, which is found in your CrypKey directory CrypKeySDK COM Object.

Using Dynamic Link Libraries (DLLs) with Visual Basic and other languages has always been a difficult task. Since LIBs are encapsulated in a COM object, the incorporation of CrypKey SDK 7 functions into an application becomes much easier. This is particularly true with Visual Basic (VB), where COM support is built in. COM objects are also supported by C++, Java, and web development tools like IIS and Cold Fusion.

Note: .NET assemblies are similar to COM objects and are used in essentially the same way. However, .NET assemblies are available only to .NET languages (VB.NET, C#.NET, C++.NET).

For each CrypKey function, you will find the prototypes used in C, C++, VB and C#.NET in Sec. 12.3: Function Descriptions.

- The C prototype is for use with the CrypKey Library.
- The C++ and VB prototypes are for use with the CrypKey COM Object.
- The C#.NET prototype is for use with the CrypKey .NET assembly.

COM OBJECT IDENTIFIERS

The COM objects in the library have specific names and types, as shown:

Table 9: COM Object Identifiers

Object	Name	C++ Type
SDK	CrypKey.SDK7	ICrypKeySDK7Ptr
Float Record	CrypKey.SDK7FloatRecord	ICrypKeySDK7FloatRecordPtr

INSTALLING A COM OBJECT

The following procedure assumes that you have completed the CrypKey base product installation on the system you are using.

Procedure

1. Switch to the directory into which you are installing the files.
2. Execute the following command:

```
regsvr32 CrypKeyCOM7.dll
```

You are now ready to use the COM object from anywhere in the system. You do not have to copy the COM object file or register it more than once.

There are two methods of using the COM object with Visual Basic:

- referencing the COM Object explicitly
- referencing the COM Object by name.

The above methods are described in the sections that follow.

A third method, also described below, uses the COM object in C++.

REFERENCING A COM OBJECT EXPLICITLY

To use this method, start Visual Basic and choose the menu Project/References. In the References window, find the item named CrypKeyCOM7 Type Library and ensure there is a check beside it. Press **OK** to close the window. Now you can write your code like this:

```
Dim myObject As New CrypKeySDK7

Dim initResult As Long

' initialize CrypKey

initResult =
myObject.InitCrypKey("c:\my_directory\my_product.exe", _

    "<<replace this with your master key>>", _

    "<<replace this with your user key>>", 0, 300)

' check if call successful
```



```

If (initResult = 0) Then

    ' check if we are authorized

    Dim authResult As Long

    Dim nOptLevel As Long

    authResult = myObject.GetAuthorization(nOptLevel, 1)

    'check if call successful

    If (authResult = 0) Then

        MsgBox "You are authorized with these options/
        levels" & nOptLevel

        'you should run your program now

    Else

        MsgBox "You are not authorized to run this
        application!"

        ' you should abort your program now

    End If

Else

    MsgBox "Initialization failed!"

    ' you should abort your program now

End If

```

REFERENCING A COM OBJECT BY NAME

To use this method, no special project setup is needed. You can write your code exactly as shown in Sec. 5.4.3, except that the first two lines are replaced with the following four lines:

```

Dim myObject As Object

Dim initResult As Long

```

```

' create an instance of the CrypKey COM object

Set myObject = CreateObject("CrypKey.SDK7")

```

The rest of the coding, starting with CrypKey initialization, is similar to that shown in the previous section, *Referencing a COM Object Explicitly*, with minor modifications.

PRE-DEFINING DLL FUNCTIONS

To illustrate the ease of using the COM object, for matters of comparison, the old method of pre-defining DLL functions is described here.

To use this method, you must define each function in the CrypKey SDK 7 DLL that you call and ensure that the DLL is in the application or system path. This method is more confusing and prone to errors than using a COM object. It looks something like this in VB:

```

' EXTERNAL FUNCTION DECLARATIONS

' Crypkey Functions

Declare Function InitCrypkey& Lib "crp32dll.dll"

    (ByVal filepath$, ByVal MasterKey$, ByVal UserKey$,

    ByVal allow_floppy&, ByVal network_max_checktime&)

Declare Function SaveSiteKey& Lib "crp32dll.dll"

    (ByVal site_key$)

Declare Function GetSiteCode& Lib "crp32dll.dll"

    (ByVal site_code$)

Declare Function GetAuthorization2& Lib "crp32dll.dll"

    (ByVal decrement&)

Declare Function GetAuthorization& Lib "crp32dll.dll"

    (ByRef oplevel&, ByVal decrement&)

Declare Function Get1RestInfo& Lib "crp32dll.dll"

```

```

(ByVal which&)

Declare Function readyToTryDays& Lib "crp32dll.dll"

(ByVal oplevel&, ByVal numdays&, ByVal version&, ByVal
copies&)

Declare Function GetLevel& Lib "crp32dll.dll"

(ByVal DefinedLevels&)

Declare Function GetOption& Lib "crp32dll.dll"

(ByVal nDefinedOptions&, ByVal nOptionNum&)

```

The rest of the coding, starting with CrypKey initialization, is similar to that shown in the previous section, *Referencing a COM Object Explicitly*, with minor modifications, using a COM Object in C++.

USING A COM OBJECT IN C++

You must reference the object DLL in your source code. This picks up the type library for the object and allows the compiler to compile your application. Following is a sample of the source code for this.

```

#import "CrypKeyCOM.dll" no_namespace named_guids
void main ()
{

    // initialize COM library
    CoInitialize (NULL);

    {

        // create an instance of the COM object
        ICrypKeySDKPtr ptr;
        ptr.CreateInstance ("CrypKey.SDK");

        // initialize CrypKey properly
        long nResult = ptr->InitCrypKey
        (_T("c:\\example.exe"),

        T("6a6167abf35cb0253b91761971cfcecf79b6935d6e779
05f5b29a45badd3a4f21315cee18798f96ef1861455c0f1864
3746f1183580caa6a3f5ef6d8e3dd7e15da0f3ab21414b8fbe
549d385552f816406b15040070d636ab2153d715e71c6c2e

```

```

eda08f58503f64ce9183759075949ff249013d045b55907aa
6924c5d707546"),
T("D050 815C D1A2 A79D B1"), 0, 300);

// check if call successful
if (nResult == 0)
{
    // check if we are authorized
    long nAuthResult;
    long nOptLevel;
    nAuthResult = ptr->GetAuthorization (&nOptLevel,
1);

    // check if call successful
    if (nAuthResult == 0
    {

        printf ("You are authorized with these
options/levels %d",
nOptLevel);

        // you should run your program now

    }
    else
    {

        printf ("You are not authorized to run this
program!");

        // you should abort your program now

    }

}
else
{

    printf ("Initialization failed!");

}

}

}

```

COM OBJECT CONSTANTS

The CrypKey SDK 7 COM Objects use the data types described below for internal operation.

Table 10: Data type: *CKExplainEnum*

Error Name	Value	Groups of error codes to be used with the ExplainErr methods
CKAuthError	1	Authorization methods errors
CKSiteCodeError	2	Get Site Code methods errors
CKSaveSiteKeyError	3	SaveSiteKey errors
CKRegisterTransferError	4	RegisterTransfer errors
CKTransferOutError	5	TransferOut errors
CKTransferInError	6	TransferIn errors
CKDirectTransferError	7	DirectTransfer errors
CKInitializeError	8	InitCrypKey errors
CKReadyToTryError	9	Ready To Try methods errors
CKKillLicenseError	10	KillLicense errors
CKAcquireError	11	Acquire methods errors
CK HardDriveError	12	Hard Drive Serial Number methods errors
CKFloatSnapError	13	Floating License Snapshot methods errors
CKCrypKeyCOMInternal	999	Errors internal to this COM object

Table 11: Data type: *CKRestrictionTypeEnum*

Condition Name	Value	License types from GetRestrictionInfo
CKRestrictionTypeNone	0	license with no restrictions

Condition Name	Value	License types from GetRestrictionInfo
CKRestrictionTypeTime	1	license with a time (number of days) restriction
CKRestrictionTypeRuns	2	license with a runs (number of runs) restriction

Table 12: Data type: *CKBooleanValueEnum*

Condition Name	Value
CKBooleanFalse	0
CKBooleanTrue	1

Table 13: Data type: *CKMaxLengthEnum*

Parameter	Value
CKMaxLenComputerName	15
CKMaxLenUserName	15

INTERNAL COM STATUS MESSAGES

Table 14: Internal COM Status Messages

Message	Value	Meaning
CKInternalOk	-1000	Operation is normal.
CKInternalNullPointer	-1001	A NULL pointer was passed as a parameter.
CKInternalFloatRecordNumber	-1002	The call to FloatingLicenseGetRecord uses a record number greater than the number of entries available.
CKInternalCustomByteIndex	-1003	The call to CustomInfoGetByte or CustomInfoPutByte uses an index value that is too large.
CKInternalMax	-1004	Flags this value as the last value in the CKInternal table of values.

REFERENCING THE FLOATING LICENSE SNAPSHOT RECORD

The COM object contains functions used to encapsulate the Floating License Snapshot (FLS) record. The following table contains C++ and C#.NET prototypes for getting or setting each data item in the FLS record. The Item Names in this table parallel the names in the FLS record itself (see *Chapter 14: Function Reference*, FloatingLicenseSnapshot).

Table 15: COM C++ and C#.NET Prototypes for FLS Record Data Items

Item Name	Language	Prototype	Description
nId	C++	HRESULT get_nId (long *pVal);	Gets the unique internal ID.
		HRESULT put_nId (long newVal)	Sets the unique internal ID.
	C#.NET	property int nId	Gets/Sets the unique internal ID.
nUpdate	C++	HRESULT get_nUpdate (long *pVal);	Gets the internal timestamp of the last update.
		HRESULT put_nUpdate (long newVal)	Sets the internal timestamp of the last update.
	C#.NET	property int nUpdate	Gets/Sets the internal timestamp of the last update.
nStatus	C++	HRESULT get_nStatus (long *pVal);	Gets the status of the license; 0=running, positive number=queue order.
		HRESULT put_nStatus (long newVal)	Sets the status of the license; 0=running, positive number=queue order.

Item Name	Language	Prototype	Description
	C#.NET	property int nStatus	Gets/Sets the status of the license; 0=running, positive number=queue order.
Ntimestamp	C++	HRESULT get_nTimestamp (long *pVal);	Gets the timestamp of when the license was started; number of seconds since 1900.
		HRESULT put_nTimestamp (long newVal)	Sets the timestamp of when the license was started; number of seconds since 1900.
	C#.NET	property int nTimestamp	Gets/Sets the timestamp of when the license was started; number of seconds since 1900.
StrUserName	C++	HRESULT get_strUserName (BSTR *pVal);	Gets the user name.
		HRESULT put_strUserName (BSTR newVal)	Sets the user name.
	C#.NET	property string strUserName	Gets/Sets the user name.
StrComputerName	C++	HRESULT get_strComputerName (BSTR *pVal);	Gets the computer name.
		HRESULT put_strComputerName (BSTR newVal)	Sets the computer name.
	C#.NET	property string strComputerName	Gets/Sets the computer name.

Item Name	Language	Prototype	Description
strSpare	C++	HRESULT get_strSpare (BSTR *pVal);	Gets the data for internal use only.
		HRESULT put_strSpare (BSTR newVal)	Sets the data for internal use only.
	C#.NET	property string strSpare	Gets/Sets the data for internal use only.

Installing the CrypKey License Service

To install the CrypKey License Service, include *SetupEx.exe* and *CKS.exe* in your installation script. The file *SetupEx.exe* is an install program. The file *CKS.exe* is a self-extracting zip file of the six required set-up files.

Note: Don't extract contents of the *CKS.exe* zip file — *SetupEx.exe* expects to see this file as-is.

If, for example, InstallShield is used, the script should ensure that both *SetupEx.exe* and *CKS.exe* are copied into the installation directory (i.e., the directory where the user's program is installed). The InstallShield script only executes *SetupEx.exe*. To install the CrypKey License Service you must be an administrator.

ERROR CODES

Setupex.exe returns an error code if it runs into a problem. This code is returned in one of two ways:

1. In the process termination code, which can be read by the process that launched *SetupEx.exe*, if this is supported.
2. If (1) is not supported, the error code is written in ASCII to a file called *setupex.xco*, at termination.

Some installation programs are able to receive this error code. Processing these codes is optional.

The reported errors that are likely to be caused by a bug in your installation are:

```
#define NTDRVR_INSTALL_ERR_NOT_NT -1
```

```
#define NTDRVR_INSTALL_ERR_REMOTE_DRIVE -2

#define NTDRVR_INSTALL_ERR_CANNOT_COPY_FILE -3

#define NTDRVR_INSTALL_ERR_CANNOT_RUN_CKSETUP -4

#define NTDRVR_INSTALL_ERR_CKS_EXE_MISSING -5

#define NTDRVR_INSTALL_ERR_CANNOT_RUN_CKS_EXE -6

#define NTDRVR_INSTALL_ERR_MISSING_FILE -7
```

The following error is reported when you do not have permission to load CrypKey License Service files.

```
#define NTDRVR_INSTALL_ERR_REG_COULD_NOT_OPEN_
SERVICEMANAGER -8
```

You can clear the above error by logging in as a local administrator of the machine.

The following errors are unlikely to occur and should be reported to CrypKey:

```
#define NTDRVR_INSTALL_ERR_REG_COULD_NOT_OPEN_SERVICE -9

#define NTDRVR_INSTALL_ERR_REG_COULD_NOT_REGISTER_
DIRECTORY -10

#define NTDRVR_INSTALL_ERR_STARTSERVICE_COULD_NOT_OPEN_
SERVICEMANAGER -11

#define
NTDRVR_INSTALL_ERR_STARTSERVICE_COULD_NOT_OPEN_SERVICE
-12

#define NTDRVR_INSTALL_ERR_STARTSERVICE_COULD_NOT_
START_SERVICE -13

#define NTDRVR_INSTALL_ERR_STOPSERVICE_COULD_NOT_OPEN_
SERVICEMANAGER -14

#define NTDRVR_INSTALL_ERR_STOPSERVICE_COULD_NOT_
OPEN_SERVICE -15

#define NTDRVR_INSTALL_ERR_STOPSERVICE_COULD_NOT_
STOP_SERVICE -16
```

Note: It is important to require a reboot on an upgrade of the CrypKey License Service. This is because the updated License Service will not come into effect until after the reboot is completed.

```
#define NTDRVR_INSTALL_ERR_REBOOT_NEEDED -17
```

Note: If the error message is not documented, please refer to `crypkey.h`.

RUN MODE

CrypKey SDK 7 has two run modes — silent and verbose.

In silent mode (SetupEx.EXE /S), *SetupEx.exe* does not display messages on the screen. It runs complete checking and installs the CrypKey License Service if it can. If it cannot install the CrypKey License Service, it returns an error code. Since silent mode does not report errors, it is up to the installation program to handle them. Note: if the *setupex.xco* has a -17 in it you will want to prompt the user to reboot if using silent mode.

In verbose mode (SetupEx.EXE), *SetupEx.exe* displays any errors it encounters. It runs complete checking and installs the CrypKey License Service if it can. If it cannot install the CrypKey License Service, it returns an error code.

INSTALLATION STRATEGIES

Simplest Method:

For any platform, the easiest installation strategy is to copy the files into the installation directory and run *SetupEx.exe*. This sets up the service, if it should be set up (i.e., it is a local Windows NT directory). If there is a problem, the license service reports it to the user, thereby doing all the necessary interface work for you.

Applying More Control:

If you need more control over your installation, you can use our installation in silent mode by calling SetupEx.EXE /S. You then must retrieve the error codes and handle all errors yourself.

UNINSTALLING THE CRYPKEY LICENSE SERVICE

The CrypKey license service has the ability to do partial and complete uninstalls. If other directories are using the license service, the uninstall removes only the directory *SetupEx.exe* from the configuration. If no other directories are using the

license service, the uninstall stops the service and remove all the files used by the license service.

Note: You must include the uninstall in your uninstall package for this functionality.

Procedure

1. Locate *SetupEx.exe* in the directory you want to uninstall.
2. From the directory, run `SetupEx.EXE /D`.

This performs all required actions. If no other directories are using the license service, all license service files are deleted and a message notifies the user that the computer must be rebooted before another installation can be done. You can avoid this message by using the silent mode (`SetupEx.EXE /D /S`).

Note: If the user attempts another installation before rebooting and the service was removed, the following “Error -9” message appears:

```
Register Directory: Unable to open CrypKey License
Service. Reason: The specified service does not exist
as an installed service.
```

The above error message disappears after the computer is rebooted.

If you wish to uninstall even if there are other directories registered, you can use the `SetupEx.exe /U` command.

WARNING

Removing the CrypKey License Service using the `SetupEx.exe /U` will remove the CrypKey License Service from the entire machine. The CrypKey License Service that is running on the machine is servicing all CrypKey protected applications that may be installed on the machine. Running the `SetupEx.exe /U` on a machine that has more than one application which uses the CrypKey License Service will render the other applications using the License Service unuseable, until the CrypKey License Service is reinstalled.

We recommend you use the command option `/D` in release versions, and use option `/U` for test purposes only.

TESTING THE INSTALLER

For a simple test, place both *SetupEx.exe* and *CKS.exe* in any directory, and then call *SetupEx.exe*. You can now see your directory in *C:\Windows\Crypkey.ini*. If the CrypKey License Service had never been on your system, you would see that too.

SUPPORTED VERSIONS OF WINDOWS

CrypKey supports Windows NT, 2000, XP, 2003 and Vista. Only the Intel binaries are shipped. Contact CrypKey for up-to-date platform support.

Chapter 8: Protection

Protect your installations against hackers.

CrypKey SDK 7 offers a bullet-proof encryption and security device we call Stealth. In addition, CrypKey recommends some complementary counter-hacker strategies you can employ, described later in this chapter.

Why Stealth

Stealth is an important component of software security and the CrypKey Software Developer's Kit (SDK). You may implement SDK functions in your software, but without anti-hacking protection, an average skilled hacker can quickly change your code to skip the security.

Note: We recommend that you run Stealth on all CrypKey SDK-protected applications possible. Please note CrypKey will not run on a standalone .NET dll.

Note: For environments that cannot use CrypKey Stealth, see the following section *Counter-Hacker Strategies* in this chapter.

Stealth version 7.0 has evolved to the highest level of security yet. Although no one can *guarantee* that your program can't be hacked, we can guarantee that Stealth makes your program extremely tedious and painful to hack. Any version of your program that is hacked will not likely operate properly.

Stealth is a strong deterrent to:

- reverse engineering, and
- patching,

helping to ensure that the security logic you put in your program stays in your program. In less than 60 seconds, Stealth automatically implements a battery of security measures in your program.

Stealth protects your program in three ways:

1. It compresses and encrypts the program. Your executable file is smaller after you have "stealthed" it. Since the code is actually encrypted, reverse engineering and debugging of the file becomes extremely difficult.

2. It guards against patching while your program is in memory. It is technically possible to debug or redirect program execution (patch) around the security code while it is in memory. Stealth detects debuggers and patches and immediately halts program execution.
3. Stealth completely rewrites your program and runs it in memory in small pieces, minimizing the amount of code that must be exposed at any given time. This process provides an extremely strong level of protection.

Stealth is simple to use. Supply it with the input and output file paths and it does the rest, rewriting your EXE or DLL and building the above security features into them.

For your convenience, we have included two versions of Stealth:

Note: To run either version of Stealth, you must obtain authorization from CrypKey for the Stealth application. Please send your authorization request to Authorize@CrypKey.com.

User Interface Stealth (*Stealthui.exe*)

This version has a Windows user interface that allows you to browse and save the input and output file paths.

Command Line Stealth (*Stealthcmd.exe*)

This version allows you to easily automate the Stealth process by adding Stealth to your make files. It accepts the input and output paths as commandline parameters. The Stealth command syntax is as follows:

```
Stealthcmd.exe {input filepath} {output filepath}
```

where {input filepath} is the EXE or DLL you want to stealth and {output filepath} is where you would like the stealthed file to be written to. The file name can be the same if it is put in a different directory.

The command line version is designed to allow you to automate the stealth task by adding it to the end of your compiling process. Many compilers support a custom step like this, so consult your compiler's documentation for full details.

Visual C++ Custom Stealth Example

Procedure

1. Create a subfolder named "Treated" under the folder that your compiler outputs the target program or DLL.

2. In the VC++ IDE, select the menu item Project\Settings, then scroll the tabs to the right until you see on the **Custom Build** tab.
3. Enter the following:

Description: Stealth

Commands: c:\CrypKey\stealth\stelthcm.exe
 \$(InDir)\your.dll
 \$(OutDir)\treated\your.dll

Note: In the above, the command InDir can be written as InputDir. The parameter your.dll can also be given as *your.exe*.

Outputs: \$(OutDir)\treated\your.dll

Note: In the above, the parameter your.dll can also be given as *your.exe*.

Completion of the above sequence causes VC++ to put a “stealthed” copy of *your.dll* or *your.exe* in the Treated directory.

Stealth File Auto-Distribution

Stealth (both the interface and command-line versions) have a useful additional feature: they can pack all CrypKey distributable files, as well as your files, into your executable.

Stealth can even install the CrypKey License Service upon first run for you. For instructions on how to use this feature of Stealth, see *Chapter 5: Demo Programs, Installing the CrypKey License Service*.

The Auto Distribution feature ensures that:

- a. your EXE has all of the files it needs to run, no matter where it is;
- b. your EXE always uses the correct version of associated files;
- c. the CrypKey License Service is installed; and
- d. your EXE is self contained and easy to use.

Using the Interface (STEALTHUI)

Procedure

1. In the CrypKey\Stealth directory, click the *stealthui.exe* filename. The Stealth window opens.

2. Click **Display Site Code**. You will see the following message:

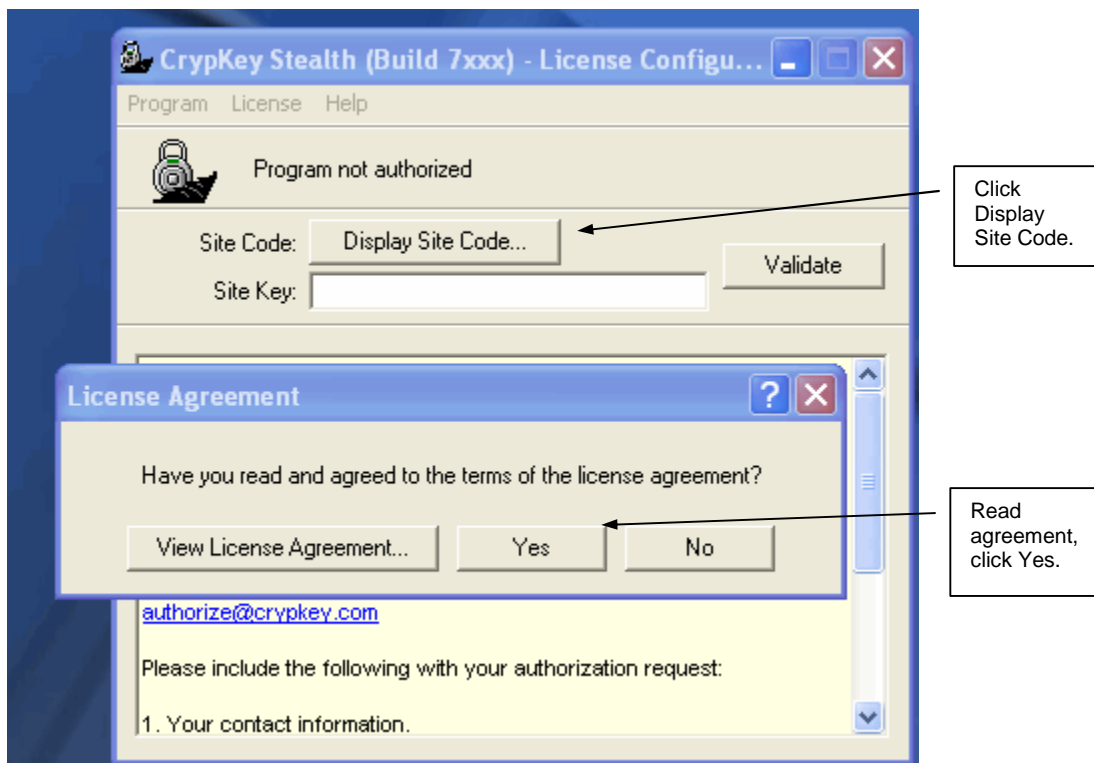


Figure 47 License Agreement pop-up

3. To read the terms of the agreement, click on **View License Agreement**. Then click **Yes**.
4. The system reveals the Site Code. Please send this Site Code to [Authorize@CrypKey.com](mailto:authorize@crypkey.com) with your company name and customer service number (with the Site Key field blank):

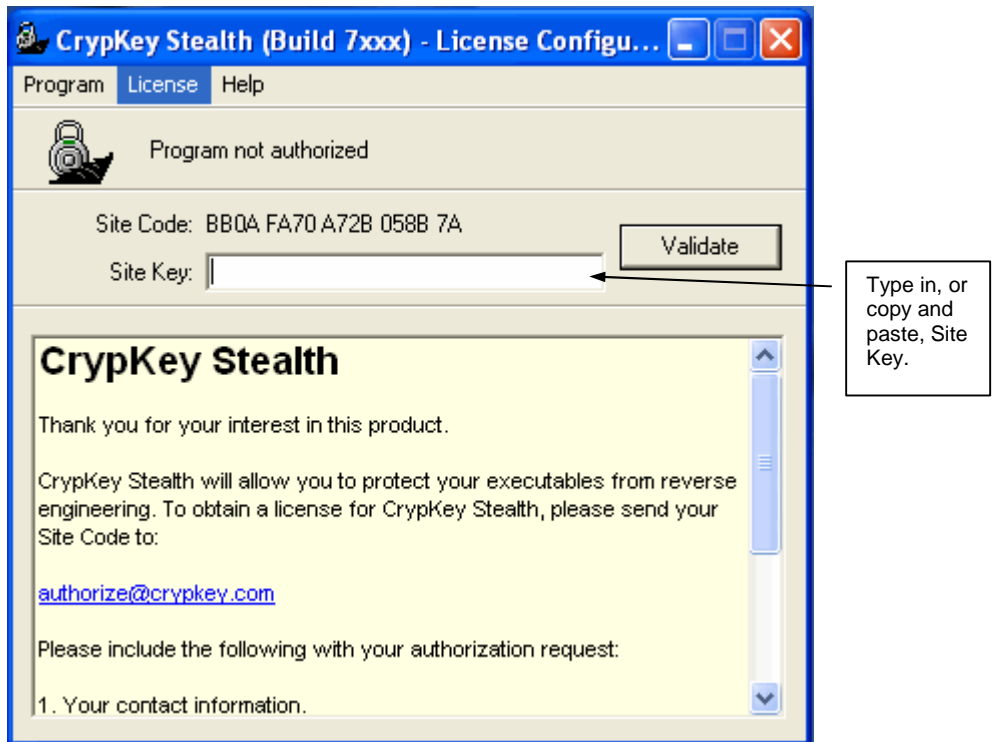


Figure 48 Enter Site Key for Stealth

5. In the Site Key field, type in the Site Key that was provided to you by CrypKey for CrypKey Stealth.
6. Click the **Validate** button. If the Site Key is valid, the system displays the Site Key and a message stating you are licensed to run the software.
7. In the Stealth authorization window, notice that the Site Code has now changed. This is standard in CrypKey. When CrypKey validates a program using a Site Key generated from the program's Site Code, it refreshes the Site Code for that program.
8. Click **OK** to proceed. The system displays the CrypKey Stealth window:



Figure 49 Stealth Configuration window

9. In the above window, use the **Pick File** buttons to browse your directories and enter the following information:
 - in the **Program to compress** field: the full path of the source dll or executable file that you wish to protect.
 - in the **Output program** field: the full path of the protected dll or executable file that is to be distributed. If you want to place this file in the same directory as the source file, you must use a different file name. For example:



Figure 50 Stealth Configuration window

- If you don't choose a different output name for a file saved to the same directory, you will see the following message. Click **OK**.

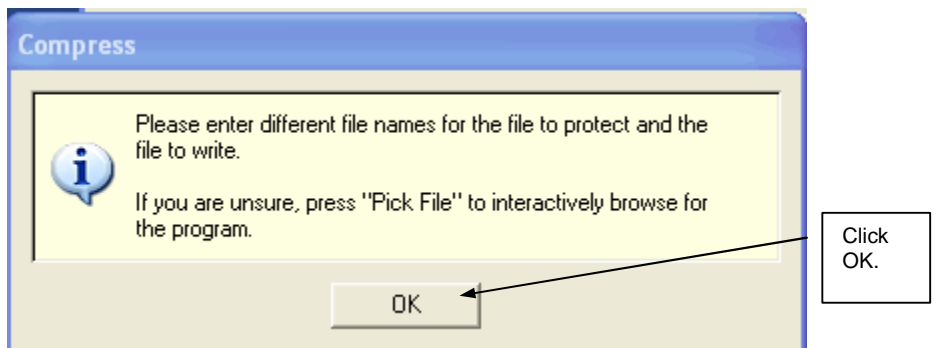


Figure 51 Stealth pick file names message

Note: We recommend you create an input and output directory. This will keep the unprotected and protected files separate.

- Optionally, select the checkbox “Attach and auto-extract additional files located in this folder” if you wish to have Stealth automatically install the CrypKey License Service when your customer installs your product:



Figure 52 Activate Pick folder option

11. Optionally, select the checkbox “Attach and autoextract additional files in this folder”. This activates the **Pick folder** button, which you can use to find and enter the name of a folder. During compression, Stealth extracts files from this folder and packs them into the executable file for your application.

Note: Any CrypKey files placed previously in the directory AutoDist are now packed. If there are files you do not need, simply remove them before running the Stealth interface. Similarly, other files that you have placed in the UserDist directory are now packed.

Note: StealthUI.exe allows you to save the configuration and specify a different user directory for each exe.

12. Use the **Pick folder** button to select the folder where the CrypKey License Service and auto-extracted files are placed.
13. To save your Stealth configuration, click the **Save** button. Or, if you are not planning to stealth your software during the current session, **Close** this window.
14. When you have finished the Stealth configuration and are ready to stealth your software, click the **Begin compression** button.

15. The system displays the following pop-up window, overlaying the Stealth Configuration window, and gives you a successful completion message when the compression is done.
16. Click **OK**.

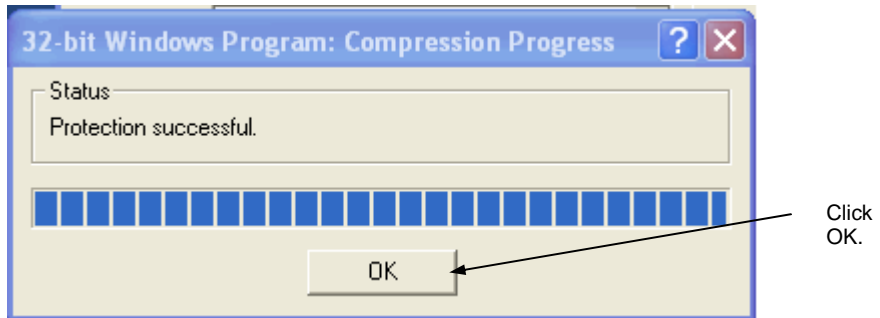


Figure 53 Stealth Compression begun

Counter-Hacker Strategies

Here are some measures you can take when developing your software to help prevent users from tampering with your product. For example, ensure your software responds with error codes when a function fails.

These measures are designed for environments that do not include CrypKey Stealth. When distributing your software, use the Stealth program that is included in CrypKey SDK 7. For details on using Stealth, see the previous section

Note: The measures presented here are not guaranteed unbreakable, but are designed to deter license system tampering and reveal tampering when it occurs.

Anti-tampering Measures

Introduce anti-tampering measures to your software with features that:

- a. prevent the license system from being compromised through the substitution of an impostor license system (i.e., one that always grants all license requests);
- b. prevent the license system from being compromised through multiple patches (there is no set of patches that can defeat the license system for all applications);
- c. prevent, through a set of guidelines, any single application's license verification subsystem from being defeated by any single patch; and
- d. require an obvious and intentional effort to defeat for each application.

The following guidelines may go against some accepted coding practices. It is your choice whether or not to implement them.

- Do not compare the authorization result immediately after calling a CrypKey function. Instead, save the result and compare it later. This is “security by obscurity”.
- Do an extra check on authorization in some obscure but frequently used part of the program. If the result is unauthorized, obtain the system time, do nothing until the time is an even multiple of 5 (for example), and then disable something important.
- Call the CKChallenge32() function, especially when using a CrypKey DLL or external linking. This is a very difficult function to hack.
- Incorporate an internal and obscure checksum over the code that deals with the license system and with the verification. Obscure the checksum code if you can.
- If applicable, take over any debug and single-step vectors that might be used by debuggers. **Take great care when using such an advanced measure.**
- Add random items to the application, such as meaningless reads, compares, and subtractions. In situations where a hardware monitor is used, this may increase the number of hardware breakpoints that occur.
- Verify the result more than once. Provide a faux compare immediately after the call to GetAuthorization().
- Don't do a comparison with a single compare. Instead, perform some mathematical operations on the result, computing another result that is verified later in the code.
- For platforms that permit self-modifying code and where the actual challenge algorithms are included in the code, hide the challenge algorithms by encrypting the code. Decrypt only long enough to verify a challenge and then re-encrypt it. Checksum the code that calls the encrypt/decrypt routines to prevent alteration. Have the entry point in the license module verify the code offset (if applicable) from which it was called. Encrypting the algorithms makes them more difficult to locate in the code. Also, encrypt the Master Key and the User Key.
- If the host operating system permits, place most of the authorization code and data into overlays that can be discarded.
- Call GetAuthorization() periodically (i.e., every 15 minutes).

- These are just a few possibilities to consider. Think about anti-virus techniques and the measures taken to prevent code modification. Those same anti-tampering techniques are applicable here.

ERROR CODES

Always have your program respond with specific error codes when a function fails. SLAPI has numerous functions with many different error codes. When something goes wrong with a particular function, it is a tremendous help to know specifically which function and operation is failing. The problem can be determined and resolved much quicker when this information is available.

Always call `InitCrypkey()` and `GetAuthorization()` before any other `CrypKey` functions.

Always check the return on `InitCrypkey()` and do not let the program run if the return is less than zero.

Ensure `EndCrypkey()` is called no matter how your program is exited. It is a common mistake to call only `EndCrypkey()` in your File/Exit menu routine. If the user exits by double-clicking the title bar, the `EndCrypkey()` function is skipped, which causes problems. This practice is imperative in network license situations as the sessions are not released unless `EndCrypKey()` is called.

Chapter 9: Network Licenses

CrypKey SDK 7 offers control of the licenses you issue, even when your software is distributed across a network..

CrypKey 7 (CrypKey Enterprise) allows flexible and on demand licensing of software over a network, without resorting to the use of a drive share mode, as with previous versions. This means you can limit the number of users and computers using your software, and transfer licenses based on individual users and computers over multiple domains on the same network.

With little or no programming, any CrypKey-protected program can easily act as either a License Server or a Client.

- Automatic restart of servers on reboot
- Supports an unlimited number of redundant servers
- Transfer of licenses across the network without any special directory read-write shares.
- Specify the number of floating licenses to transfer.
- Add licenses to a server that has existing licenses.

To run CrypKey SDK in client/server mode, there are three new utilities to make configuring server licenses easier and quicker:

- Enterprise License Manager (ELM)
- Network License Manager (NLM)
- Client License Configuration (CLC)

These utilities provide your customer with the ability to configure how their licenses are used on a network. Use your own discretion to decide whether to provide these utilities to the customer: be aware that there is a potential for a license to become disabled if the user is not familiar with the purpose and operation of the new utilities. Please refer to the user manuals for these utilities for more detailed information.

ENTERPRISE LICENSE MANAGER – see the ELM.rtf for more information on the functionality of the ELM.

The ELM offers the ability to transfer licenses and view license usage over the network. It can be used to modify network license configurations and authorize servers.

NETWORK LICENSE MANAGER – see the NLM.rtf for more information on the functionality of the NLM.

The NLM provides the user with the ability to see the server available on the local machine, create network license configurations and manage local server licenses. The NLM can be launched from the Control Panel, and provides a button to launch the ELM.

CLIENT LICENSE CONFIGURATION – see the CLC.rtf for more information on the functionality of the CLC

This utility is required when you would like the client to create or change the *cec.ini* file.

Installing the Utilities

For these applications to work correctly together, ensure that the folder containing the network license contains the *ckconfig.dll* and the Enterprise License Manager executable. The *ClientLicenseConfig.exe* is a stand-alone application that can be saved to the license folder.

The ClientLicenseConfig application can be installed on the client computers if you want the end user to have the ability to control which servers or domains they can access for a license.

Who should use the Utilities

CrypKey recommends that the network administrator or IT person be given the responsibility of using these utilities to create the client and server configuration files. The client configuration file (*cec.ini*) is then distributed to the client computers.

You must configure your installation to do the following, based on the mode of operation you have selected:

Table 16: Operation Mode and Installation Configuration

Mode	Task
Normal	<ul style="list-style-type: none"> - all files installed into local application folder - install must launch <i>SetupEx.exe</i> to install the CrypKey License Service
Drive share	<ul style="list-style-type: none"> - all files installed into drive share folder on server - run <i>SetupEx.exe</i> locally from the server, to install the CrypKey License Service.
CrypKey Network License	<p>Client Side: protected application is installed, along with:</p> <ul style="list-style-type: none"> - OS Specific Files - CrypKey Libraries - Mode Specific Files; <p>Setup.exe and CKS.exe NOT REQUIRED.</p> <p>Server Side: protected application is installed, along with:</p> <ul style="list-style-type: none"> - OS Specific Files - CrypKey Libraries - Mode Specific Files <p>Run SetupEx.exe locally from the server, to install the CrypKey License Service. Ckconfig.dll</p> <p>All files go in the application folder.</p>

Note: The protected application does not have to be installed on the server. It is only suggested here as it will logically contain the interface to manage your license requirements. If you do not install your application on the server, you need some other method of authorizing a license at the server. You can do this by creating a separate license manager to provide the network administrator with an interface to the licensing functions. Essentially, a license manager is an application that provides the user with the necessary interface components to enter a Site Key and initialize a license.

Regardless of how you manage this step, you should now be at the point where your customer can install your application.

Chapter 10: Moving Protected Programs

Your customers can transfer their licenses without even contacting you!

The license transfer system is an optional feature that allows your clients to transfer their licenses from one computer or directory to another without contacting you.

There are a number of different options for transferring licenses:

Direct Transfer: Used to move a license from one local or networked directory to another.

Media Transfer: Used to move a license from one computer to another when the two computers are not connected via a network. This type of transfer requires writeable media such as a floppy disk or USB key.

AcquireTransfer: Used to obtain a license from a licensed machine that is available on the network.

eTransfer: Gives you the ability to transfer a license over the Internet to a Casper server, then take the license off the server when you are ready to license the software in a different location. eTransfer requires Casper eRegister or eCommerce.

Network License Transfer: Gives you the ability to move one or more licenses out of a network license pool. ELM utility is required to perform this type of transfer. This is the only transfer type that allows individual licenses to be transferred out of network license pools.

Note: You can only transfer licenses that have been physically generated with CrypKey Site Key Generator or Casper. Ready To Try licenses can not be transferred.

Direct License Transfers

The `DirectTransfer()` function must be run from a program that has a valid license (the source program) and given a directory path to an existing copy of the same program that does not have a valid license (the target program).

The function simply transfers the license from the source directory to the target directory. The target directory must be local (i.e. on a drive located in the same computer) to the source directory or the directories must be connected by a network and have a CrypKey License Service running in both locations.

Example

If you have a program with one valid fixed license in `C:\Temp` and you want to move it to `C:\Apps\Bingo`:

1. Install your program to `C:\Apps\Bingo`.
2. Run the program that is in `C:\Temp`. Call `DirectTransfer()` and provide it the pathname of "`C:\Apps\Bingo`". Instantly, the copy in `C:\Apps\Bingo` has a valid license and the copy in `C:\Temp` does not.

Media Transfer

The Media transfer process involves transferring the special license files and is handled entirely by the CrypKey system. At no time during the transfer process is your license vulnerable to reproduction by bit copiers, file duplication, or file tampering.

To transfer a license from one computer to another, you must have a copy of the program installed and authorized on one computer and a copy of the program installed but not authorized on another. Assuming that the transfer is from an already-licensed source computer to an unlicensed target computer via a form of writeable media, use the following procedure.

Procedure

1. Start the program in the target computer and register the transfer by telling the program to call the `RegisterTransfer()` function. The user must supply the path to the writeable media. This function puts a registration imprint file on the writeable media. Take out the writeable media from the target computer and place it in the source computer.
2. Start the program on the source computer and tell it to call the `TransferOut()` function. The user must supply the path to the writeable media. This function reads the registration imprint file and writes a matching license imprint file to the writeable media. It also discontinues the license at the source if it had one copy or it decrements the license at the source by one if it had multiple copies. Take out

the writeable media from the source computer and return it to the target computer.

3. Restart the program in the target computer and tell the program to call the `TransferIn()` function. The user must supply the path to the writeable media. This completes the transfer and discards the intermediate imprint files.

If you want your program to be able to transfer its license via writeable media, you must program your application to allow the user to initiate the following functions:

- `RegisterTransfer()`
- `TransferOut()`
- `TransferIn()`
- `Acquire License()`

Chapter 11: ELM

Enterprise License Manager (ELM) is a utility for transferring license over a network.

Overview

Previous versions of CrypKey used the 'drive share' method to manage network licenses, and the end user (network client) required read and write access to the directory on the server where the license was located.

With CrypKey 7, the sharing of licenses is communicated invisibly and automatically through the network, and the ability to do this is built in to every C7PP (CrypKey 7 protected program).

Network clients do not require any specific privileges to the license directory on the server.

CrypKey SDK 7 Network Licensing Mode

This mode consists of the Server configuration:

Server Mode

A program operating in this mode is able to share its network license with computers running as a client. The Enterprise License Manager provides the functionality of license transfers between servers across the network. Note that a 'Server' within the context of CrypKey operating modes, does not refer to a physical server on a network. It is actually a state of operation for an application protected with CrypKey. A client computer on a network can be switched to a server, in order to serve licenses to other clients, or to transfer a license out of a network pool.

New to CrypKey 7

There are several new files:

CRP32002.ngn

This file replaces the crp32001.ngn of CrypKey SDK 6. The CRP32002.ngn (CrypKey Engine) runs as a separate process, which allows you to debug your application while using CrypKey. The ngn process can run on computers that are in Client/Server mode, Drive Share, or Normal mode. Computers running in Network-Licensing - Client mode will not run the CrypKey Engine, but will use the process running on the server.

Note: In addition to the debugging capability of CRP32002.ngn, CrypKey SDK 7 includes the Stealth feature, our anti-hacking layer. You must add Stealth to your application to be fully protected (see *Chapter 8: Protection*).

***.CEC.INI**

This file contains the configuration parameters for a computer running in Client mode.

Note: The cec.ini should never be manually modified. You should only configure the cec.ini using the CLC utility provided. You can preconfigure a cec.ini with the CLC utility and send the preconfigured cec.ini to your customers with the client installation package.

***.CES.INI**

This file contains the configuration parameters for a computer running in Sever mode.

Note: The ces.ini should never be manually modified. You should only configure the ces.ini using the ELM utility provided.

***.CED File**

The .CED file is required for correct operation of the Client/Server mode. The .CED file needs to exist in the folder that contains the license. This file enables ELM to work with your product. This is a custom file created for your company by CrypKey when developer keys are requested.

Enterprise License Manager.exe

This application is provided to modify the operating parameters of CrypKey 7, and provides the following functions:

- Automatically creates the *ces.ini* files
- Provides the ability to edit *ces.ini* files

- Allows user to view network license usage.
- Allows license transfers – please note that only servers can transfer licenses from network pools. A client must be changed to a server to obtain a license.

Implementation Notes

If you are just starting out with CrypKey, follow the procedures outlined in the CrypKey Instant or CrypKey SDK 7 manuals.

If you are upgrading from CrypKey SDK 6, then you need to re-link and recompile your application using the new libraries provided in version 7.

ELM USER

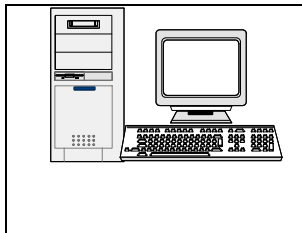
If your application is going to run on the end user's computer in Normal Mode, then you do not need to distribute the ELM program.

The ELM program should be provided to the network administrator, or the person who will be installing and maintaining the license on the server in client/server mode.

The network administrator is best suited to determine if and how the application should be distributed to the end users, and how to educate them in the use of it. A good idea is to provide a preconfigured *cec.ini* file as part of the client install. Then install the ELM utility in the license directory on the server.

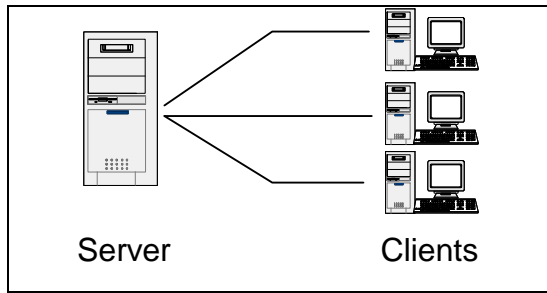
For the purpose of transferring licenses, the administrator may decide to simply copy the ELM program to the user's computer, transfer the license, and then remove the ELM program.

SAMPLE CONFIGURATIONS



Component	Status	Comment
Operation Mode	Normal	Fixed License is Installed on local machine
ELM program	N/A	N/A
Crp32002.ngn	Required	Runs as local process
CrypKey License Service	Required	Installed by <i>SetupEx.exe</i> and <i>CKS.exe</i>

Figure 54 Configuration 1: Stand-alone installation – Normal mode

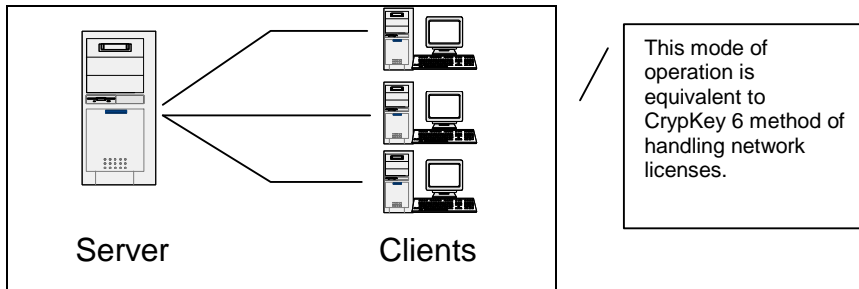


SERVER CONFIGURATION		
Component	Status	Comment
Operation Mode	Server	Licenses are held in this location
ELM program	Required	Manages licenses served by the "server" – machine with the .ces.ini file
Crp32002.ngn	Required	Runs as local process
CrypKey License Service	Required	Installed from <i>SetupEx.exe</i> and <i>CKS.exe</i>
CES.INI	Required	Created and configured using the ELM utility
CKCONFIG.DLL	Required	Contains Server specific information

CLIENT CONFIGURATION		
Component	Status	Comment
Operation Mode	Client	Obtains license from an authorized server
Crp32002.ngn	Required	Does not run as a local process.
CLC program	Optional	This utility is not required, but if you would like clients to be able to make their own configuration

CLIENT CONFIGURATION		
Component	Status	Comment
		changes to the software they will need the utility to do so.
CrypKey License Service	Not Required	
CEC.INI	Required	Either pre-configured using CLC utility and distributed to the client, or create by the client using the CLC utility

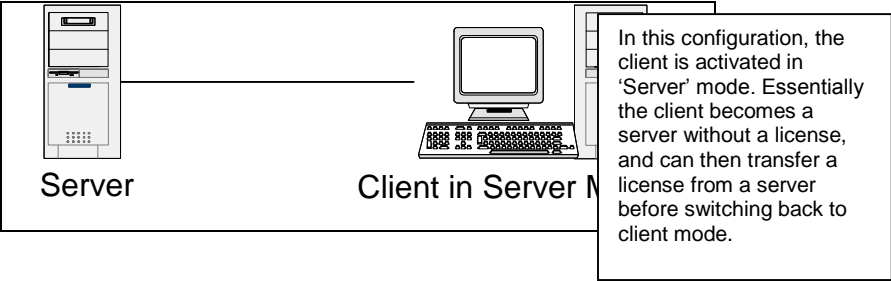
Figure 55 Configuration 2: Client/Server Configuration



SERVER CONFIGURATION		
Component	Status	Comment
Operation Mode	Server	License is Installed on local machine
Crp32002.ngn	Required	Runs as local process
CrypKey License Service	Required	Installed from <i>SetupEx.exe</i> and <i>CKS.exe</i>
Specific user security	Required	Read and write access to license directory
CLIENT CONFIGURATION		
Component	Status	Comment
Operation Mode	Client	Runs the software from a central licensed server location.
CrypKey License Service	Not Required	Running on server

Figure 56 Configuration 3: Drive Share Installation

Advanced networking: you are able to create a client/server mode for an application and have users map to the client software over a LAN or WAN. The client software then looks over the network for a license on a server for the software. Please contact Support@Crypkey.com for more information.



SERVER CONFIGURATION (for both Computers)		
Component	Status	Comment
Operation Mode	Server	Software is Installed on local machine
ELM program	Required	Needed to change modes and transfer the license from the network pool
Crp32002.ngn	Required	Runs as local process
CrypKey License Service	Required	Installed from <i>SetupEx.exe</i> and <i>CKS.exe</i>
CES.INI	Required	

Figure 57 Configuration 4: License Transfer Configuration

Using the Enterprise License Manager

Purpose of the ELM



Enterprise License
Manager.exe

The Enterprise License Manager (ELM) provides the ability to manage the licenses of the available servers, and to transfer licenses. The ELM program is designed to run from the same directory location as the license. If you try to run the program from a remote directory, or select a license outside of the application directory, it will result in limited functionality.

To use the ELM to manage server licenses, you must select a license. Do so from the Actions Menu. Select the *.CED file in the same directory as the ELM program.

RUNNING IN SERVER MODE

A protected application knows whether or not it should run in Server mode, based on the status flag in the *ces.ini* file. If the status=disabled, or the *ces.ini* file does not exist, then the application operates in Normal mode.

In Server mode, you can:

- Activate or Start the server, which makes its license available to other clients and servers on the network.
- Stop the server, which blocks access to its license by other clients or servers.

Server mode: opens Network License Configuration wizard (see the *Network License Manager Guide*).

Edit Menu

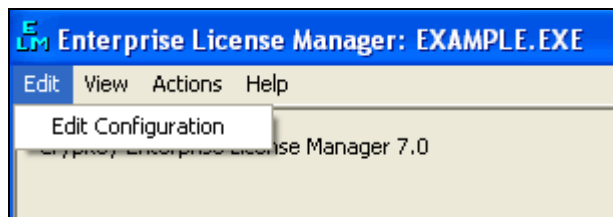


Figure 58 Edit Menu: Edit Configuration option

View Menu

The View menu provides several views, only available in Server mode:

- **Network Clients view.** This view shows which users are accessing a network license, and will create a file called ELM.LOG containing output of the view. There must be a valid network license to be able to see information on network clients.



Figure 59 View Menu: Network Clients option

- **All CrypKey Servers.** This view shows all other systems on the network that are running in server mode for the selected application:

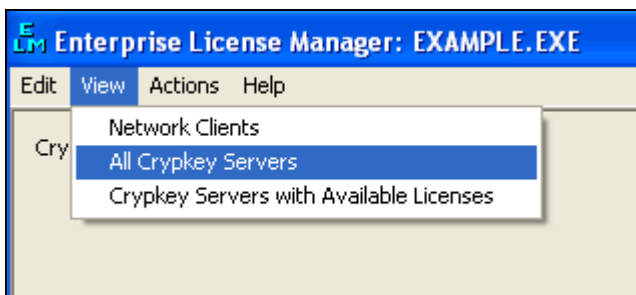


Figure 60 View Menu: All CrypKey Servers option

- **CrypKey Servers with Available Licenses.** This view will show all other systems on the network that are running in Server mode, and have licenses available for use or transfer:

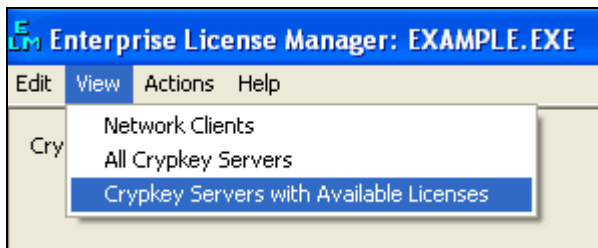


Figure 61 View Menu: All CrypKey Servers with Available Licenses option

Actions Menu

The Actions menu provides the following functions:

In Server mode:

- Start Server: changes the status to enabled in the *ces.ini* file.

Note: Please ensure that the correct .ced file is selected as it will start a server for the selected .ced file.

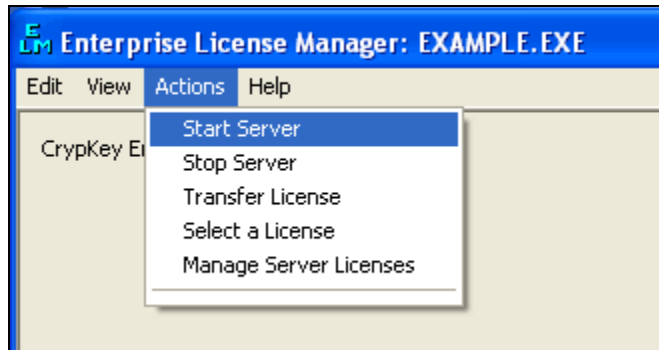


Figure 62 Actions Menu: Start Server

- Stop Server: change the status to disabled in the *ces.ini* file:

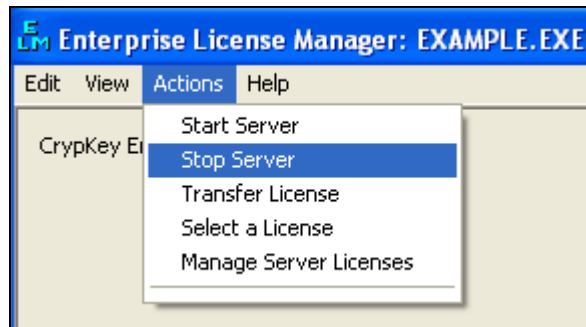


Figure 63 Actions Menu: Stop Server

- Transfer License: opens a window that allows you to transfer licenses between servers. This functionality requires that the machine which is to receive the license, is running in Server mode. Once it has a license, then it can run in Normal or Server mode:



Figure 64 Actions Menu: Transfer License

- Select a License:

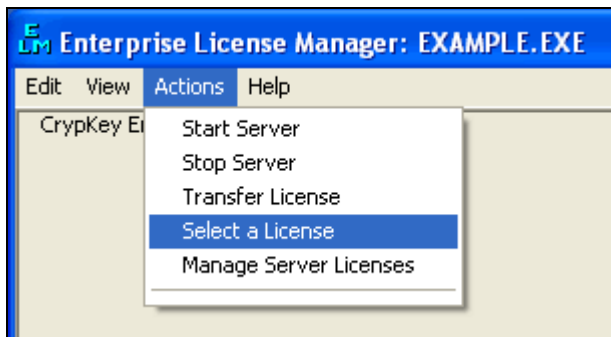


Figure 65 Actions Menu: Select a License

- Manage Server Licenses – allows you to see the license information for a specific server.

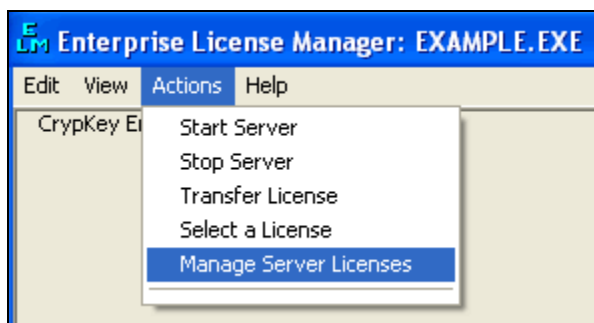


Figure 66 Actions Menu: Manage Server Licenses

Transferring Licenses

LICENSE TRANSFER MANAGEMENT SCREEN

This is the screen you use to manage the transferring of licenses.

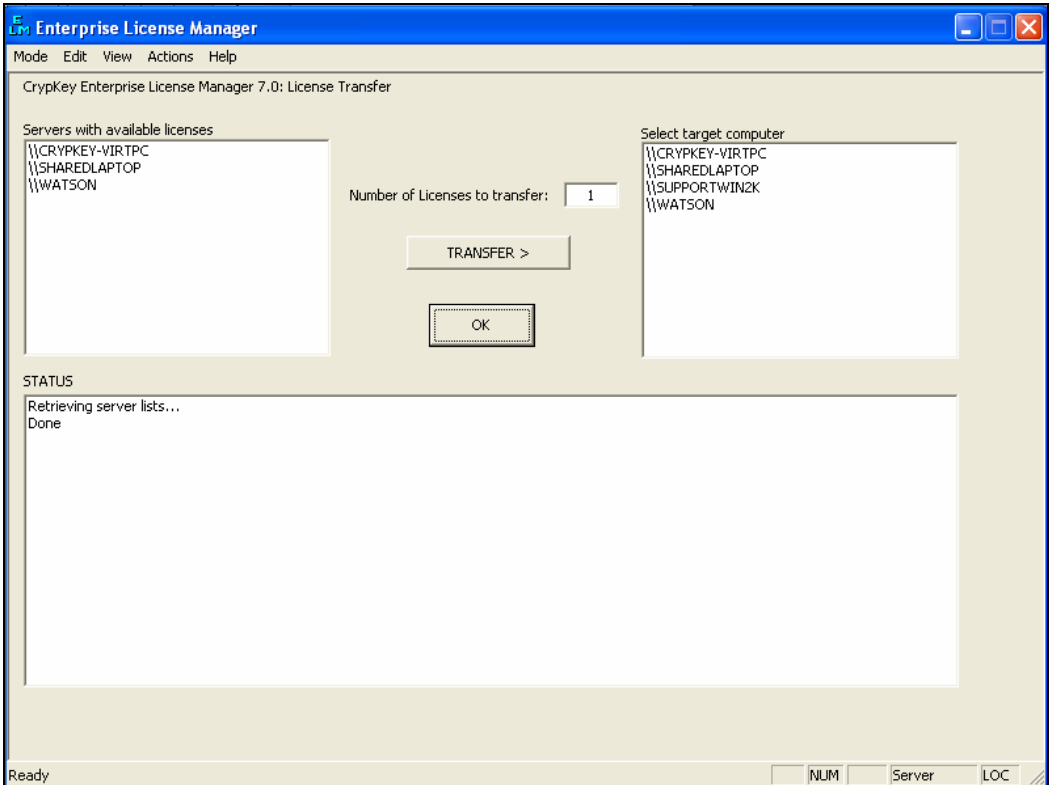


Figure 67 Enterprise License Manager screen

Servers with licenses are listed on the left of the screen. All Servers are listed on the right of the screen, including those that do not have a license.

To transfer a license:

1. Select the source server on the left.
2. Select the target server on the right.
3. Select the number of licenses you want to transfer.
4. Click on the **Transfer** button.

The transfer process will take approximately fifteen seconds, but may vary. Once the transfer process is done, the server lists refresh and you will see the computer

appear in the lefthand list (if it has received its license correctly). This shows this server now has available license (s).

SOME NOTES ABOUT LICENSE TRANSFERS

There are a few things to be aware of when transferring licenses:

- a. The license transfers can be used to add licenses to, or remove licenses from, an existing pool of floating licenses.
- b. You can transfer a license to a computer that does not have a license.
- c. You can transfer (add) a license to a computer that has a floating license.
- d. You cannot transfer a license to a computer that has a fixed license. Only network licenses can be transferred using ELM.
- e. In order for a computer to appear on the 'Servers with Available Licenses' list, it must have at least one floating license available, and be running in server mode.
- f. You can not transfer licenses that have different authorizations. So if one server has different options enabled than the machine you are transferring from, you can not transfer the license.

Troubleshooting

The ELM is intended to be run locally, from the folder that contains the applications license. If you try to run the ELM from a different context, then certain menu options will become disabled.



LOC – indicates correct local operation. If this says REM, then some functions will be disabled.

If you run into issues not documented here, please contact us at Support@CrypKey.com. We look forward to your suggestions for improving the Enterprise License Manager.

Chapter 12: NLM

The Network License Manager is used to control server licenses on a local computer.

Network License Manager (NLM)

To Launch the NLM, either launch the NLM from the Control Panel (see below), do the following or go to the C:\Program Files\CrypKey SDK\Build 7xxx\License Management Utilities directory.

1. Launch the NLM from the control panel.

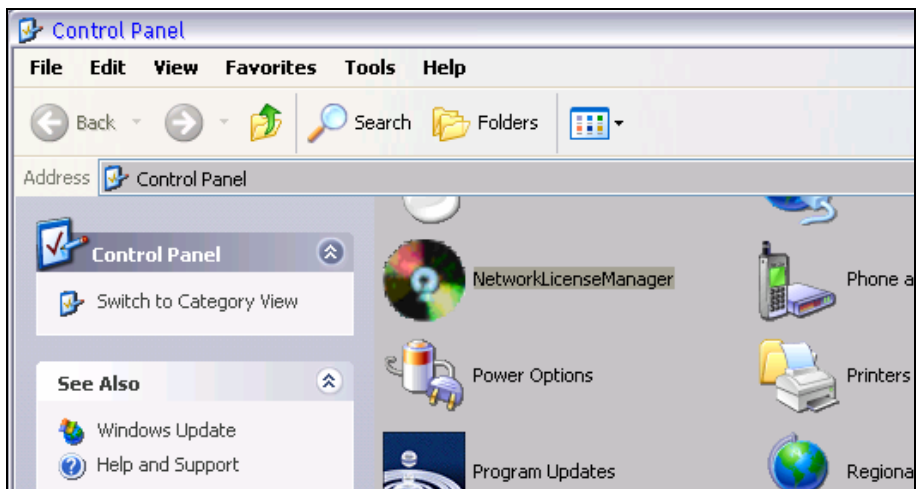


Figure 68 Launch NLM from the Control Panel

2. The NLM main screen appears:

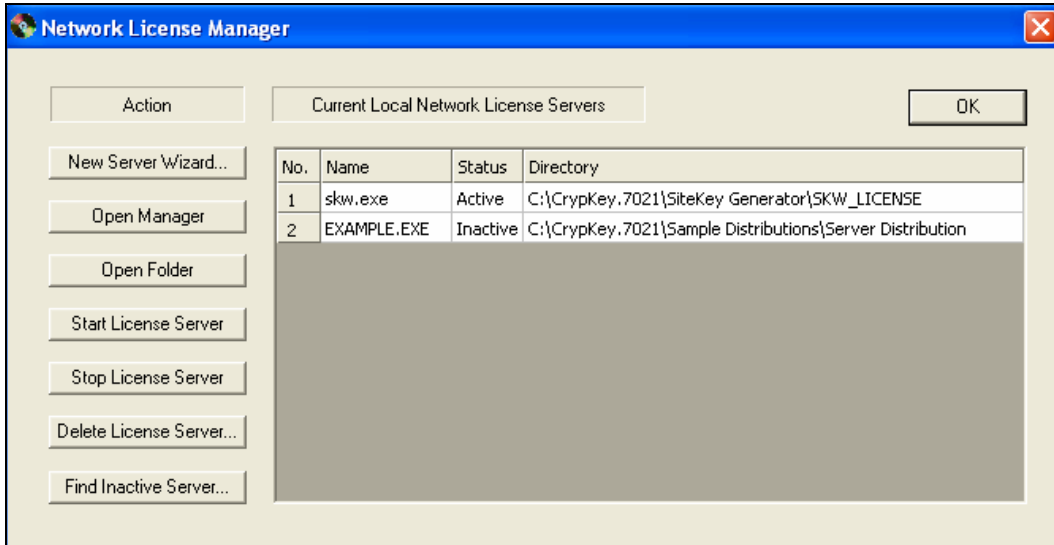


Figure 69 Network License Manager Main screen

3. From here you can select:
 - a. **New Server Wizard** – to create the configuration files for CrypKey server licenses on the local computer.
 - b. **Open Manager** – to launch the Enterprise License Manager application for the selected license.
 - c. **Open Folder** – to launch an Explorer window for the location of the selected license.
 - d. **Start License Server** – to start the server process for the selected license.
 - e. **Stop License Server** – to stop the server process for the selected license.
 - f. **Delete License Server** – to delete the configuration for the selected license.
 - g. **Find Inactive Server** – list server licenses that do not have an attached, active server process.

New Server Wizard

When you select New Server Wizard, you will be presented with a series of screens that allow you to create a server configuration for a selected license.

The first screen you will see is the New License Server Configuration:

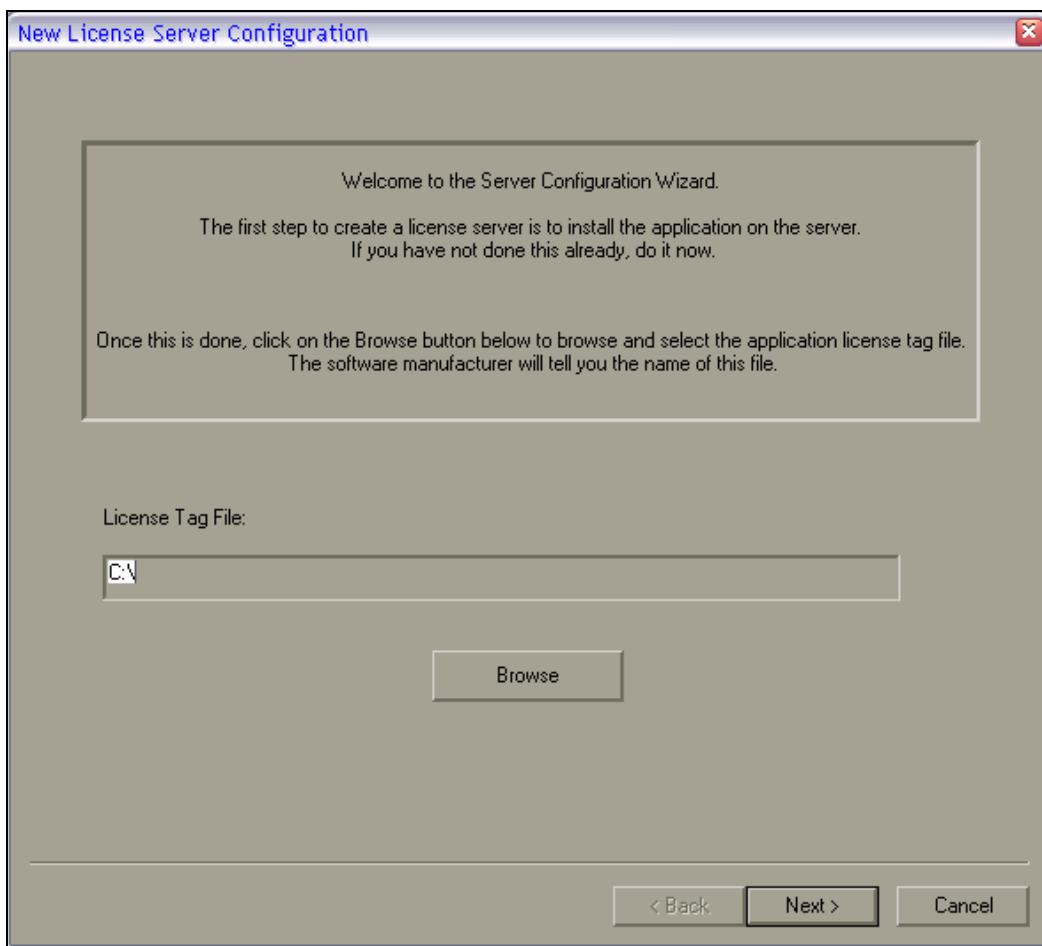


Figure 70 Server Configuration Wizard

4. Browse to the location of the license that you want to share on the network:

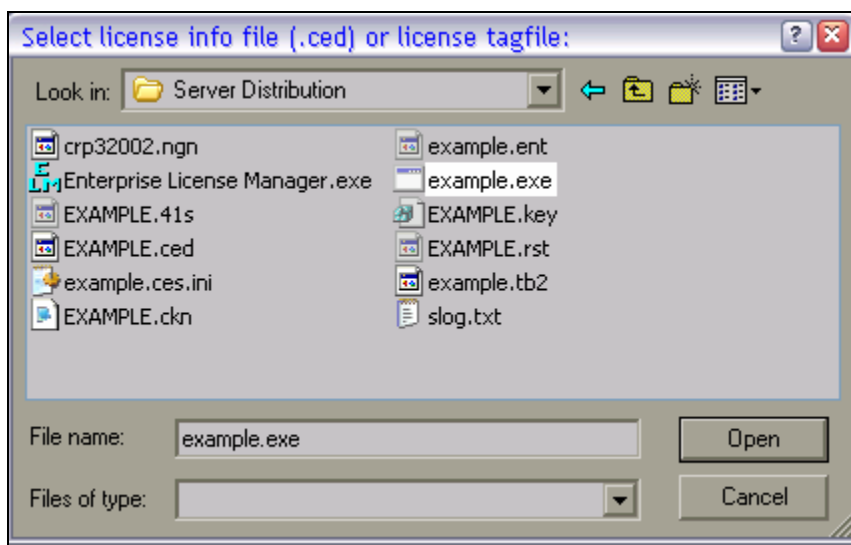


Figure 71 Select License

5. Once you have selected the location, click **Next** to continue:

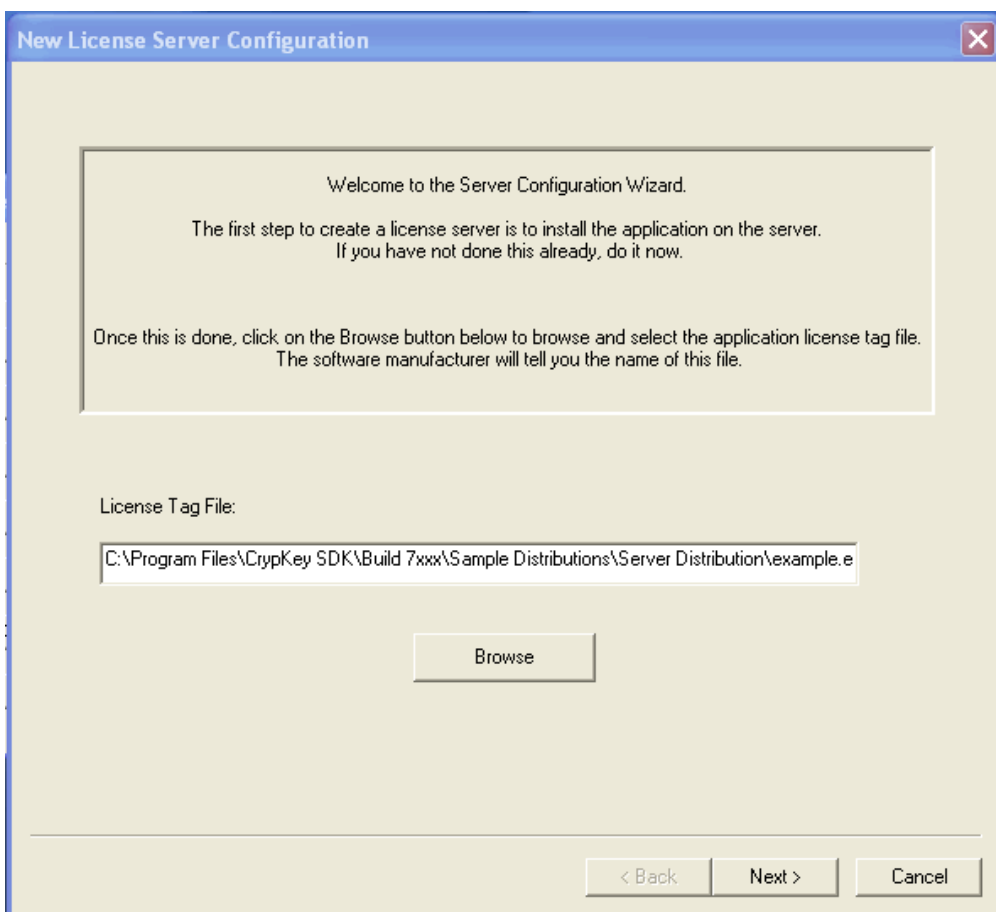


Figure 72 License Selection processing

6. The License Domains screen displays (see Figure 73). From here you can select the domains that may access your license.

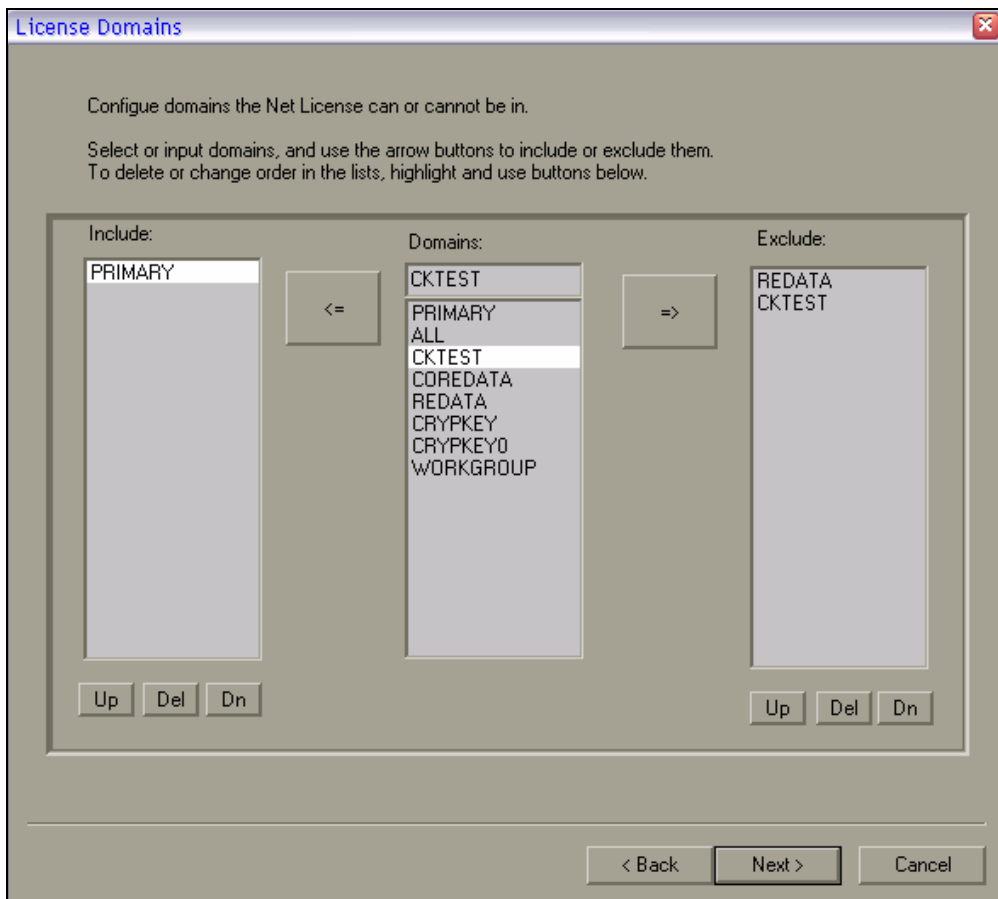


Figure 73 License Domains screen

7. You can exclude certain domains from accessing your license. Use the **Up Dn** buttons to change the access priority of the served domains: domains at the top of the list will have access to a license before those lower on the list.
8. Use the **Del** button to remove items from the list.
9. Click **Next** to continue.

Note: The Initializing Data screen may appear throughout the process, indicating the application is busy collecting information from the network. This screen (see Figure 74) may take a little while to gather information.

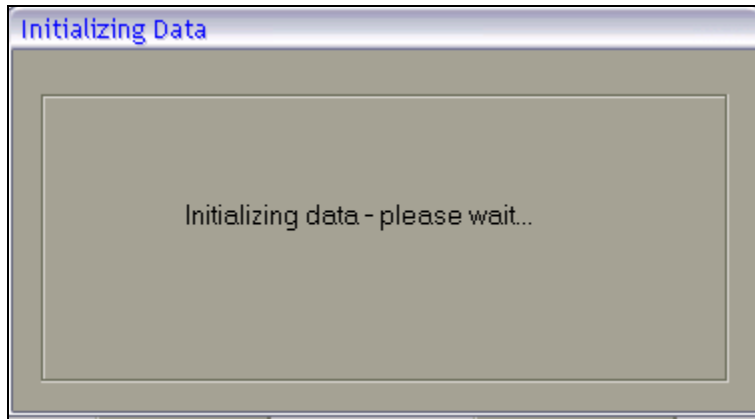


Figure 74 Initializing Data screen

10. Next, the License Machines screen appears (see Figure 75). From here you may assign which computers can or cannot access the license on your computer.

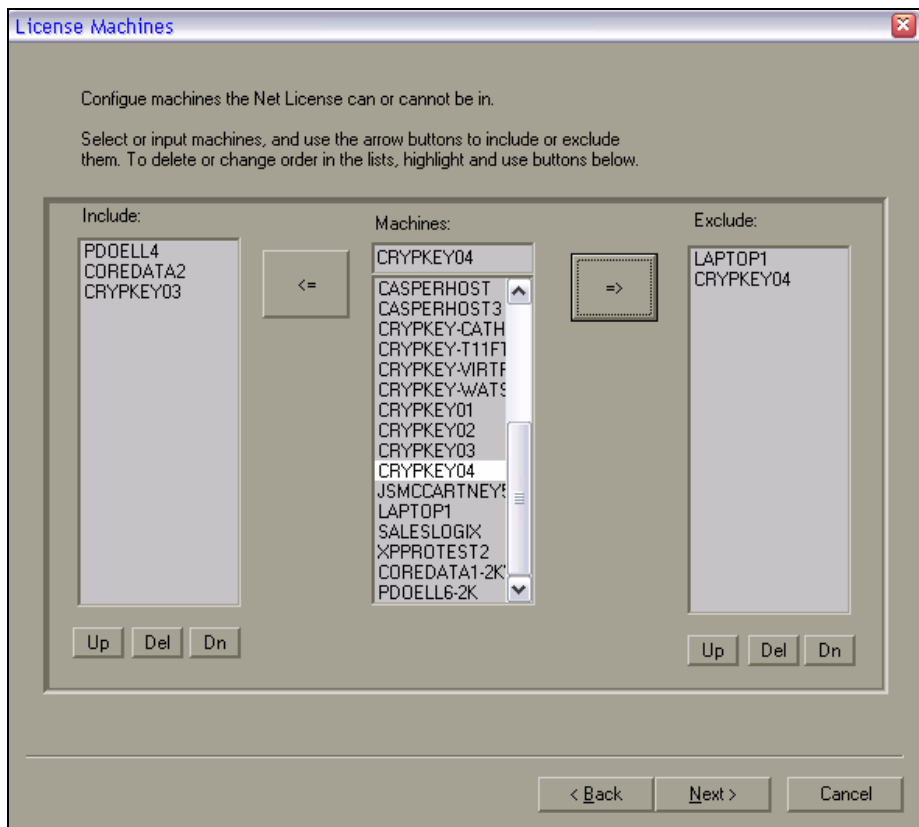


Figure 75 License Machines screen

11. Click **Next** to continue.
12. The next screen is the License Users screen. It performs the same functions as the previous screens (for domains and computers), but for users. Select which users you want to include and which you want to exclude.

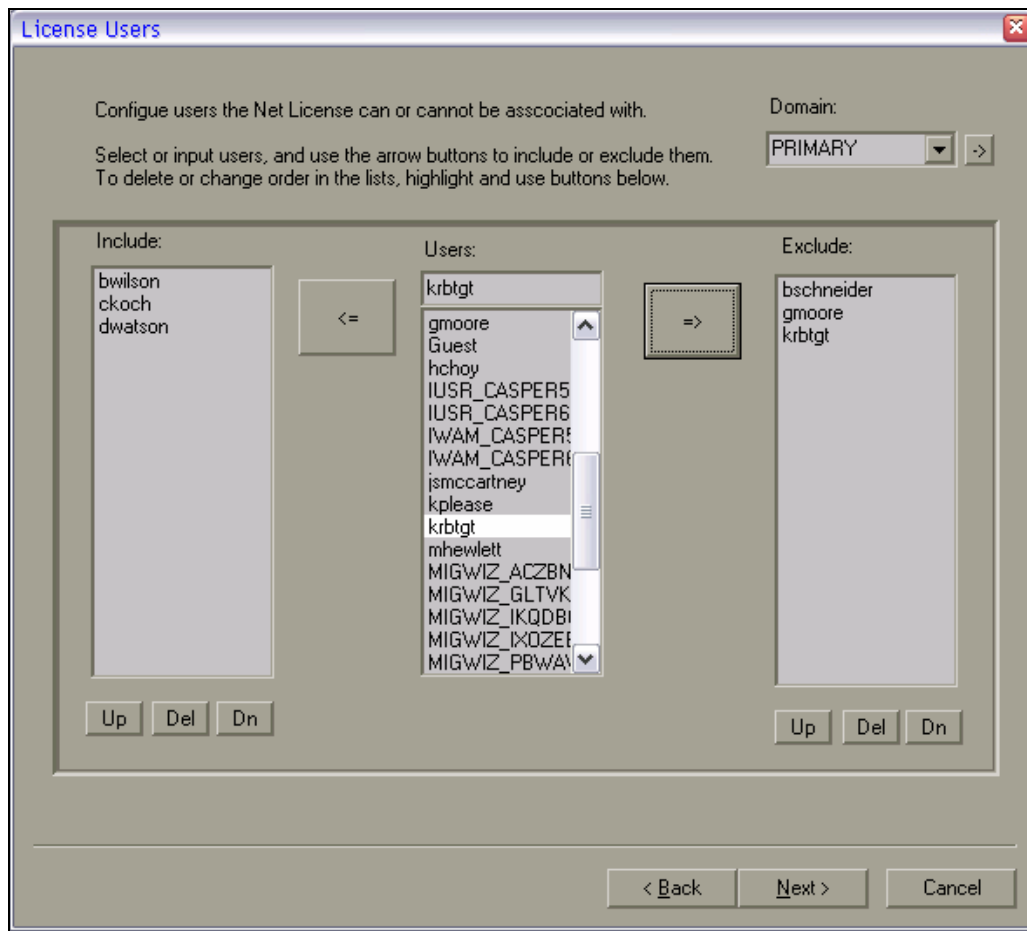


Figure 76 License Users screen

13. Click **Next** to continue.
14. On the final screen, you can select to activate the server process for the license (or you can do it later from the main NLM menu).

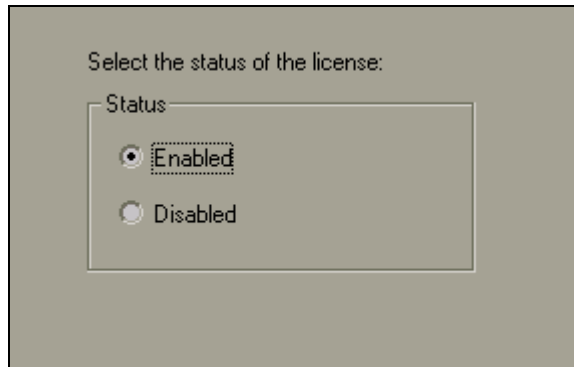


Figure 77 License Status Selection screen

15. Click **OK** to start serving the license to the network.

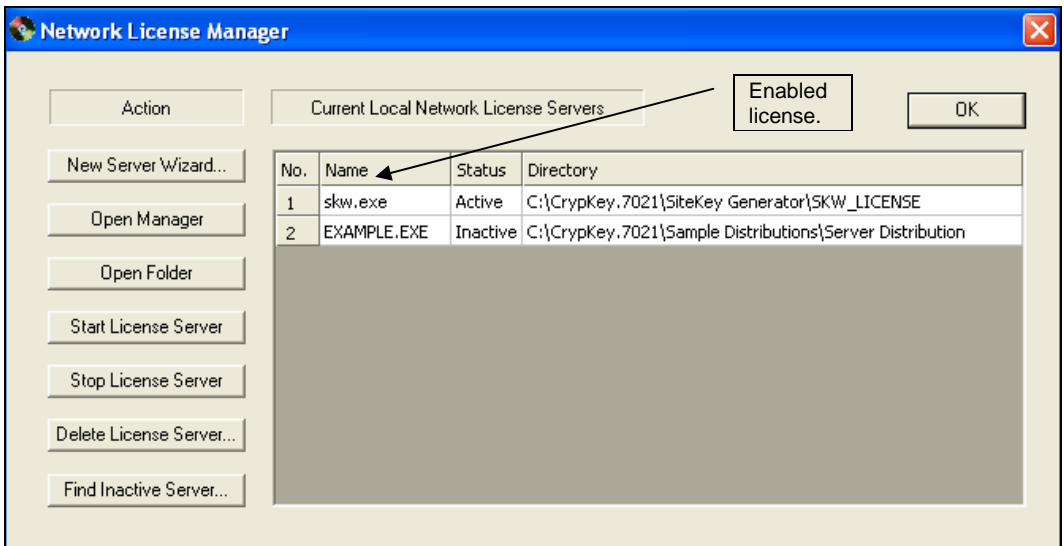


Figure 78 Network License Manager screen with enabled license

16. At this point, the server is running and clients can now access the license.

Open Manager

This function will launch the Enterprise License Manager (ELM). The ELM includes most of the functionality of the Network License Manager. The ELM can configure server licenses, license transfers, and monitor network license usage on its local machine.

To launch the Enterprise License Manager (ELM), click on a server in the list, then click on the **Open Manager** button. To use the ELM, please refer to the ELM User Manual included in the documentation set for this release.

Open Folder

When you select **Open Folder**, an Explorer window will open so you can search for the location of the selected license, for your file management purposes.

Start License Server

Start License Server will launch a process called `crp32002.ngn` that serves the license to clients on the network.

Stop License Server

Stop License Server will stop a license from being available to network clients, by closing down the related `ngn` process.

Delete License Server

Delete License Server will delete the configuration (`ces.ini`) file for the license.

Find Inactive Server

This function will open an Explorer window so that you can manually browse the computer for licenses that do not appear on the main list. See image below.

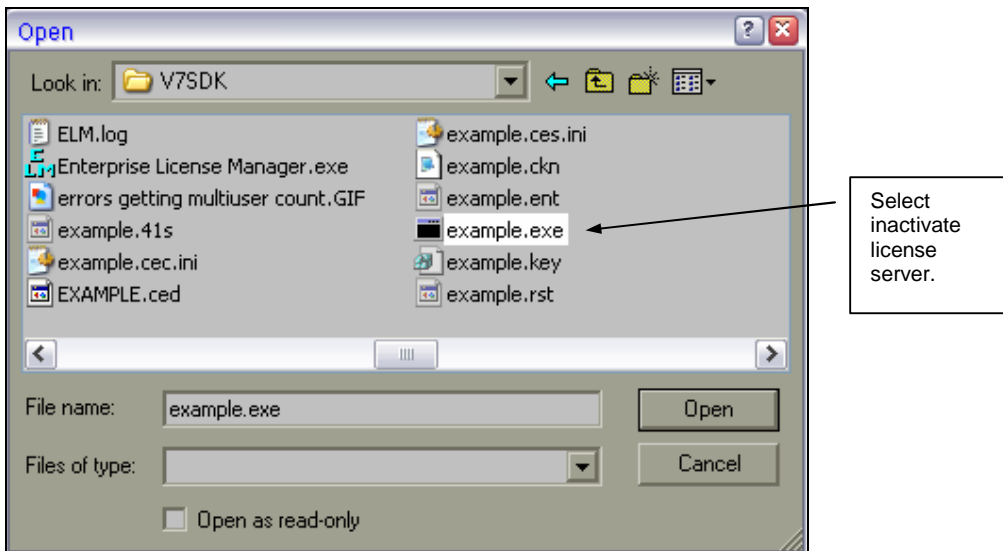


Figure 79 Browse for Inactive Servers

Once you select a license and click on **Open**, the server license will be activated:

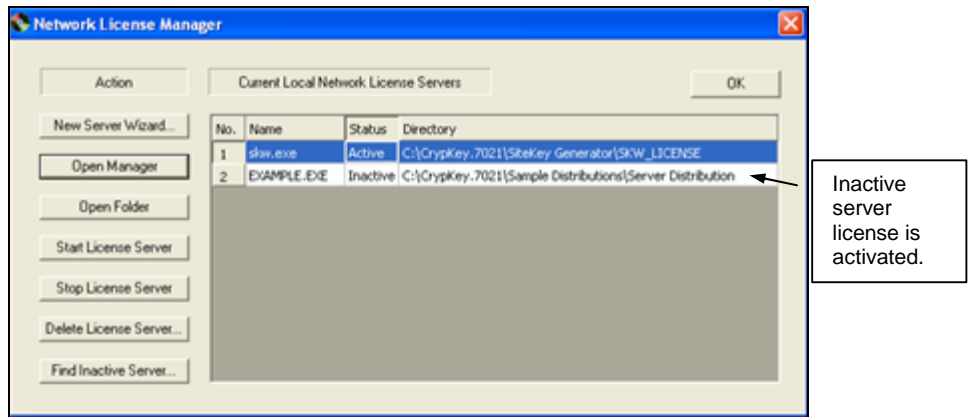


Figure 80 Find Inactive Server

Example configuration file

The *ces.ini* file contains the setting you configured for the license:

Example.ces.ini:

```
[GENERAL]
STATUS=Enabled
[INCLUDED_DOMAINS]
D1=ALL
[INCLUDED_USERS]
D1=ALL
[INCLUDED_MACHINES]
[EXCLUDED_DOMAINS]
[EXCLUDED_USERS]
[EXCLUDED_MACHINES]
```


Chapter 13: CLC

The CLC manages client license configuration on a network.

Client License Configuration

The Client License Configuration (CLC) is used to manage client license configurations. As the software vendor, you can provide your customers with this configuration tool for the purpose of defining where on the network the client application should look for a license.

The utility is installed in the application folder on the client machine and allows for the following types of configuration:

- a. Specify on which domains to look for a license;
- b. Specify on which machines to look for a license; and
- c. Enable a client application to look on the network for a license.

Typically, the installation of applications on a network, and the configuration of their licenses, is managed by the network administrator or equivalent. You can also provide a preconfigured *cec.ini* file to the client machines.

Using the Client License Configuration Utility



ClientLicenseConfig.exe

The prerequisites for creating a client license configuration are:

1. The server licenses have been configured and initialized beforehand. (For details on this process, please refer to *Chapter 11: ELM OR* Chapter 4: Getting Started.)
2. The first step is to start *ClientLicenseConfig.exe*. Double-click on the icon located in the license folder.
3. After you've started the *clientlicenseconfig.exe*, select the license file or existing configuration file from the browse screen. The configuration file will have the extension *cec.ini*. You no longer need to browse for the license using the CLC; when you start the CLC in the directory in which the application is located, it uses the *.ced* file in the directory.

Note: If the computer has the folder view options set to “hide known extensions”, then all you will see is the .cec extension.

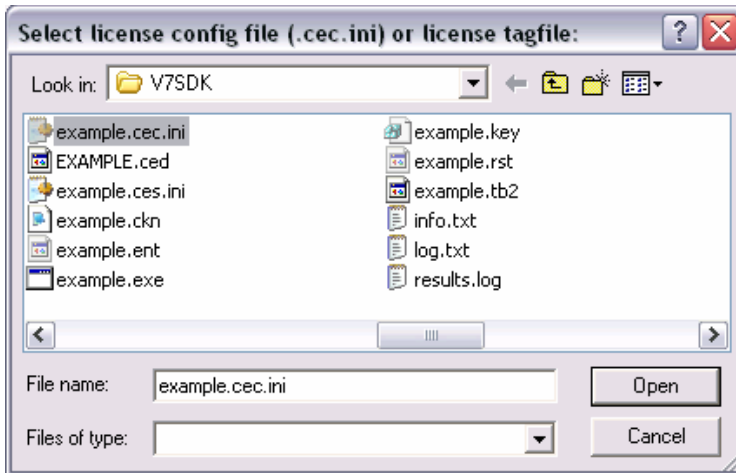


Figure 81 Select License Configuration file

4. Select the .cec configuration file for your application.
5. Select **Open**. The Client Properties screen appears, License Domains:

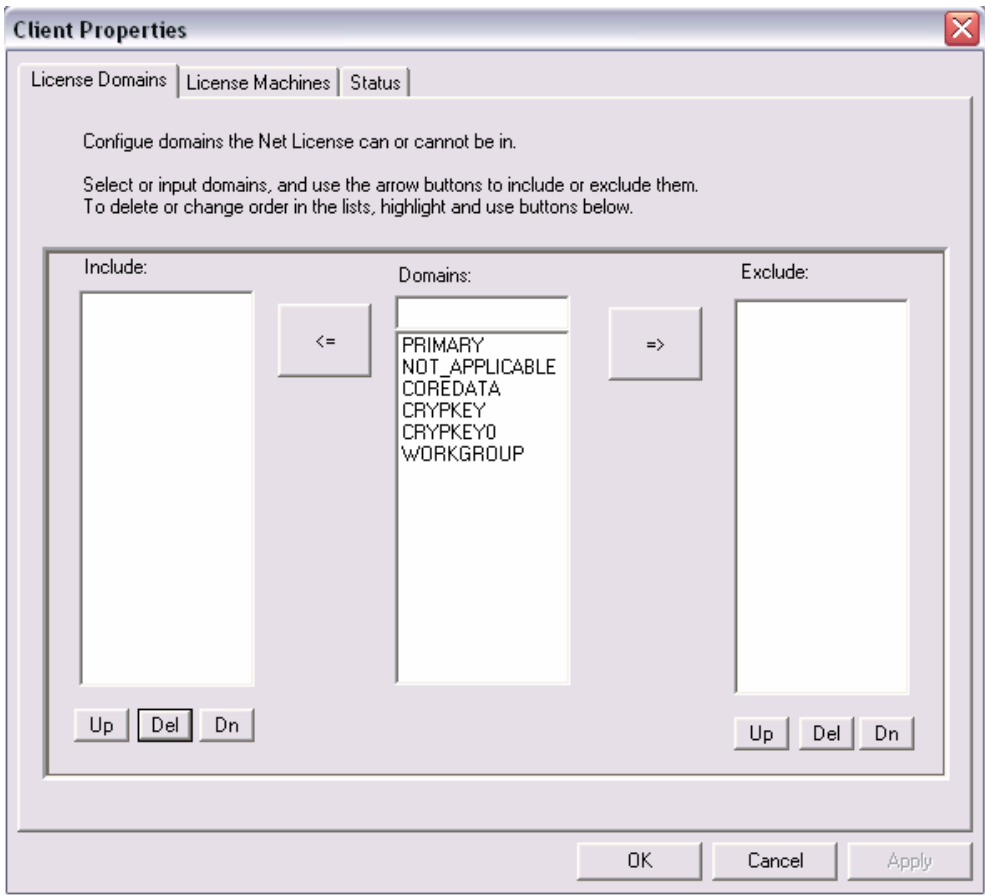


Figure 82 Client Properties screen

6. In the License Domains page, you can specify the domains to search. You can also control the order in which they will be searched by using the **Up** and **Dn** (down) buttons. Click these buttons to adjust the position of the domains in the search queue. The order of search is top to bottom in the list.

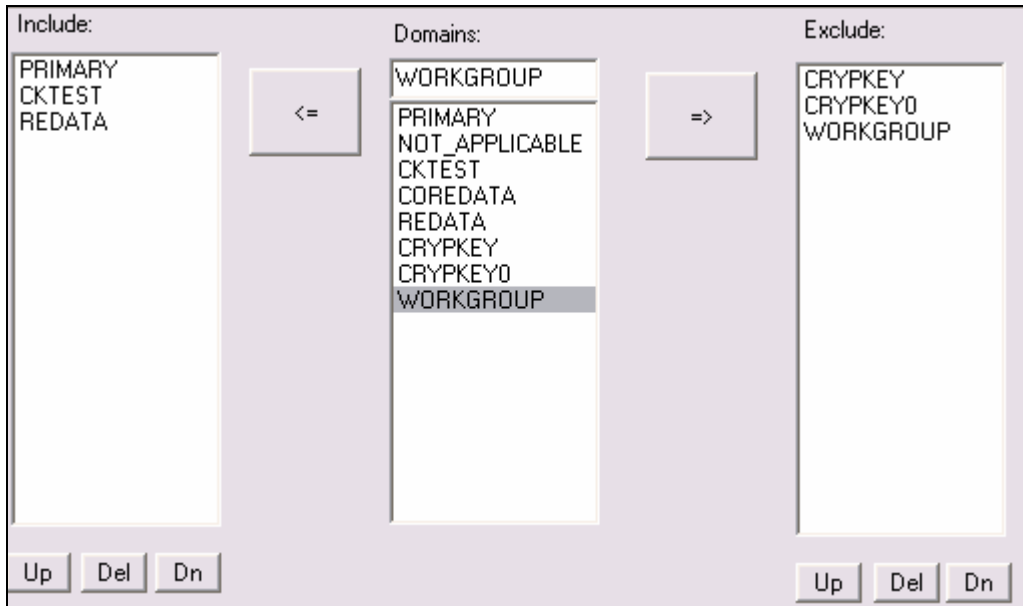


Figure 83 License Domains page

7. In this example, the Primary domain (the domain where the client computer is located) is searched first; if no license is found in the Primary domain, CKTEST and REDATA are searched, in that order.
8. Instead of, or in addition to searching domains, you may choose to specify the servers (License Machines) to search on for a license. If you are going to directly specify the servers (License Machines) where the license is located, select NOT_APPLICABLE. In this case NOT_APPLICABLE would be the ONLY entry in the include list.
9. You can also exclude domains from the search as shown in Figure 83.
10. Now click on the License Machines tab. In the License Machines page you can specify which computers to search for a license. While this screen populates, you will see a message that says "Initializing Data"; shortly after a list of machines will appear. This is a list of all the computers on the network, so it may take a while to compile.

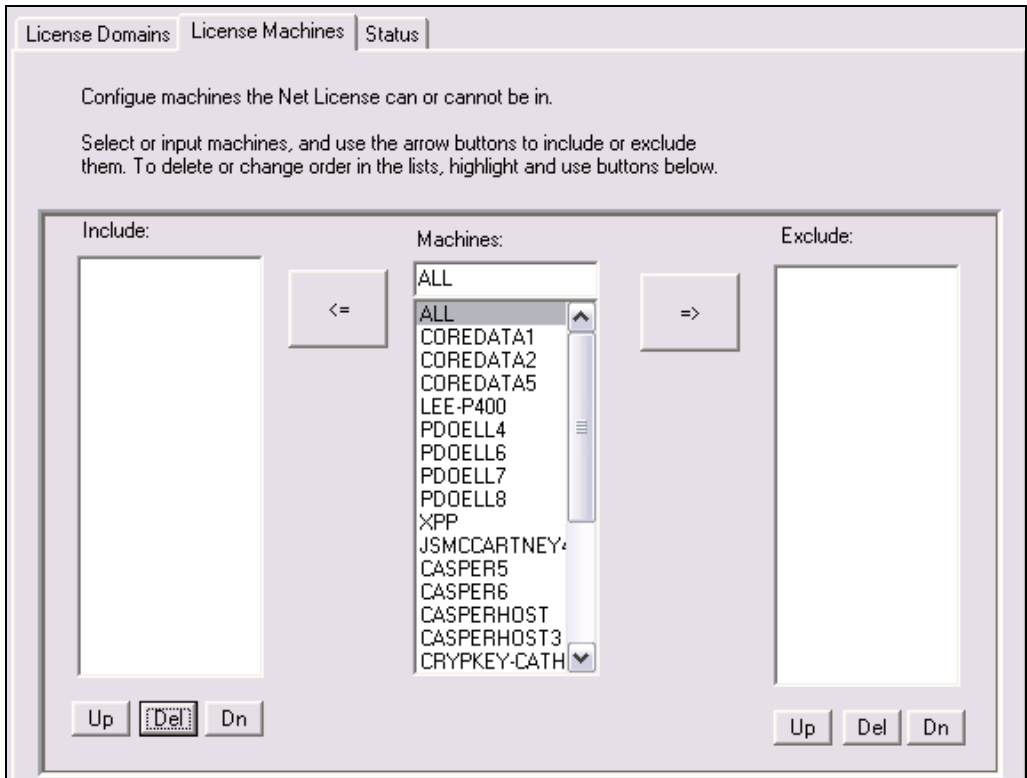
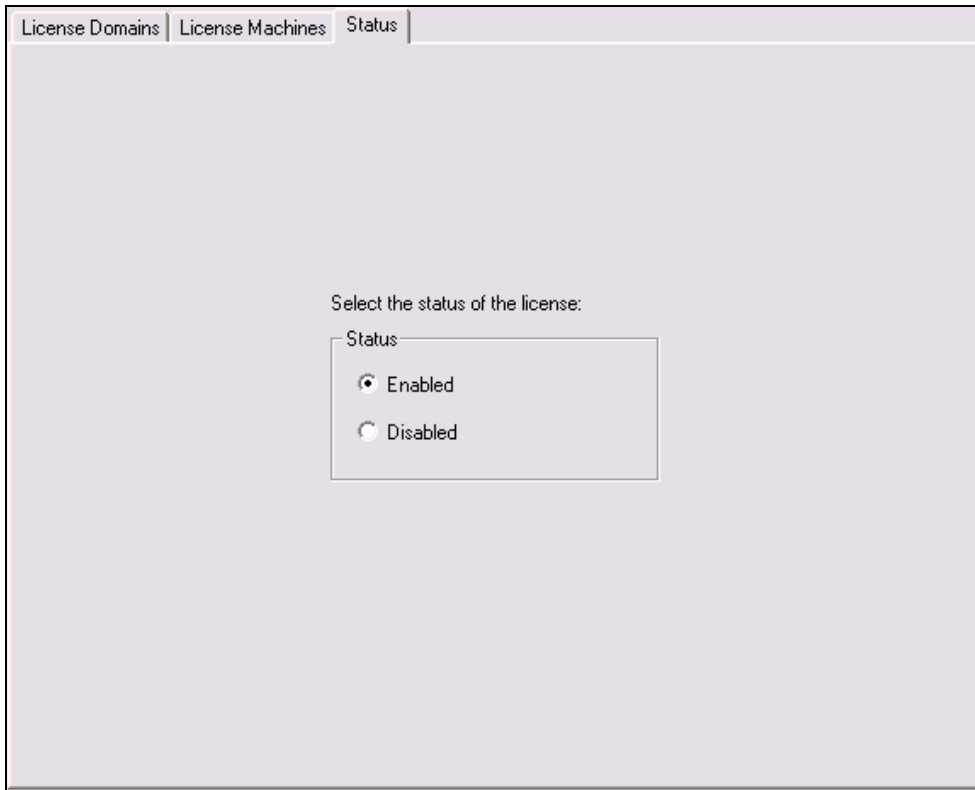


Figure 84 License Machines page

11. You may choose to search ALL the computers.

Note: If you have selected NOT_APPLICABLE in the License Domains page (meaning the search isn't filtered by domain), then the 'ALL' option is not available and you must add computers to the Include list individually. If you have selected some domains to search, and choose 'ALL', then all the computers in the selected domains will be searched. As in the License Domains page, you can also control the search order for the computers in the Include list by rearranging the order using the **Up** and **Dn** buttons.

12. Now click on the Status tab. The Status page allows you to activate the Client License Configuration (CLC) wizard to look for licenses on the network using the parameters you have defined.
13. Select **Enabled** and the client will be ready to look on the network for a license. Selecting **Disabled** allows the client to use a normal license, i.e a license installed on the local computer.



The screenshot shows a web interface with three tabs at the top: "License Domains", "License Machines", and "Status". The "Status" tab is selected. The main content area is light purple and contains the text "Select the status of the license:". Below this text is a form with a label "Status" and two radio button options: "Enabled" (which is selected) and "Disabled".

Figure 85 License Status page

Sample CEC.INI file

This is an example of what you could expect to see in the example.cec.ini file.

```
[INCLUDED_DOMAINS]
E1=ALL
[INCLUDED_MACHINES]
E1=ALL
[GENERAL]
STATUS=Enabled
```

Chapter 14: Function Reference

Certain CrypKey functions require background information and technical details.

This chapter contains reference information that supports the other chapters in this manual. It describes the functions and general error values used by CrypKey and provides assistance for programming your CrypKey code. For function declarations, refer to the sample code.

Note: If you encounter an undocumented error value for a specific function call, it is possible that CrypKey has updated the call, but has not had the opportunity to update the manual at that time. Please check *crypkey.h* from the latest version of the software for any undocumented errors. If you are still not able to find it, you can contact us at Support@Crypkey.com.

Function Summary

For a quick reference of function syntax, see *Appendix A: CrypKey Functions*.

Table 17: Function Summary

Name	Function
AcquireLicense()	Allows an unlicensed CrypKey-protected program to acquire a license from a licensed copy. It is the opposite of the DirectTransfer() function.
CKChallenge32()	Used to authenticate the CrypKey DLL, ensuring that it is not an impostor.
CKTimeString()	Translates, into a text string, the ULONG time values returned by the functions GetAuthorization() and FloatingLicenseSnapshot().

Name	Function
CrypkeyVersion()	Returns the version number of the CrypKey engine (>NGN) currently in use. The function name in Visual Basic is crypkeyVersion.
DirectTransfer()	Called from an application that has authorization, this function transfers the application's authorization to another directory.
EndCrypkey()	Notifies CrypKey that the program is terminating. Ensure this function is called if you are using networked licenses.
ExplainErr()	Returns a text string that explains the return code of various functions.
ExplainErr2()	Returns a text string (written into string text) that explains the return code of various functions.
FloatingLicenseGetRecord()	C#.NET function only: returns a single record of the current floating license status. The function FloatingLicenseTakeSnapshot() needs to be called before this function.
FloatingLicenseSnapshot()	Returns a snapshot of all users holding or waiting for a network-floating license.
FloatingLicenseTakeSnapshot()	C#.NET function only: returns a snapshot of all users holding or waiting for a network floating license.
Get1RestInfo()	Returns authorization options, the number allowed, and the number used.
GetAuthorization()	Gets the level at which the program is authorized to run or an authorization failure code. Also used to decrement the number of uses count.
GetAuthorization2()	The same as GetAuthorization(), except this function decrements the number of uses count only.

Name	Function
GetCustomInfoBytes	The function gets the data that was written to a file by the SetCustomInfoBytes() function.
GetDrivePermanentSerialData()	Proprietary CrypKey software function that accesses the following hard drive information on a customer's computer: model number, serial number, firmware data.
GetLastError()	Returns the last error text from the last call to the CrypKey library.
GetLastErrorCode()	C#.NET function only: Returns the error code from the last call to the CrypKey library. Some functions in the CrypKeyNet.dll assembly may not provide error codes directly back to the caller; in this case, you can use this function. You could also use this function as a convenient and a common way to check the result of all calls to the CrypKey library.
GetLevel()	This function is used to determine the Level in which the protected program is authorized to run.
GetLibraryType()	This function returns a number that indicates the type of CrypKey client library currently in use.
GetLibraryVersion()	This function returns the version number of the CrypKey client library currently in use.
GetLicenseInfo()	This function returns information about the current operating license mode and location.
GetNetHandle()	Gets the network license handle of the protected program.
GetNumCopies()	Gets the number of license copies that the current site has been granted.

Name	Function
GetNumMultiUsers()	Gets the number of customers to whom the site license has been granted.
GetOption()	For a specific option, the function returns 1 if the option is on, 0 if the option is off, and -1 if the program is not authorized.
GetPCSerial()	This function returns the computer bios serial data.
GetRestrictionInfo()	Gets information on any restrictions that are present in the license.
GetSiteCode()	Gets the site code for the program installation location (site).
GetSiteCode2()	Returns a pointer to the site code. The same as GetSiteCode(), except text is not written to a string.
InitCrypkey()	Initializes CrypKey with the required runtime information.
IsEasyLicense()	This function indicates if the current license is an Easy License
KillLicense()	Kills the protected program's license and provides a confirmation code.
ReadyToTry()	Sets up a days class, one-time automatic license without version control. The function name is readyToTry in Visual Basic.
ReadyToTryDays()	Sets up a days class, one-time automatic license. The function name is readyToTryDays in Visual Basic.
ReadyToTryRuns()	Sets up a runs class, one-time automatic license. The function name is readyToTryRuns in Visual Basic.

Name	Function
RegisterTransfer()	Called from an application that does not have authorization, this function registers the application by placing special files on a disk or directory.
SetCustomInfoBytes()	This function sets data to a file that can only be read by the GetCustomInfoBytes() function.
SaveSiteKey()	Used to save the site key that is acquired by the customer. The key is checked before it is saved.
SetNetHandle()	Sets the network license handle reference of the protected program.
TransferIn()	Called from an application that does not have authorization, this imports an authorization license by reading special files from a disk or in a directory that were created by the TransferOut() function.
TransferOut()	Called from an application that has authorization, this exports the application's authorization license by writing special files out to a directory where a registration file was created using the RegisterTransfer() function. Only the registered application can successfully call the TransferIn() function to import this license.

General Error Values

Most CrypKey functions that return values can return general error codes in addition to function-specific error codes (described in the *Function Descriptions* following). These general errors include network and DLL communication errors. You should always check for these return values and give some explanation to the customer if they occur — ExplainErr() (see ExplainErr() on page 182) returns an appropriate message.

Note: As mentioned in Table 18, errors in the –200 to –299 range result from security checks. The most common cause of these errors is the placement of the .ngn file in a different directory from the one where the executable resides. In these cases, moving the .ngn or executable to the proper location should clear the error. If it does not, please contact CrypKey technical support at Support@Crypkey.com.

Table 18: General Errors and Solutions

Name	Value	Message	Solution
FILE_INFO_FAIL	-100	Could not get the licensing information for unknown reason (wrong operating system or insufficient memory)	If you see this error, ensure you have the most recent software. Send an email message to Support@Crypkey.com containing the dates of the CrypKey files you are using and we will tell you if you have the most recent software.
NETWORK_DISCONNECTED	-101	The network connection appears to have been lost	If you encounter this error, ensure you are using the most recent driver. Do this by sending an email message to support@crypkey.com that states the date of the driver you are using.
NETWORK_NO_CKSERVE	-102	CrypKey License service installation has failed or the service is not running.	The CrypKey license service must be installed and running in order for a protected application to run. In order to install the CrypKey license service, you must be Administrator. If you are running in normal mode or drive share mode,

Name	Value	Message	Solution
			<p>the local everyone group and the local user must have full control of the application directory. If the application is running in client/server mode, the service must be running on the server and the clients must be able to see the drive. Please also ensure that the CrypKey license service is not disabled.</p> <p>If none of the above resolves the error, Please contact Support@Crypkey.com.</p>
NETWORK_BAD_REPLY	-103	The reply file was either damaged or tampered with	<p>This error is usually caused by a conflict with an old driver. Contact Support@Crypkey.com for more information.</p>
INIT_NOT_SUCCEEDED	-104	CrypKey initialization did not succeed	Check the return value from InitCrypKey. This should provide you with information about the problem.
NO_MEMORY	-107	A request for memory failed	Email us at support@crypkey.com if this error occurs.
NETWORK_BAD_CRC	-110	A reply to a network request is corrupted	Email us at support@crypkey.com if this error occurs.
INIT_NOT_CALLED	-111	All functions require an InitCrypKey() function to be called and	Ensure InitCrypkey() is called before calling any

Name	Value	Message	Solution
		returned successfully in order to run.	other functions.
EASYLIC_OPERATION_NOT_ALLOWED	-112	Certain operations such as license transfers are not allowed with EasyLicence (see <i>Chapter 6: SKG, EasyLicence</i>).	Instruct the user that this operation is not allowed.
REMOVEABLE_DRIVE_NOT_ALLOWED	-113	CrypKey licenses are based off of the hard drive. We can not write a license to a removeable drive unless CrypKey USBKey licensing is being used.	Regular CrypKey licensing in normal, drive share or client/server mode, key off of a users hard drive. The CrypKey license can not be written to a removable drive. The only exception to this is if you are using the CrypKey USBKey licensing.
NETWORK_COM_ERR	-114	Could not communicate with a network target	Check network connections (and drive shares if appropriate)
SERVER_APP_MUST_START_LOCALLY_ERR	-115	"This application is configured as a Server and can only be started locally."	"Go to the computer the application is on to start it, or configure it to not be a Server."
CLIENT_OPERATION_DISALLOWED	-117	This operation is not allowed when operating as a client	Try again with a program that is configured and licensed to run locally or as a server

Name	Value	Message	Solution
[various messages]	-200 to -299	The .ngn file is possibly not in the same directory as the executable.	The 200-series errors are generated by security checks. Ensure that the .ngn file is in the same directory as the executable. If this action does not clear the error, email us at Support@Crypkey.com

Function Descriptions

This section provides detailed information about the purpose, use, syntax, parameters, and interaction of CrypKey functions.

You can also see a summary of these functions in *Function Summary* on page 163, and in *Appendix A: Quick Reference*.

Where applicable, each description includes a prototypes table, showing the following:

- C prototype, for use with the CrypKey Library
- C++ prototype, for use with the CrypKey COM object
- Visual Basic (VB) prototype, for use with the CrypKey COM object
- C#.NET prototype, for use with the CrypKey .NET assembly.

Note that some of the C#.NET methods are unsafe code because they pass values by reference. These functions are provided for compatibility only. Alternate methods exist that contain safe code and are in many cases provided in this document.

AcquireLicense()

Description:

This function will transfer a license from a different location to the calling app's license location. This function is very similar to the `DirectTransfer()`. The difference being, instead of sending a license to a new directory, you obtain the license from an already licensed directory. It is essentially the opposite of the `DirectTransfer()` function.

Function Usage:

Use this function whenever you need to transfer a license from one location to the application's license location, providing the application has a full access to both license locations.

For this function to be successful, certain conditions must be met:

- a. the target directory (directory which will obtain the license) must not already have a valid license;
- b. the source (directory which will transfer the license to the target directory) must have at least one valid license; and
- c. if either source or target directory is not local, a CrypKey license service must be operation on the machine the directory is on.

If the source license is a fixed license and has more than one copy, this function transfers exactly one copy to the target and decrement the source number of copies by one.

If the source license is a floating license with x users allowed, the entire license including x users is transferred.

Note: If the source directory is not local, the source license does not have to be a floating license in order to be transferred. As well, if you would like to transfer only one network seat out of a pool you would want to use the ELM transfer functionality.

PROTOTYPES

Language	Syntax
C	int AcquireLicense (char *licensePath);
C++	nt AcquireLicense (char *licensePath)
VB	Declare Function AquireLicense& Lib "CRP32DLL">DLL (ByVal directory\$)
C#.NET	CKAcquireErrorEnum AcquireLicense (String *pstrDirectory);

PARAMETER

The license path is a pointer to a NULL terminated string that gives the directory from which the license is to be obtained.

RETURN VALUE

If the function operates normally, it returns the value 0. Otherwise, it returns one of the error values described in the table below.

Name	Value	Message	Solution
AL_SOURCE_NOT_AUTHORIZED	-1	Source application must be authorized for this operation	Ensure the source application has a valid license.
AL_TARGET_APP_NOT_FOUND	-2	Target application was not found	Ensure there is the copy of the application in the target directory.
AL_FLOPPY_NOT_ALLOWED	-3	Operation not allowed on portable media	Do not acquire licenses from or to portable media
AL_SITEKEYFILE_NOT_FOUND	-4	Source site key file was not found	The license.key files was not found at the source location.
AL_RSTKEYFILE_NOT_FOUND	-5	Source .RST file was not found	The license.rst files was not found at the source location.
AL_COULD_NOT_WRITE_SITEKEYFILE	-6	Could not write to target site key file	The application does not have full control of the license.key file.
AL_TARGET_WRITE_PROTECTED	-7	Target site he filed his write protected	The application does not have full control of the target directory.
AL_SOURCE_SAME_AS_TARGET	-8	Source directory is the same has target directory	Ensure the source and target directories are different.
AL_THIS_IS_AUTHORIZED	-9	Target already has authorization	The target location already has a license.
AL_LICENSE_SOURCE	-10	Source of application was not	The source directory does not have a

Name	Value	Message	Solution
RCE_NOT_FOUND		found	license.
AL_COULD_NOT_INITIALIZE_SOURCE	-11	Could not initialize CrypKey in source directory	CrypKey could not initialize for one of many reason. Please contact Support@Crypkey.com for more information.

CKChallenge32()

This function is used to verify that the CrypKey DLL has not been replaced with an imposter DLL. The basic premise of the verification is that you know your 'company' number and 'password' number (CrypKey assigns these to you when we request your developer keys) and the DLL will know your 'company' number and 'password' number. These numbers are referred to as 'the secrets'. If you do a calculation with these numbers and two other numbers chosen at random and then call the DLL to do the same calculation with the same four numbers, it should come up with the same answer. Only the real DLL can decode the Master Key and User Key to get your 'company' number and 'password' number for the calculation, so an imposter DLL will have great difficulty providing the correct answer for any possible values of the two random numbers.

FUNCTION USAGE

Call this function when the program first starts to ensure the authenticity of the DLL. If the application does not have a valid license, CKChallenge32() will not return the correct codes.

PROTOTYPES

Language	Syntax
C	<code>long CKChallenge32 (long random1, long random2);</code>
C++	<code>long CKChallenge32 (long nRandom1, long nRandom2);</code>
VB	Declare Function CKChallenge32& Lib "CRP32DLL.DLL" (ByVal random1&, ByVal random2&)

Language	Syntax
C#.NET	int CKChallenge32 (int nRandom1, int nRandom2);

PARAMETERS

Random1

Random1 is a 32-bit number chosen at random. A good way to get a random number is to use the system time.

Random2

Random2 is a 32-bit number chosen at random. A good way to get a random number is to use the system time..

Return Value

This function returns the result of the challenge calculation done using the two parameters of this function. The actual calculation in C' code would be as follows:

```
long challenge32(long companyNum, //companyNum and passNum are
assigned to you by CrypKey
long passNumDiv2, //divide PassNum by 2 to get PassNumDiv2

        long random1,

        long random2)

{
    int i;

    long ret;
    ret = 1;

    for(i=2;i<11;i++)
    {
        ret = ret%32769 * ( (random1/i)%32769 + (companyNum/i)%32769);
        ret = ret%32769 * ( (random2/i)%32769 + (passNumDiv2/i)%32769);
    }
}
```

```
return(ret);
```

```
{
```

where:

- ret is the result of the calculation.
- random1 and random2 are two 32-bit numbers chosen at random.
- companyNum and passNum are your CrypKey-assigned secret numbers.

The actual calculation that you would do in C code is also shown in *example.c* in the Sample directory.

Notes

The InitCrypkey() and GetAuthorization() functions must be called and returned successfully for CKChallenge32() to work. Note that CKChallenge32() does not return the correct code if the program is not authorized — this gives you another way (besides GetAuthorization()) to check for a valid license.

CrypKeyBuildNum()

DESCRIPTION

This function returns the build number of the CrypKey system engine currently in use.

FUNCTION USAGE

Call this function whenever you need to know which build of the CrypKey engine (.ngn) is being used by your program.

DECLARATION (C/C++)

```
int CrypkeyBuildNum(void);
```

DECLARATION (VISUAL BASIC)

```
Declare Function CrypkeyBuildNum& Lib "CRP32DLL.DLL" ()
```

PARAMETERS

None.

RETURN VALUE

This function returns a number indicating which build of the CrypKey engine (.ngn) is in use.

CKTimeString()

The functions `GetAuthorization()` and `FloatingLicenseSnapshot()` both return `CK_time_t` values that represent time. This function translates the time into a text string.

See *example.c* for an example of how to use this function.

PROTOTYPE

```
char * CKTimeString(char *timestringbuf, CK_time_t cktime)
```

PARAMETERS

timestringbuf

timestringbuf is a buffer to put the time string at least `CK_MAX_TIMESTRING_LEN` characters wide. The time string is in the form `YY-MM-DD HH:MM:SS`.

t

t is the 32-bit integer value received from the CrypKey function to be translated to text. It returns a pointer to *timestringbuf* for convenience.

RETURN VALUE

This function retruns the appropriate error message.

Note: The pointer returned is to a buffer internal to the CrypKey library. Subsequent calles to `CKTimeString()` will overwrite the contents of the buffer.

CrypkeyVersion()

DESCRIPTION

This function returns the version number of the CrypKey engine (.ngn) is currently in use.

FUNCTION USAGE

Call this function whenever you need to know which version of CrypKey is engine (.ngn) is being used by your program code.

CrypkeyVersion() returns the version number of the CrypKey system currently in use.

PARAMETERS

None.

PROTOTYPES

Language	Syntax
C	<code>int CrypkeyVersion ();</code>
C++	<code>int CrypkeyVersion();</code>
VB	Declare Function CrypkeyVersion& Lib "CRP32DLL.DLL
C#.NET	<code>int CrypKeyVersion ();</code>

RETURN VALUE

CrypkeyVersion() returns a two-digit number that indicates which version of the CrypKey library is in use (65 indicates 6.5, 70 indicates 7.0, etc.).

DirectTransfer()

Called from an application that has authorization, this function will transfer the application's authorization to another directory. This transfer could be from one directory to another on the same computer or from one network drive to another.

The directory where the function is initiated must also be the directory where the license resides.

Note: The InitCrypKey() and GetAuthorization() functions must be successfully called before DirectTransfer() is used.

FUNCTION USAGE

Use this function whenever you need to transfer a license from one location to another and the application has local or network read-write access to both license locations.

PROTOTYPES

Language Syntax

C `int DirectTransfer (char *directory);`

C++ `int DirectTransfer(char *directory);`

VB `Declare Function DirectTransfer& Lib`
 `"CRP32DLL.DLL" (ByVal directory$)`

C#.NET `CKDirectErrorEnum DirectTransfer (String`
 `*pstrDirectory);`

PARAMETER

Directory

Directory is a pointer to a null-terminated text string that contains the path of the directory to send the license to.

RETURN VALUE

Table 19: DirectTransfer() Values

Name	Value	Message	Solution
DT_OK	0	The function completed successfully	
DT_THIS_NOT_AUTHORIZED	-1	This site is not authorized; there is nothing to transfer	If there is no authorization, you cannot transfer a license. Obtain authorization before attempting to transfer your license.

Name	Value	Message	Solution
DT_TARGET_APP_NOT_FOUND	-2	A copy of the program must be in the target directory	A copy of the program must be located in the target directory. If it is not there, you cannot transfer the license.
DT_FLOPPY_NOT_ALLOWED	-3	It is not possible to transfer a license to portable media.	Running a DirectTransfer() requires you to transfer a license to a physical hard drive. The DirectTransfer() cannot be used to transfer a license to portable media.
DT_SITEKEYFILE_NOT_FOUND	-4	The key file could not be opened.	This function must have access to the key file. Ensure that full control has been granted to the target and the source directories for the DirectTransfer() to be successful.
DT_RSTKEYFILE_NOT_FOUND	-5	The restriction file could not be opened.	This function must have access to the restriction file. Ensure that full control has been granted to the target and the source directories for the DirectTransfer() to be successful.
DT_COULDNOT_WRITE_SITEKEYFILE	-6	The target directory may be write protected.	This function must read and write to specific files in your directory. You must grant full control to the directory in order to have this

Name	Value	Message	Solution
			function run successfully.
DT_SOURCE_WRITE_PROTECTED	-7	The directory of the source license for the transfer is write protected.	This function must have full control of the source and target directories.
DT_SOURCE_SAME_AS_TARGET	-8	The source directory is also being used as the target directory.	To use this function you must be transferring from one directory to another. You cannot transfer a license from a directory into the same directory.
DT_TARGET_HAS_LICENSE	-9	To use the DirectTransfer(), the location that you are transferring to (target) can not already have a valid license	Kill the license in the target directory before attempting to perform a DirectTransfer()

General error values also apply — see the section *General Error Values* on page 167.

Notes

The DirectTransfer() function will overwrite an existing license at the target location without warning. Therefore, always check whether or not this is a desirable result.

EndCrypkey()

EndCrypkey() notifies CrypKey that the protected program is being terminated. This allows CrypKey to ensure that its state of operation is released properly beforehand.

Function Usage

Make sure that this function is called before the protected program is going to terminate. A good place to do this is in the application object destructor (if you are writing in C++).

Prototypes

Language	Syntax
C	<code>void EndCrypkey();</code>
C++	<code>void EndCrypKey();</code>
VB	<code>Declare Function EndCrypkey Lib "CRP32DLL.DLL" ()</code>
C#.NET	<code>void EndCrypKey();</code>

ExplainErr()

This function returns a string explaining the return codes of CrypKey functions.

FUNCTION USAGE

It is not mandatory that the developer show these particular explanations to the user, but it is recommended that some explanation be given when a CrypKey error does occur.

Note: We recommend that at minimum the return code and the function call the error is returned from is retrieved in order to be able to troubleshoot the problem efficiently.

PROTOTYPES

Language	Syntax
C	<code>char *ExplainErr (int functioncode, int errcode);</code>
C++	<code>char *ExplainErr (int functioncode, int errcode);</code>

VB	This function will not work properly under Visual Basic. Use ExplainErr2() instead
C#.NET	String *ExplainErr (CKExplainEnum nFunctionCode, int nErrorCode);

PARAMETERS

functioncode

Must be one of the function codes from the table below which are defined in cryptkey.h. This determines the function being explained.

errcode

Errcode is the specific error code returned by one of the SLAPI functions. The functions are listed in Table 20.

Table 20: ExplainErr() Functioncode Values

Name	Value	Function
EXP_AUTH_ERR	1	GetAuthorization()
EXP_GET_SITECODE_ERR	2	GetSizeCode()
EXP_SAVE_SITEKEY_ERR	3	SaveSiteKey()
EXP_REG_ERR	4	RegisterTransfer()
EXP_TO_ERR	5	TransferOut()
EXP_TI_ERR	6	TransferIn()
EXP_DT_ERR	7	DirectTransfer()
EXP_INIT_ERR	8	InitCrypkey()
EXP_RTT_ERR	9	ReadyToTry(),...Days(), ...Runs()
EXP_KL_ERR	10	KillLicense()
EXP_AL_ERR	11	AquireLicense()

Name	Value	Function
EXP_HDSN_ERR	12	GetDrivePermanentSerialID ata()
EXP_FLS_ERR	13	FloatingLicenseSnapshot()
GET_PHYSICAL_ERR	15	GetPCSerial()

RETURN VALUE

ExplainErr() returns the appropriate error message.

Note: The pointer returned is to a buffer internal to the CrypKey library. Subsequent calls to ExplainErr() or ExplainErr2() will overwrite the contents of the buffer.

ExplainErr2()

This function fills a buffer with a string explaining the return codes of various other SLAPI functions. This function is an alternative to using [ExplainErr\(\)](#), but the error message is written into your string text instead of being returned as a string.

FUNCTION USAGE

It is not mandatory that the developer show these particular explanations to the user, but it is recommended that some explanation be given when a CrypKey error does occur.

Note: We recommend that at minimum the return code and the function call the error is returned from is retrieved in order to be able to troubleshoot the problem efficiently.

PROTOTYPES

Language	Syntax
C	<code>void ExplainErr2(int functioncode, int errcode, char *text)</code>
C++	<code>void ExplainErr2(int functioncode, int errcode, char *text);</code>

VB	Declare Function ExplainErr2 Lib "CRP32DLL.DLL" (ByVal functioncode&, ByVal errcode&, ByRef text\$)
C#.NET	String *ExplainErr2 (CKExplainEnum nFunctionCode, int nErrorCode);

PARAMETERS

Functioncode

Must be one of the function codes from the table below which are defined in cryptkey.h. This determines the function being explained.

Errcode

This is the specific error code returned by one of the SLAPI functions noted within the function code table below.

Text

This is a pointer to the buffer where the error message is written to. It must be 80 characters in size

Table 21: ExplainErr(2) Functioncode Values

Name	Value	Function
EXP_AUTH_ERR	1	GetAuthorization()
EXP_GET_SITECODE_ERR	2	GetSizeCode()
EXP_SAVE_SITEKEY_ERR	3	SaveSiteKey()
EXP_REG_ERR	4	RegisterTransfer()
EXP_TO_ERR	5	TransferOut()
EXP_TI_ERR	6	TransferIn()
EXP_DT_ERR	7	DirectTransfer()
EXP_INIT_ERR	8	InitCrypkey()

Name	Value	Function
EXP_RTT_ERR	9	ReadyToTry(), ...Days(), ...Runs()
EXP_KL_ERR	10	KillLicense()
EXP_AL_ERR		11
AcquireLicense()		
EXP_HDSN_ERR	12	GetDrivePermanentSerialData()
EXP_FLS_ERR	13	FloatingLicenseSnapshot()
EXP_FLS_ERR	13	FloatingLicenseSnapshot()
EXP_PCSerial_ERR	15	GetPCSerial()

FloatingLicenseGetNumEntries()

This function takes a snapshot of the floating licenses. Returns the number of records in the ref int parameter.

PROTOTYPES

Language	Syntax
C#.NET	<pre>public int FloatingLicenseTakeSnapshot (ref int nEntries)</pre>

Notes

This function operates only in COM Objects and C#.NET Assemblies.

FloatingLicenseGetRecord()

This function is used only in COM Objects and C#.NET Assemblies. Called from an application that has authorization, the function returns a single record of the current floating license status. The function FloatingLicenseTakeSnapshot (), used only in COM Objects and C#.NET Assemblies, must be called before this function.

PROTOTYPES

Language	Syntax
C	[n/a]
C++	long FloatingLicenseGetRecord (long nRecord, lCrypKeySDKFloatRecord **ppRecord, long *pnResult);
VB	[n/a]
C#.NET	int FloatingLicenseGetRecord (int nRecord, CrypKeyFloatRecord *pRecord);

Notes

This function operates only in COM Objects and C#.NET Assemblies.

FloatingLicenseSnapshot()

The FloatingLicenseSnapshot() function gives a snapshot of all users holding or waiting for a network floating license. The function fills the buffer with entries, one for each user accessing the license.

Each entry contains the computer name, user login name, local time at which the user started using or waiting for a license, and the user's queue status (0 means they have a license, otherwise it is an integer that indicates the order in which each entry gets a license). This is the pertinent information given; the rest is for internal use.

PROTOTYPES

Language	Syntax
C	int FloatingLicenseSnapshot (unsigned long bufSize, int *numEntries, FLS_REC *buf);;
C++	int FloatingLicenseSnapshot (unsigned long bufSize, int *numEntries, FLS_REC *buf);
VB	Not Supported in VB

PARAMETERS

Bufsize

Bufsize is the size of the buffer allocated to receive the information.

NumEntries

NumEntries is a pointer to an integer-sized buffer that receives the number of entries in the table.

Buf

Buf is a pointer to a buffer of at least *bufsize* bytes in size and receives one entry for each user that is currently accessing, or attempting to access, the license.

Return Value

Table 22: Floating License Snapshot – Return Values

Name	Value	Meaning	Solution
FLS_OK	0	The function completed successfully.	
FLS_NO_NETWORK_LICENSE	-1	Application does not have a floating license.	Ensure you have a floating license before using this function.
FLS_NETWORK_TABLE_NOT_FOUND	-2	Network table file (.tb2) could not be found.	Contact CrypKey technical support if you get this error.
FLS_COULDNOT_OPEN_USERTABLE_FILE	-3	Network table file (.tb2) could not be opened.	Contact CrypKey technical support if you get this error.
FLS_BUFFER_TOO_SMALL	-4	Buffer not large enough to hold snapshot information.	Ensure you have allocated enough space in your buffer to receive this information.

Notes

The version of this function used in COM Object and C#.NET is named FloatingLicenseTakeSnapshot() and is used in conjunction with the COM Object and C#.NET function FloatingLicenseGetRecord().

As commented in the prototypes, the first C#.NET code given:

```
CKFloatSnapErrorEnum FloatingLicenseTakeSnapshot (long
*pnEntries);
```

is unsafe code.

The second C#.NET code given:

```
FloatSnapErrorEnum FloatingLicenseTakeSnapshot2 (); safe
- plus -

long FloatingLicenseGetNumEntries ();
```

is safe code.

FLS_REC is the structure of each informational entry. It is defined in crypkey.h. The contents of each record are described in Table 23. See also *Referencing the Floating License Snapshot Record* on page 101, where you will find the C++ and C#.NET prototypes for functions used to encapsulate the FLS record.

Table 23: Floating License Snapshot – Record Structure

Item Name	Size (Bytes)	Description
Id	2	unique id for the record, used internally
Update	4	timestamp of last update, used internally
Status	2	<ul style="list-style-type: none"> if running: 0 if waiting: a positive integer representing the queue position
Starttime	4	Server-local timestamp of when the application started running or waiting in the queue — would be seconds since 1970
UserName	16	text user name

Item Name	Size (Bytes)	Description
ComputerName	16	text computer name
Spare	32	reserved

FloatingLicenseTakeSnapshot2()

The FloatingLicenseSnapshot2() function takes a snapshot of the floating licenses. This is the same function as the function FloatingLicenseTakeSnapshot() except that there is no parameter. Use the function FloatingLicenseGetNumEntries() to get the number of records.

PROTOTYPES

Language	Syntax
C#.NET	<pre>public int FloatingLicenseTakeSnapshot2 ()</pre>

Notes

This function operates only in COM Objects and C#.NET Assemblies.

Get1RestInfo()

This function is used to return one of three different types of restriction information, depending on the value of the “which” input parameter.

FUNCTION USAGE

Use this function whenever you need to get license restriction information.

PROTOTYPES

Language	Syntax
C	int Get1RestInfo (int which);
C++	int Get1RestInfo(int which);
VB	<pre>Declare Function Get1RestInfo& Lib "CRP32DLL.DLL" (ByVal which&)</pre>

Language	Syntax
C#.NET	CKGet1ErrorEnum Get1RestInfo (CKWhichEnum nWhich);

PARAMETER

Which

is set to one of the following:

1. Returns authopt (indicating whether the license is unlimited, days-based, or runs-based).
2. Returns num_allowed (indicating the number of days or runs allowed).
3. Returns num_used (indicating the number of days or runs used).

Return Value

Table 24: Get1RestInfo() Values

Name	Value	Message
not defined	0-32768	Returns the which variable as defined above.
G1_OUT_OF_RANGE	-1	The which variable is out of range.

Note

This function is an optional alternative to using the GetRestrictionInfo() function.

GetAuthorization()

This function enables the level at which the protected program is authorized to run along with the options enabled or an authorization failure code. It is also used to decrement the number of runs, if the application is restricted by runs.

FUNCTION USAGE

This function can be called at the programmer's discretion to determine the level of authorization that the user has been granted. It can also be called to decrement the number of uses the program has left if it is restricted by number of 'runs'. This function should be called after calling any function that changes the license state,

such as `SaveSiteKey()`, `TransferOut()` or `TransferIn()`, so that the program can alter its behavior according to the change that has occurred.

PROTOTYPES

Language	Syntax
C	<code>int GetAuthorization(unsigned long *oplevel, int decrement);</code>
C++	<code>int GetAuthorization(unsigned long *oplevel, int decrement);</code>
VB	This function will not work in VB. Please use <code>GetAuthorization2()</code>
C#.NET	<code>CKAuthErrorEnum GetAuthorization (int *pnOptLevel, int nDecrement); // unsafe</code>

PARAMETERS

Oplevel

The `oplevel` is returned in this variable if the program is authorized. It is a combination of option and level information stored in 32 bits.

Decrement

If this number is non-zero and the protected program is restricted to number of runs allowed, the number of runs remaining decrements by this number.

Return Value

Table 25: `GetAuthorization()` Values

Name	Value	Meaning	Solution
not defined	>0	The program is authorized to run. The program has a floating license and more users are requesting to	

Name	Value	Meaning	Solution
		use the program than the license allows. The number indicates how many users must quit before the next user is authorized to run. CrypKey will queue this user's request for access and, providing the user does not quit, will give him/her access on a first-come, first-served basis.	
AUTH_OK	0	The program is authorized to run.	
AUTH_INIT_FAIL	-1	CrypKey has not been initialized.	If you get this error message, please ensure you are calling InitCrypKey() prior to calling this function. If you are calling InitCrypKey() obtain the return value and contact Support@Crypkey.com .
AUTH_DISALLOW_FLOPPY	-2	An attempt is being made to run the protected program from a portable media device.	If this error occurs, you are probably trying to run a protected program from a portable media device. This is not possible. The only situation where CrypKey SDK 7 can run on a portable media device is when using the CrypKey USBKey technology.
AUTH_BAD_PATH	-3	Could not read the key file.	If this error occurs, there is a problem reading the key file on your hard drive. You must have full control of the application directory to access the key file.
AUTH_NOT_	-4	The program has never been authorized at the	If you receive this error, the program needs to

Name	Value	Meaning	Solution
PRESENT		current site.	be authorized. You can authorize the program using the Site Key Generator.
AUTH_DIFFERENT	-5	The program is moved or the password is incorrect.	If you receive this error, ensure the User Key you are using in the InitCrypkey() function call is the same as the User Key that was given to you by CrypKey. Also ensure that the password that you configured in the Site Key Generator is the same password that you specified to CrypKey. If you continue to receive this error, please email Support@Crypkey.com .
AUTH_BAD_MASTERKEY	-6	The Master Key is not correct for this program.	If you receive this error, ensure the Master Key you are using is identical to the Master Key that CrypKey provided to you and that the filename that you specified in the InitCrypKey() filepath parameter is the same as the filename you registered with CrypKey to obtain your developer keys. If you continue to have problems, please email Support@Crypkey.com .
AUTH_SITEKEY_CRC	-7	The site key file is damaged or tampered with.	If you receive this error, delete the license files (*.rst, *.41s, *.key and *.ent) and run the application again. You will need to provide another authorization

Name	Value	Meaning	Solution
			once the the license files are deleted. Please email Support@CrypKey.com if this does not work for you.
AUTH_TIME_TOO_EARLY	-8	The system time is set to a date earlier than the date on which the Site Key was issued.	This error occurs when the system time is changed by more than 75 minutes. You must re-authorize the software if this error occurs.
AUTH_TIME_SETBACK	-9	The program is restricted by days and the system time is set back.	This error occurs when the system time is moved. You must re-authorize the software if this error occurs.
AUTH_TIME_RUNOUT	-10	The program is restricted by days and the time has expired.	The user's days limited license is expired. The user will need to be re-authorized to use the software.
AUTH_RUNS_RESTR	-11	The program is restricted by runs and the number of runs allowed has been exceeded.	The number of runs the user was authorized for has been exceeded. The user will need a new authorization to run the software.
AUTH_NOT_ENOUGH_RUNS	-12	The user tried to perform an operation that requires more runs than are available.	In order to run the protected application the user must have more runs available than the decrement parameter. When authorized by runs.
AUTH_MISSING_RST_FILE	-13	The restriction file is deleted.	The CrypKey-protected application cannot find the .rst file in the application directory. If this file is moved or deleted, the protected

Name	Value	Meaning	Solution
			application will not run. The only way to solve this problem is to re-authorize the user.
AUTH_RST_BAD_CRC	-14	The restriction file is damaged or tampered with.	This error occurs if the user has attempted to alter the .rst file. The only solution to this problem is to re-authorize the user.
AUTH_RST_BAD_LOCATION	-15	The restriction file is moved or overwritten.	This error occurs if the user has moved, overwritten or tampered with the .rst file. If the user gets this error you must reauthorize.
AUTH_ENTRY_CHECK_FAIL	-16	The user has attempted to reuse an old site key.	This error is created when a user is trying to enter a site key that was not created for the present site code. Request the user's current Site Code and reauthorize.
AUTH_NETTABLE_FILE_FAIL	-17	Could not open or read the network user table file.	This error will occur in drive share or normal mode. If this error occurs, ensure the user is able to read and write to the license file directory. When running CrypKey protected applications we create a file with the extension .tb2. This file is opened and closed every time you run a CrypKey-protected application.
AUTH_NETMAX_EXCEEDED	-18	More people are using the protected program than SLAPI can handle.	This error occurs when more users try to use the protected application than authorization restrictions

Name	Value	Meaning	Solution
			allow. Authorize the end user for more network seats.
AUTH_NETWORK_NOT_ALLOWED	-19	An attempt is being made to run the protected application over a network, or via remote desktop/terminal services without a floating license being issued.	This error occurs when the user who is trying to run the application has not received a site key that has network privileges. If the application is being run via remote desktop or terminal services, you must issue a network license for as many users as you would like to currently run the software from the remote session.
AUTH_RSTFILE_WRITE_PROTEC	-20	.RST file is write protected	Ensure that the correct permissions are applied to the application directory. Contact Support@Crypkey.com if you have more questions.
AUTH_TIME_CLOCK_TAMPERING	-21	Computer clock tampering detected	Reboot your computer. If the computer's time clock is simply out of sync, the reboot will allow the application to run again without reauthorization as it will re-sync the time clock.
KEY_BAD_LOCATION	-22	Application site key is moved or copied	This error occurs in the .key file has been tampered with. You need to reauthorize to use the protected application.
AUTH_NGN_VERSION_OUTDATED	-23	NGN file being used in not the current version	This error occurs if you are using an older version of the NGN file. This is most likely to

Name	Value	Meaning	Solution
			occur on a major version upgrade of your CrypKey-protected software. Make sure the NGN file you are using is up to date. If the problem persists contact Support@Crypkey.com
AUTH_INVALID_STATIC_KEY	-24	Your Static SiteKey is not valid	The key does not match this software. Ask the vendor to check and re-issue this key
AUTH_USBKEY_NOT_FOUND	-25	A USB key with a license was not found	This software requires a licensed USBKey be connected to the computer. Ask the software vendor for a key and/or check that it is plugged in to this computer
AUTH_NETWORK_MAXLICENSE_EXCEEDED	-26	The maximum number of network seats allowed by this application's MasterKey has been exceeded	Check with the software vendor to get a license for a reduced number of seats, or a version of the software that supports more seats

General error values also apply — see *General Error Values* on page 167.

Notes

The options are single bits of information that can be turned on or off individually when a license is created using the Site Key Generator. The options exist so that you can turn different features of your product on or off.

The Level is a number that can also be set by the Site Key Generator. The Level can be used to incrementally command the quality of some service your software provides, such as the number of records allowed for a database.

The Options are stored in *oplevel* as follows:

Option #1 -> bit 31=1 if enabled, 0 if not

Option #2 -> bit 30=1 if enabled, 0 if not

Option #3 -> bit 29=1 if enabled, 0 if not

Option #n -> bit (32-n)=1 if enabled, 0 if not

The number of bits (n) used by Options depends on the number of Options you have configured for the program in the SKW.INI file, to a maximum of 32. The rest of the bits left over can be used for the Level.

For example, if four options are defined in *skw.ini*, *oplevel* looks like this:

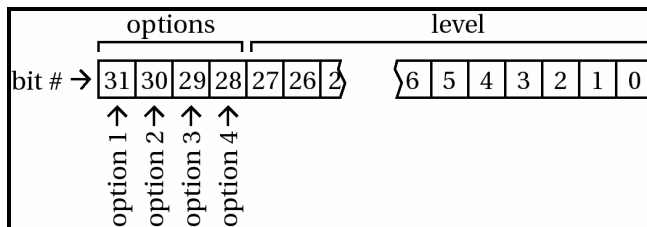


Figure 86 Options and Levels

In this example, the maximum level is 228 or 268,435,456.

InitCrypkey() must be called at least once by your application before using this function and this function must be called at least once before calling any other SLAPI function.

The Options or Level can also be obtained by using *GetOption()* or *GetLevel()*. For security reasons, this is the only function, along with *GetAuthorization2()*, that can tell you if an incorrect password has been entered into the Site Key Generator. *InitCrypkey()* must be called at least once by your application before using the *GetAuthorization()* function, and this function must be called at least once before any other SLAPI function is called.

The C#.NET prototype for the *GetAuthorization()* function is unsafe code. The safe C#.NET alternative is *GetAuthorization2()*.

Example:

An application with 27 Options would use bits 31 down to 5 for Options 1 to 27. Five bits are left to form the Level with a value range of 0 to 31. Once again, the Options bits are stored in the high end of the oplevel and the Level bits are in the low end.

GetAuthorization2()

This function is similar to [GetAuthorization\(\)](#) except that it does not get oplevel information.

FUNCTION USAGE

This function is used in the same manner as [GetAuthorization\(\)](#) and is intended for languages that are unable to obtain the oplevel data or for cases where the programmer does not want to do this.

PROTOTYPES

Language	Syntax
C	<code>int GetAuthorization2 (int decrement);</code>
C++	<code>int GetAuthorization2(int decrement);</code>
VB	<code>Declare Function GetAuthorization2& Lib "CRP32DLL.DLL" (ByVal decrement&)</code>
C#.NET	<code>CKAuthErrorEnum GetAuthorization2 (int nDecrement);</code>

PARAMETER

Decrement

If this number is non-zero and if the protected program has been restricted by the number of 'runs' allowed, the number of 'runs' remaining is decremented by this number.

Return Value

See `GetAuthorization()` for a description of the return values for this function.

Notes

`InitCrypkey()` must be called at least once by your application before using this function and this function must be called at least once before any other SLAPI function.

For security reasons, this is the only function, along with `GetAuthorization()`, that can tell you if an incorrect password is entered into the Site Key Generator.

Either `GetAuthorization()` or `GetAuthorization2()` must be called after `InitCrypKey()`.

GetCustomInfoBytes

This function gets data that was written to file by the `SetCustomInfoBytes()` function.

FUNCTION USAGE

`SetCustomInfoBytes()` and `GetCustomInfoBytes()` can be used by your application to save data in a safe place. The data is encrypted before being written to the hard drive, and `CrypKey` will insure the file is not compromised by an attacker replacing it with a previous version of this file.

Use this function to store sensitive data, such as licensing data, that needs to be protected from being altered by users.

PROTOTYPES

Language	Syntax
C	<pre>int GetCustomInfoBytes(unsigned char * bpdataArray);</pre>
C++	<pre>int GetCustomInfoBytes(unsigned char * bpdataArray);</pre>
VB	This function is not available in VB

PARAMETERS

BpDataArray This must be an unsigned character buffer and must be 500 bytes in size. The function will copy the data previously written by SetCustomInfoBytes() into the buffer.

Return Values Note that this function is not covered by the ExplainErr() function.

ERROR VALUE	MEANING
0	Function performed Successfully
-1	Could not find custom byte info file
-2	Could not open custom byte info file
-3	CRC Error on Custom Info Byte file
-4	Custom info byte file moved or replaced
-5	Customer info byte file corrupted

GetDrivePermanentSerialData()

This function returns the system drive burned in serial and manufacturing data.

FUNCTION USAGE

GetDrivePermanentSerialData() is a proprietary CrypKey software function that accesses the following hard drive information on a customer's computer: model number, serial number, firmware data.

GetLastError()

This function returns the text for the last error. Unlike the SDK functions ExplainErr() and ExplainErr2(), the function code does not need to be supplied. The function code is set each time one of the functions in the COM object is called. Thus, this function returned only the error text for the last function called.

Prototypes

Language	Syntax
C	[n/a]

C++	BSTR *GetLastError ();
VB	GetLastError () as String
C#.NET	String *GetLastError ();

Notes

See also GetLastErrorCode().

GetLastErrorCode()

This function, used only in C#.NET Assemblies, returns only the number for the last error. Unlike the SDK functions ExplainErr() and ExplainErr2(), the function code does not need to be supplied. The function code is set each time one of the functions in the COM object is called. Thus, this function returns only the error number for the last function called.

PROTOTYPES

Language	Syntax
C	[n/a]
C++	[n/a]
VB	[n/a]
C#.NET	int GetLastErrorCode();

GetLevel()

This function returns the level at which the protected program is authorized to run.

FUNCTION USAGE

You may use this function at your discretion.

PROTOTYPES

Language	Syntax
C	unsigned long GetLevel (int

	numDefineOpts);
C++	long GetLevel (long nNumOptions);
VB	GetLevel (nNumOptions as Long) as Long
C#.NET	int GetLevel (long nNumberOfOptions);

PARAMETER

NumDefineOpts

NumDefineOpts is the number of options that the protected program has defined.

Return Value

The return value is the current level that is available to the user from the program.

GETNETHANDLE()

The GetNetHandle() function gets and returns the network license handle of the protected program.

FUNCTION USAGE

If you want several SLAPI-licensed programs sharing a single network seat, you can use this function to get the handle and then distribute it using the SetNetHandle() function. This function should be called only after successful calls to InitCrypkey() and GetAuthorization() have been made.

PROTOTYPES

Language	Syntax
C	unsigned short GetNetHandle ();
C++	long GetNetHandle ();
VB	GetNetHandle () as Long
C#.NET	int GetNetHandle ();

Return Value

This function returns the handle to a network resource.

Notes

The user must design a way to transmit the return value to other executables sharing the network seat (see the `SetNetHandle()` function description in `SetNetHandle()` on page 232 for more information.

Only one of the programs sharing the network seat can call `EndCrypkey()` successfully — preferably, the last program to stop executing calls this function since this means that usage of the network seat is clearly no longer required.

GetNumCopies()

`GetNumCopies()` obtains the number of copies that the site is granted under the current license provided to it.

FUNCTION USAGE

If you want to grant multiple copy site licenses, the program should display the number of copies it is granted to the user. The copies can be distributed by the user through the transfer functions.

PROTOTYPES

Language	Syntax
C	<code>int GetNumCopies ();</code>
C++	<code>Long GetNumCopies ();</code>
VB	<code>GetNumCopies () as Long</code>
C#.NET	<code>CKCopiesErrorEnum GetNumCopies ();</code>

Return Value

Table 26: `GetNumCopies()` Values

Name	Value	Message
Not defined	0-255	Returns the number of license copies available.
GNC_CRYPTKEY_NOT_INITIALIZED	-1	CrypKey has not been initialized.

GetNumMultiUsers()

GetNumMultiUsers() gets the number of users that the site has granted in its license.

FUNCTION USAGE

If you want to grant multiple-user site licenses, the application should display the number of users it has granted to the user.

PROTOTYPES

Language	Syntax
C	int GetNumMultiUsers ();
C++	long GetNumMultiUsers ();
VB	GetNumMultiUsers () as Long
C#.NET	int GetNumMultiUsers ();

Return Value

This function returns the number of concurrent network uses granted in the license.

GetOption()

GetOption() determines the status of a specific option. The function gets the option bit specified in the current authorization license.

FUNCTION USAGE

You can call this function at your discretion.

PROTOTYPES

Language	Syntax
C	int GetOption (int numDefineOpts, int optnum);

Language	Syntax
C++	long GetOption (long nNumOptions, long nOptionNum);
VB	GetOption (nNumOptions as Long, nOptionNum as Long) as Long
C#.NET	CKOptionErrorEnum GetOption (int nNumberOfOptions, int nOptionNumber);

PARAMETERS

NumDefineOpts

NumDefineOpts is the number of options that the protected program has.

Optnum

Optnum is the number of the option for which you want a status.

Return Value

Table 27: GetOption() Values

Name	Value	Message
OPTION_ON	1	The specified option is on.
OPTION_OFF	0	The specified option is off.
OPTION_UNAUTHORIZED	-1	The program is not authorized.

Note

This function is a simpler alternate to obtaining the option from the information passed by the GetAuthorization() function.

GetRestrictionInfo()

GetRestrictionInfo() gets information on any restrictions that are present in the license.

FUNCTION USAGE

It is not necessary to take any action on this information, however, you may want to display this information at your discretion.

PROTOTYPES

Language	Syntax
C	int GetRestrictionInfo (int *authopt, unsigned long *start_date, int *num_allowed, int *num_used);
C++	long GetRestrictionInfo (long *pnAuthOpt, long *pnStartDate, long *pnNumAllowed, long *pnNumUsed);
VB	GetRestrictionInfo (nAuthOpt as ByRef Long, nStartDate as ByRef Long, nNumAllowed as ByRef Long, nNumUsed as ByRef Long) as Long
C#.NET	CKRestrictionErrorEnum GetRestrictionInfo (int *pnAuthOpt, int *pnStartDate, int *pnNumberAllowed, int *pnNumberUsed);

PARAMETERS

Authopt

Authopt is set to one of the values listed in the following table (as defined in cryptkey.h):

Table 28: Authopt Values

Name	Value	Meaning
RESTR_NONE	0	Unlimited
RESTR_TIME	1	Days
RESTR_RUNS	2	Runs

Start_date

Start_date is set to the date the license was issued, in seconds since the start of the year 1970 (Universal Coordinated Time).

Num_allowed

Num_allowed is set to the number of days or runs allowed by the current license.

Num_used

Num_used is set to the number of days or runs in the current license used up so far.

Return ValueTable 29: *Num_used* Values

Name	Value	Meaning
GRI_OK	0	The restriction information is valid.
GRI_INVALID	-1	The restriction information is not valid.

Notes

The `GetRestrictionInfo()` function is valid only if the protected program is authorized to run or if the program is not authorized to run due to expired restrictions (i.e., if the days or runs have expired).

The algorithm for breaking down the *start_date* for display in C code is as follows:

```
#include "time.h"

char* _sgtime(long ltime);

m_datetime = _sgtime(start_date);

char* _sgtime(long ltime)
{
    static char time_str[60];
```

```

    struct tm *t;

    ltime -= 2209075200;

    t = gmtime(&ltime);

    if(t != NULL)
    {
        strcpy(time_str,_ultoa(t->tm_mday,text,10));
        strcat(time_str,"-");
        strcat(time_str,_ultoa(t->tm_mon+1,text,10));
        strcat(time_str,"-");
        strcat(time_str,_ultoa(t->tm_year,text,10));
        strcat(time_str," ");
        strcat(time_str,_ultoa(t->tm_hour,text,10));
        strcat(time_str,":");
        strcat(time_str,_ultoa(t->tm_min,text,10));
        strcat(time_str,":");
        strcat(time_str,_ultoa(t->tm_sec,text,10));
    }
    else
    {
        strcpy(time_str,"Predates 01-01-70  00:00:00");
    }

    return(time_str);
}

```

In Visual Basic, you can pass the value of *start_date* returned by `GetRestrictionInfo()` to a single precision variable and then add 216 (65,536) to the result to get the correct number. This, in effect, replaces the bit that was truncated because Visual Basic's long integer is signed, but the value that is returned is unsigned.

GetSiteCode()

This function gets the site code for the computer location. The Site Code must be reported by the user before the developer can issue a Site Key.

FUNCTION USAGE

If the protected program detects it is not authorized via the `GetAuthorization()` function, it must, at minimum, show the program's site code and allow the user to enter a site key.

PROTOTYPES

Language	Syntax
C	<code>int GetSiteCode(char *site_code);</code>
C++	<code>int GetSiteCode(char *site_code);</code>
VB	Declare Function GetSiteCode& Lib "CRP32DLL.DLL"
C#.NET	<code>CKCodeErrorEnum GetSiteCode (StringBuilder *pbstrSiteCode);</code>

PARAMETER

Site_code

Site_code must be a character buffer of at least `CK_SITECODE_LEN` bytes in size. The function stores the site code in the buffer.

Return Value

Table 30: GetSiteCode() Values

Name	Value	Message	Solution
GSC_OK	0	Function was performed successfully.	
GSC_CRYPTKEY_NOT_INITIALIZED	-1	CrypKey has not been initialized.	This occurs if there is a problem in the InitCrypkey() function. Please refer to the description about this function for more information. If you continue to receive this message, please contact Support@Crypkey.com.
GSC_ENTRY_FILE_OPEN_FAIL	-2	The entry file could not be opened.	This occurs when you cannot access the .ent file. You must be able to read from and write to this file. If you continue to receive this message, please contact Support@Crypkey.com.

GetSiteCode2()

This function returns a pointer to the Site Code for the program location. The Site Code must be reported by the user before the developer can issue a Site Key.

FUNCTION USAGE

If the program detects it is not authorized via the GetAuthorization() function, it must, at minimum, show the program's site code and allow the user to enter a site key.

PROTOTYPES

Language	Syntax
C	char *GetSiteCode2 (void);
C++	BSTR *GetSiteCode2 (void);

Language	Syntax
VB	GetSiteCode2 () as ByRef String
C#.NET	String *GetSiteCode2 ();

Return Value

This function returns a pointer to the Site Code. This is a pointer to an internal buffer in the library. This internal buffer is large enough to safely pass the pointer to CKFormat with spaces().

Note

This function is an optional alternative to using the GetSiteCode() function.

InitCrypkey()

This function initializes CrypKey with runtime information and must be called before any other function in SLAPI except SetNetHandle().

FUNCTION USAGE

Call InitCrypkey() at the program startup. For example, if you are using C++, a good place to do this is in the application object constructor.

PROTOTYPES

Language	Syntax
C	int InitCrypkey (char *filepath, char *masterkey, char *userkey, int allow_floppy, unsigned network_max_checktime);
C++	long InitCrypKey (BSTR bstrFilePath, BSTR bstrMasterKey, BSTR bstrUserKey, long nAllowFloppy, long nNetworkTimeout);
VB	Declare Function InitCrypkey& Lib "CRP32DLL.DLL" (ByVal filepath\$, ByVal masterkey\$, ByVal userkey\$, ByVal allow_floppy&, ByVal

Language	Syntax
	<code>network_max_checktime&</code>
C#.NET	<pre>CKInitErrorEnum InitCrypKey (String *pstrFilePath, String *pstrMasterKey, String *pstrUserKey, int nAllowFloppy, int nNetworkTimeout);</pre>

PARAMETERS

filepath

This is the pathname of the program file that will be checked for authorization.

masterkey

This is the hexadecimal string assigned to your program by CrypKey (Canada) Inc. at purchase time.

userkey

This is the hexadecimal string assigned to your password by Kenonic at purchase time.

allow_floppy

Set this to 1 to allow authorization of a program on a floppy. This is usually set to 0, as the entire floppy can be successfully bit copied.

network_max_checktime

This is the time, in seconds, that CrypKey will wait for a program to check for authorization under a floating license. After this time, the user will be deleted from the queue of users who have, or are waiting for, access. This is how CrypKey handles users who may not have logged off properly by using *EndCrypkey()* (perhaps because their computer crashed). If the user still desires access to the program, the request will be moved to the back of the queue behind any other users who may be waiting. The number you choose for this parameter will depend on how often you call *GetAuthorization()*, but it will also determine how fast a hung computer is detected by CrypKey and its license is released for others to use. It is recommended to call *GetAuthorization()* every 5 to 20 minutes, and *network_max_checktime* should be 2 to 3 times this number.

Return Value

Table 31: InitCrypkey() Values

Name	Value	Meaning	Solution
INIT_OK	0	CrypKey initialization was successful.	
INIT_FILE_NOT_FOUND	-1	The program file could not be located.	This error occurs if the license file that you specified is not located in the application directory or the directory specified by the filepath parameter. You must ensure that the file that you specified in the filepath parameter of the InitCrypKey() function call is in the location that you stated. If you continue to receive this error, please contact support@crypkey.com.
INIT_MASTERKEY_CRC_FAILURE	-2	The Master Key in the program source code is incorrect.	If you receive this error, ensure the file name in the filepath parameter matches the file name you sent to CrypKey for your Master Key. Also, ensure the Master Key that was sent to you is the same as the Master Key that is in your masterkey parameter in the InitCrypkey() function call. If you continue to receive this error, please contact Support@Crypkey.com.
INIT_BAD_PRODUCT_NAME	-3	The Master Key does not match the program name.	This error occurs if the Master Key in the InitCrypkey() function does not match the file name you specified in the InitCrypkey()

Name	Value	Meaning	Solution
			function.
INIT_KEYFILE_CREATION_FAIL	-4	There was no available disk space to write the key file or the directory is write protected.	If you encounter this error, ensure you have full permission to the application directory. CrypKey must be able to read and write directly to the hard drive. If you continue to receive this error, please contact support@crypkey.com.
INIT_NETWORK_NOT_PURCHASED	-5	An attempt is being made to run the program in a network context without purchasing the network license.	If this error occurs, ensure you have the ability to create network licenses.
INIT_MULTIPLE_CALL_TO_INIT	-10	This function has already been called during the execution of the protected program.	This error occurs if you call InitCrypkey() more than once before the EndCrypKey() function is called. If you need further assistance, please contact Support@Crypkey.com.
INIT_32_DLL_CORRUPT	-14	A file is corrupt or has been tampered with.	If you need further assistance, please contact Support@Crypkey.com.
INIT_PRE_V6_MASTERKEY	-21	You are using a master key from an older version of CrypKey than you are presently using.	Contact CrypKey to date developer keys. Please note that generally, Interversion builds do not require a new Master Key, however major version updates do require new Master Keys. To request your new Master Keys for a new version of CrypKey e-mail

Name	Value	Meaning	Solution
			authorize@Crypkey.com with your company name and customer service number.
INIT_NGN_VERSION_OUTDATED	-22	You are using a older version of the NGN	Update your NGN. Contact Support@crypkey.com if the problem persists.
INIT_NGN_FILE_NOT_FOUND	-23	The InitCrypKey() does not find the NGN in the location specified.	Ensure that the NGN file is in the license file location. Contact Support@crypkey.com if the problem persists.
INIT_DRIVER_FAILED_OR_BUSY	-24	The CrypKey License service is not currently operating normally	Ensure that the CrypKey License service was installed by administrator and that it is not disabled. Make sure the local everyone group and local user has full control of the application directory. If the problem persists contact Support@Crypkey.com
INIT_USBKEY_NOT_FOUND	-25		
INIT_MASTERKEY_OUTDATED	-26		
INIT_SERVER_NOT_FOUND	-27		
INIT_SERVER_ALREADY_EXISTS	-28		
INIT_COULDNOT_START_NGN	-29		
INIT_CED_APP	-30		

Name	Value	Meaning	Solution
_MISMATCH			

General error values also apply — see the section *General Error Values* on page 167.

IsEasyLicense()

DESCRIPTION

This function indicates if the current license is an Easy License.

The "EasyLicense" property is assigned during the creation of a Site Key. It has the advantage that it can be backed up and restored, and will always work for the particular PC the license was generated for. Due to this ability, the license does not support any license transfer functions.

FUNCTION USAGE

It is not actually necessary to call this function in normal CrypKey usage. Use this function if you wish to get this data for your own purposes.

DECLARATION (C/C++)

```
int EasyLicense (void);
```

DECLARATION (VISUAL BASIC)

```
Declare Function EasyLicense& Lib "CRP32DLL.DLL" ( )
```

PARAMETERS

None.

RETURN VALUE

This function returns 1 if the current license is an EasyLicense, 0 if not. If there is no existing valid license, this function returns a negative number.

KillLicense()

DESCRIPTION

This function is used to disable an existing license and provides a Confirmation Code that can be entered into the Site Key Generator in order to display the details of the de-authorization.

FUNCTION USAGE

The programmer would use this function to remove an existing license and provide the customer with the means of proving that the license has been deactivated via the Confirmation Code.

DECLARATION (C/C++)

```
int KillLicense(char * confirmCode);
```

DECLARATION (VISUAL BASIC)

```
Declare Function KillLicense& Lib "CRP32DLL.DLL" (ByRef  
confirmCode$)
```

PARAMETERS

ConfirmCode

This is a hexadecimal string which can be entered into the Site Key Generator to display information confirming that the license has been deactivated. This is only written to if the function succeeds and returns KL_OK. Use the CK_SITEKEY_LEN as the buffer for the Site Key.

RETURN VALUES

Defined name in crypkey.h	Value	Error message
KL_OK	0	Function was performed successfully.
KL_CRYPTKEY_NOT_INITIALIZED	-1	CrypKey has not yet been initialized.
KL_CRYPTKEY_NOT_AUTHORIZED	-2	The protected program is not authorized.

KL_LICENSE_WRITE_PROTECTED -3

The license files are currently write protected.

Note: The Confirmation Code provided by this function can be entered in the Site Code field of the Site Key Generator. Information about the license that was terminated will be displayed when the 'check' button is pressed.

This function does not remove the special hard-drive signature of the Automatic Licensing functions.

Notes

The confirmation code provided by this function can be entered in the Site Code field of the Site Key Generator. Information about the license that was terminated is displayed when the **Check** button is clicked.

This function does not remove the special hard-drive signature of the automatic licensing functions.

As commented in the C#.NET prototype, the first code given:

```
CKKillErrorEnum KillLicense (StringBuilder
    *pbstrConfirmCode);
```

is unsafe code.

The second C#.NET code given:

```
String *KillLicense ();
```

is safe code.

ReadyToTry()

DESCRIPTION

This function implements a one-time Automatic License for a CrypKey-protected application. This license will be based on a 'days' time period without version control.

USAGE

This function needs to be called only once, just after the InitCrypkey() call is made. The first time it is called, it will take a minute or so to set up the Automatic

License. For subsequent calls at the same site, this function will take no action as it will detect that it has already been run.

DECLARATION (C/C++)

```
int ReadyToTry(unsigned long oplevel, int numDays);
```

DECLARATION (VISUAL BASIC)

```
Declare Function ReadyToTry& Lib "CRP32DLL.DLL" (ByVal  
oplevel&, ByVal numDays&)
```

PARAMETERS

Oplevel

These are the 32 bits which make up the Level and Options. The value that is passed here will be returned as the oplevel in the GetAuthorization() function.

numDays

This is the number of days that you wish to allow the program to be temporarily authorized for.

RETURN VALUES

Defined name in crypkey.h	Value	Error message
RTT_OK	0	The function completed successfully.
RTT_COULD_NOT_GET_SITE_CODE	-1	The function could not get the Site Code information.
RTT_COULD_NOT_GET_SITE_KEY	-2	The function could not get the Site Key information.
RTT_COULD_NOT_SAVE_SITE_KEY	-3	The function could not save the Site Key to

Defined name in cryptkey.h	Value	Error message
		the hard-drive.
RTT_RESERVED_4	-4	Reserved.
RTT_BAD_DOS	-5	MS-DOS 3.31 or higher must be running on the site.
RTT_BAD_TRUENAME	-6	Could not get the full pathname of the protected program.
RTT_NO_REDIRECT	-7	This function does not work on network-redirector drives.
RTT_NO_DPB	-8	Could not get the drive parameter block from the site hard-drive.
RTT_NO_MEM	-9	Not enough memory was available to execute this function.
RTT_CANT_GET_CLUSTER	-10	Could not read a cluster from the site hard-drive.
RTT_BAD_STAT	-11	Could not get file statistics from the site hard-drive.
RTT_NO_ROOM	-12	The site hard-drive has run out of space.
RTT_BAD_SECTOR_READ	-13	There has been

Defined name in crypkey.h	Value	Error message
		a bad sector read from the site hard-drive.
RTT_BAD_SECTOR_WRITE	-14	There has been a bad sector write to the site hard-drive.
RTT_FILE_SEARCH	-15	The file search of the site hard-drive has failed.
RTT_FILE_ACCESS	-16	The function could not access the protected program file.
RTT_FILE_NOT_FOUND	-17	The function could not find the protected program file.
RTT_FILE_OPEN	-18	The function could not open the protected program file.
RTT_DONE_THIS	-19	This program has already had its trial period at this hard-drive site.
RTT_NO_SIG	-20	The function could not find a signature on the site hard-drive.
RTT_NO_LISTFILE	-21	The function could not find a listfile on the site hard-drive.

Defined name in <code>crypkey.h</code>	Value	Error message
<code>RTT_CANT_FIND_DRIVE</code>	-22	The function could not find the site hard-drive.
<code>RTT_NO_GOOD_PUTS</code>	-23	The function could not find a good signature on the site hard-drive.
<code>RTT_NO_REAL_DRIVE</code>	-24	The function could not find a real drive.
<code>RTT_32BIT_FILE_ACCESS</code>	-25	Windows 3.11 32-bit file access must be disabled.
<code>RTT_CLOSE_ALL_FILES</code>	-26	Another program has a disk lock on the site hard-drive.
<code>RTT_MAX_RTTREQ_EXCEEDED</code>	-27	The maximum number of days is currently set at 30.

Note

This function does not feature version control like its derivative variants *ReadyToTryDays()* and *ReadyToTryRuns()*. For programming purposes, its version number parameter is always considered to be zero.

ReadyToTryDays()

The `ReadyToTryDays()` function implements a one-time automatic license for a `CrypKey`-protected application. This license is based on a days time period with version control.

This function needs to be called only once, just after the `InitCrypkey()` call is made. The first time it is called, it will take a minute or so to set up the Automatic

License. For subsequent calls at the same site, this function will take no action unless the version number is incremented.

PROTOTYPES

Language	Syntax
C	<pre>int ReadyToTryDays (unsigned long oplevel, int numDays, int version, int copies);</pre>
C++	<pre>int ReadyToTryDays(unsigned long oplevel, int numDays, int version, int copies);</pre>
VB	<pre>Declare Function ReadyToTryDays% Lib ""CRP32DLL.DLL" (ByVal oplevel&, ByVal numDays&, ByVal version&, ByVal copies&)</pre>
C#.NET	<pre>CKTryErrorEnum ReadyToTryDays (int nOptLevel, int nDays, int nVersion, int nCopies);</pre>

PARAMETERS

Oplevel

These are the 32 bits which make up the Level and Options. The value that is passed here will be returned as the oplevel in the GetAuthorization() function.

NumDays

This is the number of days that you wish to allow the program to be temporarily authorized for.

Version

This number is used to distinguish one Automatic License implementation from another, which will allow subsequent licenses to be created by newer versions of the protected program.

Copies

This is the number of fixed license copies that you wish to grant the user when this function is run.

Return Value

See *ReadyToTry()* on page 220 for a list of return values.

Notes

The version parameter normally is begun at 1 and proceeds incrementally upwards from there for each new release of the program.

In order to create a floating license with a specific number of multi-users, you can enter a negative value in the copies parameter. This results in a floating license with a number of users equal to the absolute value of the number entered (e.g., -7 would create a floating license with 7 users).

ReadyToTryRuns()

The *ReadyToTryRuns()* function implements a one-time Automatic License for a CrypKey protected application. This license is based on a 'runs' restriction with version control.

FUNCTION USAGE

This function needs to be called only once, just after the *InitCrypkey()* call is made. The first time it is called, it will take a minute or so to set up the Automatic License. For subsequent calls at the same site, this function will take no action unless the version number is incremented.

PROTOTYPES

Language	Syntax
C	<pre>int ReadyToTryRuns (unsigned long oplevel, int numRuns, int version, int copies);</pre>
C++	<pre>int ReadyToTryRuns(unsigned long oplevel, int numRuns, int version, int copies);</pre>
VB	<pre>Declare Function ReadyToTryRuns& Lib "CRP32DLL.DLL" (ByVal oplevel&, ByVal numRuns&, ByVal version&, ByVal copies&)</pre>
C#.NET	<pre>CKTryErrorEnum ReadyToTryRuns (int nOptLevel, int nRuns, int nVersion, int nCopies);</pre>

PARAMETERS

Oplevel

These are the 32 bits that comprise the level and options. The value that is passed here is returned as the *oplevel* in the `GetAuthorization()` function.

NumRuns

This is the number of runs that you wish to allow the program to be temporarily authorized for.

Version

This number is used to distinguish one Automatic License implementation from another, which will allow subsequent licenses to be created by newer versions of the protected program.

Copies

This the number of fixed license copies that you wish to grant the user when the function is run.

Return Value

See the `ReadyToTry()` function description in `ReadyToTry()` on page 220 for a list of return values.

Notes

The version parameter normally begins at 1 and proceeds incrementally upwards for each new release of the program.

In order to create a floating license with a specific number of multi-users, you can enter a negative value in the copies parameter. This results in the creation of a floating license with a number of users equal to the absolute value of the number entered (e.g., -7 creates a floating license with 7 users).

RegisterTransfer()

This function is part of the following sequence of functions used to transfer a license using some sort of media such as floppy or USB drive:

Step 1. *RegisterTransfer()* - performed by the instance of the application that is at the target location of the license to be transferred.

Step 2. *TransferOut()* - performed by the instance of the application that is at the source location of the license to be transferred.

Step 3. *TransferIn()* - performed by the instance of the application that is at the target location of the license to be transferred.

This function is called from an application that does not yet have authorization and will register the application by placing registration files on a disk or directory. These registration files are used by an application that does have authorization to transfer the authorization out.

FUNCTION USAGE

The directory for transfer is most likely A:\. because the transfer is usually from one computer to another using a floppy disk. However, this transfer could be from one directory to another on the same computer, or even from one network drive to another. For this reason, the user must be able to enter this parameter.

PROTOTYPES

Language	Syntax
C	<code>int RegisterTransfer (char *directory);</code>
C++	<code>long RegisterTransfer (BSTR bstrDirectory);</code>
VB	<code>Declare Function RegisterTransfer& Lib "CRP32DLL.DLL" (ByVal directory\$)</code>
C#.NET	<code>CKRegisterErrorEnum RegisterTransfer (String *pstrDirectory);</code>

PARAMETER

Directory

This is a pointer to a null-terminated text string that contains the path of the directory in which to place some encrypted text files (the registration).

Return Value

Table 32: RegisterTransfer() Values

Name	Value	Message	Solution
REG_OK	0	The function completed successfully.	
REG_THIS_ALREADY_AUTHORIZED	-1	Only unauthorized sites can receive transferred licenses.	This error occurs if there is already a license present for the application. You can only transfer a license into an unauthorized application. This includes trial licenses.
REG_COULDNOT_OPEN_TARGET_REGFILE	-2	The file could not be opened in the supplied directory due to invalid directory, write protection, disk not ready or no disk space.	If this error occurs, please ensure you have read and write access to the application directory. If you continue to receive this error, please contact Support@Crypkey.com.
REG_TARGET_ALREADY_REGISTERED	-3	There are already imprint files present in the supplied directory.	This error occurs if the application you try to transfer already has registration. This also occurs if you have tried to register the transfer more than once.
REG_SOURCE_ALREADY_REGISTERED	-4	The site can only have one outstanding registration at a time.	This error occurs if the application you try to transfer already has registration. This also occurs if you have tried to register the transfer more than once.
REG_CANNOT_OPEN_REGFILE	-5	The matching registration file in the same	If this error occurs, please ensure you have read and write access

Name	Value	Message	Solution
		directory as the program could not be opened.	to the application directory. If you continue to receive this error, please contact Support@Cypkey.com.
REG_CANNOT_WRITE_REGFILE	-6	The matching registration file in the same directory as the application could not be written to.	If this error occurs, please ensure you have read and write access to the application directory. If you continue to receive this error, please contact Support@Crypkey.com.

General error values also apply — see the section *General Error Values* on page 167.

Note

See *Chapter 10: Moving Protected Programs* for more information.

SaveSiteKey()

This function is used to save a Site Key that has been acquired by the user in order to authorize a license. Before the Site Key is saved it is checked against the Site Code. GetSiteCode() must be called in order to call SaveSiteKey(). Therefore, if the application has been closed down since the Site Code was given, ensure that all the appropriate function calls are made prior to trying to run SaveSiteKey().

FUNCTION USAGE

If the program detects it is not authorized via the GetAuthorization() function, it should, at minimum, show the program's site code and allow the user to enter a site key.

PROTOTYPES

Language	Syntax
C	<code>int SaveSiteKey (char *site_key);</code>
C++	<code>int SaveSiteKey (char</code>

Language	Syntax
	<code>*site_key);</code>
VB	<code>Declare Function SaveSiteKey& Lib "CRP32DLL.DLL" (ByVal site_key\$)</code>
C#.NET	<code>CKKeyErrorEnum SaveSiteKey (String *pstrSiteKey);</code>

PARAMETERS

Site_key

This is a pointer to the string containing the user-entered site key. Spaces are removed from the key by this function before it is processed.

Return Value

Table 33: SaveSiteKey() Values

Name	Value	Message	Solution
SITE_KEY_OK	0	The function completed successfully.	
SITE_KEY_ENTRY_CHECK_FAIL	-1	The user is likely trying to reuse an old key.	This error occurs if the user attempts to re-enter a site key from a previous Site Code. The user must give you the site code they currently have and must enter the Site Key you create for that site code.
SITE_KEY_ENTRY_CRC_FAIL	-2	The key has likely been mistyped.	This error occurs only if the site key the user has entered is incorrect. The most common mistake is to enter a "1" rather than an "l".
SITE_KEY_FILEWRITE_FAILURE	-3	The key file could not be written (disk may be full).	This error occurs if there are no read and write privileges to the application directory. If you continue to receive this error, please contact support@crypkey.com.

SITE_KEY_NO_LIC NSE_TO_ADD_TO	-4	No license exists to which this license can be added.	This can occur if the customer's license expires before he has he used his update key. The customer needs a new authorization.
SITE_KEY_WRONG_ ADD_ON_TYPE	-5	Cannot add licenses of different types. Days vs Runs.	Ensure that add-ons to the customer's license have the same restriction type that was present in the original license. For example, you can't add days to a runs-restricted license.

General error values also apply — see the section *General Error Values* on page 167

Note

See *Chapter 5: Demo Programs*, Basic Programming Steps for more information.

GetSiteCode() must be called before the SaveSiteKey() function is called. We recommend that you always show the Site Code to the user before or while allowing the user to enter the Site Key.

The program should always call GetAuthorization() to find out the change in the state of the license immediately after this function.

SetNetHandle()

The SetNetHandle() function sets the network license handle (i.e. passes the handle of a SLAPI network resource to the program).

FUNCTION USAGE

If you want several SLAPI-licensed programs sharing a single network seat, you can use GetNetHandle() to get the handle and then distribute it using the SetNetHandle() function. The SetNetHandle() function should always be called after the call to InitCrypkey() is made and before the GetAuthorization() call is made.

PROTOTYPES

Language	Syntax
C	void SetNetHandle (unsigned short

	<code>net_handle);</code>
C++	<code>long SetNetHandle (long nNetHandle);</code>
VB	<code>SetNetHandle (nNetHandle as Long) as Long</code>
C#.NET	<code>void SetNetHandle (int nNetHandle);</code>

PARAMETER

Net_handle

Net_handle is a handle to a network resource obtained by calling `GetNetHandle()`.

Note

Only one of the programs sharing the network seat can call `EndCrypkey()` successfully — preferably, the last program to stop executing calls this function, since this means that the network seat is clearly no longer required.

SetCustomInfoBytes()

DESCRIPTION

This function sets data to file that can only be read by the `GetCustomInfoBytes()` function.

FUNCTION USAGE

GetCustomInfoBytes() and *SetCustomInfoBytes()* can be used by your application to save data in a safe place. The data is encrypted before being written to the hard drive, and `CrypKey` will insure the file is not compromised by an attacker replacing it with a previous version of this file.

Use this function to store sensitive data, such as licensing data, that needs to be protected from being altered by users.

DECLARATION (C/C++)

```
int SetCustomInfoBytes(unsigned char * bpDataArray);
```

DECLARATION (VISUAL BASIC)

Not available in VB.

PARAMETERS

bpdataArray

Pointer to a MAX_CUSTOM_BYTES byte buffer containing data you wish to save. The function will write the data from the buffer into an encrypted file.

RETURN VALUES

Value	Error message
0	The function performed successfully.
-2	Could not open Custom Info Byte file.

TransferIn()

This function is part of the following sequence of functions used to transfer a license using some sort of media such as floppy or USB drive:

Step 1. *RegisterTransfer()* - performed by the instance of the application that is at the target location of the license to be transferred.

Step 2. *TransferOut()* - performed by the instance of the application that is at the source location of the license to be transferred.

Step 3. *TransferIn()* - performed by the instance of the application that is at the target location of the license to be transferred.

Called from an application that does not have authorization, this function will import another authorization license in by reading the transfer files on a floppy disk or in a directory. To be a valid transfer, the site calling this function must have begun the process by using the RegisterTransfer() function.

FUNCTION USAGE

The directory for transfer most likely will be A:\, as the transfer is usually from one computer to another using a floppy disk. However, this transfer could be from one directory to another on the same computer, or even from one network drive to another. For this reason, the user must be able to enter this parameter.

PROTOTYPES

Language	Syntax
C	<code>int TransferIn (char *directory);</code>
C++	<code>int TransferIn (char *directory);</code>
VB	<code>Declare Function TransferIn& Lib "CRP32DLL.DLL" (ByVal directory\$)</code>
C#.NET	<code>CKInErrorEnum TransferIn (String *pstrDirectory);</code>

PARAMETER

Directory

This is a pointer to a null terminated text string that contains the path of the directory from where the license files will be imported.

Return Value

Table 34: TransferIn() Values

Name	Value	Message	Solution
TI_OK	0	The function completed successfully.	
TI_THIS_ALREADY_AUTHORIZED	-1	A transfer in is allowed by an unauthorized program only.	This error occurs if you try to transfer a license into an application that already has authorization (including a RTT authorization). You can only transfer a license into an unauthorized application. A trial license is considered a license, so you must to kill your trial

Name	Value	Message	Solution
			license before you can transfer in a full authorization.
TI_HARDDISK_REGFILE _NOT_FOUND	-2	Registration file in the same directory as the application is missing.	If you receive this error, your system is not able to find the .reg file needed to perform the transfer. You must have completed the RegisterTransfer() and the TransferOut() functions successfully before you can perform the TransferIn() function.
TI_HARDDISK_REGFILE _CRC_FAILURE	-3	Registration file in the same directory as the application has been damaged or tampered with.	If you receive this error, the .reg file is corrupted. The only way to solve this problem is to delete the .reg file you have and try the transfer process again. If the problem persists, you must request authorization.
TI_REGFILE_NOT_FOUN D	-4	Registration file not found in the given direc-tory.	This error occurs if the .reg file is not found in the directory specified by the transfer function. This file must be in the correct directory to complete the transfer. The RegisterTransfer() and the TransferOut() functions must be completed successfully before you can perform the

Name	Value	Message	Solution
			TransferIn() function.
TI_REGFILE_CRC_FAILURE	-5	Registration file in the given directory has been damaged or tampered with.	If you receive this error, the .reg file is corrupted. The only way to solve this problem is to delete the .reg file you have and try the transfer process again. If the problem persists, you must request authorization.
TI_HARDDISK_REGFILE_MOVED	-6	Registration file in the same directory as the application has been moved.	This error occurs if the .reg file is not found in the directory specified by the transfer function. This file must be in the correct directory to complete the transfer. The RegisterTransfer() and the TransferOut() functions must be completed successfully before you can perform the TransferIn() function.
TI_REG_FILES_DONT_MATCH	-7	A copy of the transfer files from a previous license transfer is probably being used.	This error occurs if the registration file in the directory to which you are trying to transfer is different than the registration information on the floppy disk you are using to transfer.
TI_SOURCE_HAS_NO_LICENSE	-8	The transfer out operation has not been done yet.	In order to complete the TransferIn() function, you must first complete the

Name	Value	Message	Solution
			RegisterTransfer() and the TransferOut() functions.
TI_SITEKEYFILE_NOT_FOUND	-9	The Site Key file was not found in the given directory.	If you receive this error, please contact support@crypkey.com.
TI_DIFFERENT_SITEKEY	-10	The Site Key file or registration file has been damaged or tampered with.	If you receive this error, the .reg file is corrupted. The only way to solve this problem is to delete the .reg file you have and try the transfer process again. If the problem persists, you must request authorization.
TI_COULDNOT_WRITE_SITEKEYFILE	-11	A write error has occurred in the same directory as the application.	If you receive this error, please contact support@crypkey.com.
TI_RSTKEYFILE_NOT_FOUND	-12	The restriction file was not found in the given directory.	This error occurs if the .rst file is not in the directory specified by the transfer function. This file must be in the correct directory to complete the transfer. In order to complete the TransferIn() function, you must first complete the RegisterTransfer() and the TransferOut() functions.
TI_DIFFERENT_RSTFILE	-13	The restriction file or the registration file has been damaged or	If you receive this error, the .rst file is corrupted. The only

Name	Value	Message	Solution
		tampered with.	way to solve this problem is to delete the .rst file you have and try the transfer process again. If the problem persists, you must request authorization.
TI_COULDNOT_WRITE_RSTKEYFILE	-14	A write error has occurred in the same directory as the application.	This error occurs if you do not have the ability to read and write to the directory. If you continue to receive this error, please contact support@crypkey.com.

General error values also apply — see the section *General Error Values* on page 167.

Notes

See *Chapter 10: Moving Protected Programs* for more information.

TransferOut()

This function is part of the following sequence of functions used to transfer a license using some sort of media such as floppy or USB drive:

Step 1. *RegisterTransfer()* - performed by the instance of the application that is at the target location of the license to be transferred.

Step 2. *TransferOut()* - performed by the instance of the application that is at the source location of the license to be transferred.

Step 3. *TransferIn()* - performed by the instance of the application that is at the target location of the license to be transferred..

Called from an application that does have authorization, this function will export the application's authorization license out by placing transfer files on a floppy disk or in a directory. These transfer files must be used by the same application that created the initial registration files to transfer the license in..

FUNCTION USAGE

The directory for transfer most likely will be A:\, as the transfer is usually from one computer to another using a floppy disk. However, this transfer could be from one directory to another on the same computer, or even from one network drive to another. For this reason, the user must be able to enter this parameter.

PROTOTYPES

Language	Syntax
C	<code>int TransferOut (char *directory);</code>
C++	<code>long TransferOut (BSTR bstrDirectory);</code>
VB	<code>Declare Function TransferOut& Lib "CRP32DLL.DLL" (ByVal directory\$)</code>
C#.NET	<code>CKOutErrorEnum TransferOut (String *pstrDirectory);</code>

PARAMETER

Directory

This is a pointer to a null-terminated text string that contains the path of the directory to which the license files will be exported.

Return Value

Table 35: TransferOut() Values

Name	Value	Message	Solution
TO_OK	0	The function completed successfully.	
TO_THIS_NOT_AUTHORIZED	-1	The application must be authorized in order to do a transfer out.	This error occurs if you try to transfer a license out from an application that does not contain authorization. You can only transfer a license out of an authorized

Name	Value	Message	Solution
			application.
TO_REGFILE_NOT_FOUND	-2	No registration file was found in the given directory.	If you receive this error, your system cannot find the .reg file that is needed to do the transfer. You must complete the RegisterTransfer() function successfully before you able perform the TransferOut() function.
TO_REGFILE_CRC_FAILURE	-3	The registration file has been damaged or tampered with.	If you receive this error, the .reg is corrupted. The only way to solve this problem is to delete the .reg file you have and try the transfer process again. If the problem persists, you must request authorization.
TO_DIFFERENT_APPLICATION	-4	The transfer is being attempted for a different application.	This error occurs if you try to transfer a license out from a different application than the RegisterTransfer() function has recorded in its license information. You can only transfer a license from the same application into which you are transferring.
TO_TARGET_ALREADY_HAS_LICENSE	-5	The transfer files are already in the given directory.	This error occurs if the TransferOut() function has already been performed once. Please ensure that you only attempt to transfer the license out of the application once.
TO_SITEKEYFILE_NOT_FOUND	-6	The key file in the given directory is	If you receive this message, please

Name	Value	Message	Solution
		missing.	contact support@crypkey.com.
TO_COULDNOT_WRITE_SITE KEYFILE	-7	The key file could not be written to the given directory.	This error occurs if you do not have read and write access to the specific directory. If you continue to receive this error, please contact support@crypkey.com.
TO_RSTKEYFILE_NOT_FOUND	-8	The restriction file in the given directory is missing.	If this error occurs, ensure you have a file in your directory with the extension .rst. If the file is not there, try to run the application again. This process should create the file. If this does not work, you must request authorization.
TO_COULDNOT_WRITE_RST KEYFILE	-9	The restriction file could not be written to the given directory.	This error occurs if you do not have read and write access to the specific directory. If you continue to receive this error, please contact support@crypkey.com.
TO_COULDNOT_WRITE_REGKEYFILE	-10	The registration file could not be written to the given directory.	This error occurs if you do not have read and write access to the specific directory. If you continue to receive this error, please contact support@crypkey.com.
TO_SOURCE_WRITE_PROTECTED	-11	The source directory is write protected.	This error occurs if you do not have read and write access to the specific directory. If you continue to receive this error, please contact support@crypkey.com.

General error values also apply — see the section *General Error Values* on page 167.

Note

See *Chapter 10: Moving Protected Programs* for more information.

Appendix A: Quick Reference

This Appendix contains brief descriptions of CrypKey functions and their C prototypes. See also Chapter 14: Function Reference.

AcquireLicense()

allows an unlicensed CrypKey protected program to acquire a license from a licensed copy.

CKTimeString()

Translates, into a text string, the ULONG time values returned by the functions GetAuthorization() and FloatingLicenseSnapshot().

CKFormatWithSpaces()

This function translates an input string into a format where . For example “HELLOWORLD” generates an output of “HELL OWOR LD”. This is used for formatting site keys and site codes into more human readable forms to display to a user.

CrypKeyVersion()

returns the version number of the CrypKey system currently in use.

CKSystemTime()

This function retrieves the local system time, in seconds since January 1st, 1970, from the License Service. You can use CKTimeString to format it in human readable form.

DirectTransfer()

Called from an application that has authorization, this function transfers the application’s authorization license to another directory.

CrypkeyBuildNum()

This function returns the build number of the CrypKey system engine currently in use.

EndCrypkey()

Notifies CrypKey that the program is being terminated. The function should always be called when the program is terminated.

CKChallenge32()

Verifies that the CryKey DLL has not been replaced with an impostor DLL.

ExplainErr()

Returns a text string explaining the return code of various functions.

ExplainErr2()

Returns a text string explaining the return code of various functions. This function is similar to the Explain Err() function, but your error message is written into your string text.

FloatingLicenseSnapshot()

Returns a snapshot of all users holding or waiting for a network floating license.

FloatingLicenseGetRecord()

Used only in COM Objects and C#.NET Assemblies: called from an application that has authorization, this function returns a single record of the current floating license status.

[Prototypes available for C++ and C#.NET but not for C or VB.]

FloatingLicenseTakeSnapshot()

Used only in COM Objects and C#.NET Assemblies: returns a snapshot of all users holding or waiting for a network floating license.

[Prototypes available for C++ and C#.NET but not for C or VB — see Chapter 14: Function Reference, FloatingLicenseSnapshot.]

Get1RestInfo()

Used to return one of three different types of data depending on the value of the “which” input parameter.

GetAuthorization()

Gets the level at which the program is authorized to run or an authorization failure code. Also used to decrement the number of uses count. Must be called, together with InitCrypKey, before any other CrypKey function.

GetAuthorization2()

Determines if the program is authorized to run, and decrements the license count — but does not obtain option and level information.

GetDrivePermanentSerialData()

proprietary CrypKey software function that accesses the following hard drive information on a customer’s computer: model number, serial number, firmware data.

GetLastError()

Returns the text for the last error reported.

[Prototypes available for C++, VB and C#.NET but not for C.]

GetLastErrorCode()

C#.NET function only: returns the number of the last error reported.

GetLevel()

Used to determine the level at which your program has been authorized to run.

GetNetHandle()

Gets and returns the network license handle of the protected program.

GetCustomInforBytes()

This function gets data that was written to file by the SiteCustomInfoBytes() functions.

GetDrivePermanentSerialData()

This function returns the system burned in serial and manufacturing data.

GetNumMultiUsers()

Gets the number of multi-users this site has been granted.

GetLibraryType()

This function returns a number that indicates the type of CrypKey client library currently in use.

GetLibraryVersion()

This function returns the version number of the CrypKey client library linked to your program.

GetLicenseInfo()

This function is new for CrypKey 7, it returns the information about the current operating license mode and location.

GetOption()

Used to determine the status of a specific option.

GetRestrictionInfo()

Gets information on any restrictions that are present in the license.

GetSiteCode()

Gets the site code for this program location. The site code must be reported by the customer before the developer can issue the site key.

GetSiteCode2()

Returns a pointer to the site code for this program location.

GetNumCopies()

Gets the number of license copies that the

GetPCSerial()

This function returns the computer bios serial data.

InitCrypkey()

Initializes CrypKey with runtime information. This function and GetAuthorization() must be called before any other function in CrypKey.

IsEasyLicense()

This function indicates if the current license is an EasyLicense.

KillLicense()

Disables an existing license and provides a confirmation code that can be entered into the Site Key Generator in order to display the details of the de-authorization.

ReadyToTry()

Implements a one-time automatic license based on a “days” time period, without version control.

ReadyToTryDays()

Implements a one-time automatic license based on a “days” time period, with version control.

ReadyToTry_Runs()

Implements a one-time automatic license based on a “runs” restriction with version control.

RegisterTransfer()

Called from an application that does not yet have authorization, this function registers the application by placing license imprint files on a disk or in a directory.

current site has been granted.

SaveSiteKey()

This function is used to save to file the site key that has been acquired from the user. The key is checked before it is saved.

SetNetHandle()

Sets the network license handle, i.e. passes the handle of a SLAPI network resource to the program.

```
void SetNetHandle (unsigned short  
net_handle);
```

TransferIn()

Called from an application that does not have authorization, this function transfers the application's authorization license in by reading transfer out license imprint files on a disk or in a directory.

TransferOut()

Called from an application that has authorization, this function transfers the application's authorization license out by placing license imprint files on a disk or in a directory.

```
int TransferOut(char far *directory);
```

Glossary

This glossary contains definitions of terms used in this manual.

These definitions help you understand the CrypKey SDK 7 product and its protection strategies.

.NET

.NET is a framework for objects and how they work together. .NET objects are language-neutral and may be written in C#, C++, Visual Basic, J #, or other languages. The code generated for .NET is an intermediate language that needs the Common Language Runtime (CLR) to interpret the code. The use of the CLR allows .NET code to run on different operating systems and platforms without having to recompile the code.

.NET Assembly

A .NET assembly is a file containing .NET classes. The assembly contains class definitions that allow other .NET programs to identify and use the classes within it.

Automatic License

This is a license that is created by using the ReadyToTry() function or one of its variants. Such licenses are created automatically by the software without requiring you or your distributor to provide a site key.

Common Object Model (COM)

The Common Object Model is a specification for objects and how they work together. COM objects are language-independent and may be written in C++, Visual Basic, Java, or other languages. The code generated for COM is standard Windows machine executable binaries.

COM Object

A COM object is a file containing COM classes. The COM object contains class definitions that allow other COM programs to identify and use the classes within it.

Company Number

This is one of two special keys used in the CKChallenge32() function (the other key is the password number). The company number key is based on a private number that CrypKey assigns to your company when the developer keys are requested.

Developer Keys

These are the four keys that CrypKey issues you after receiving payment. These include:

- company number
- password number
- Master Key
- User Key

Fixed License

A product license usable only on the computer where it was installed (node-locked license). Although the program can be copied to other computers, its inherent security ensures it can only run on the computer where it was installed.

Floating/Networked License

This is the number of concurrent seats that can run on the network.

Level

A parameter defined for a product license. The Site Key Generator communicates the level to your product via the site key, enabling you to control how your product runs and which instance of your product is authorized. CrypKey SDK 7 communicates only the level value to the protected application; you must implement the action(s) within the product based on the level.

License

A protected product's authorization is defined by the license. The license contains details of the authorization, including levels, options, license type, and license

restriction. In physical terms, the license consists of four hidden system files that reside in the same directory location as the protected program.

Master Key

A hexadecimal number issued by CrypKey, based on your company number and product name. To use CrypKey with your product, you need a Master Key and User Key (these keys are entered into the InitCrypkey() function).

Option

Selection of a specification that may include or exclude a component as part of a product license. Together, the numeric symbols for the options and levels defined for your product comprise a 32-bit number used as part of the site key.

Password Number

One of two special keys used in the CKChallenge32() function when using the CrypKey API (the other key is the company number). The password number is based on the product password, which you specify. CrypKey creates the password number from your product password and provides it to you.

Site

In software licensing, the internal boot device (hard drive) of the computer where a protected program is installed. The term is generally synonymous with the computer itself.

Site Code

A site-specific hexadecimal number issued by a protected product when installed on a client's computer. The site code is provided by the GetSiteCode() function. Your customers provide the site code when requesting licensing changes for the product.

Site Key

A hexadecimal number based on a client's site code, created by via the Site Key Generator. The site key unlocks client software features and license restrictions.

User Key

A hexadecimal number issued by CrypKey, based on your product password. To use CrypKey with your product, you need a Master Key and a User Key (enter these keys in the InitCrypkey() function).

Index

This index includes page references for significant occurrences of Casper topics and terms in this manual.

- .NET assembly 32, 37, 93, 171
- .ngn file 85, 168
- AcquireLicense 21, 163, 244
- anti-hacking 3
- authorization transfer 88
- Authorizing
 - program 88
- Automatic License 166, 248
- automatic licensing 220
- CKChallenge32..... 163, 176
- cks.exe 44, 47, 103, 107
- clock manipulation 64
- CloneBuster 4, 64, 65
- COM object..... 32, 90, 202
 - in C++..... 97
- COM Object..... 37, 93, 186
 - constants 99
 - installation..... 94
 - referencing..... 94, 96, 97
 - referencing by name..... 95
- company number..... 92
- Company Number..... 9, 249
- crp32001.ngn 130
- CRP32001.NGN 90
- CrypKey Business Contact
 - information 13
- CrypKey Instant 11, 15
- CrypKey SDK..... 28
 - features..... 2
 - installation..... 18, 28
 - new features 4
 - system requirements..... 12
 - upgrade process 15
- crypkey.h..... 32, 89
- CrypkeyVersion..... 164, 178
- DAL See Distributor Authorizing License
- developer keys
 - .CED file..... 130
 - and authorized license.... 9
 - and secret numbers 92
 - examples for testing..... 10
 - temporary..... 11
 - types 8
- development
 - and testing keys 10
 - platforms 13
 - schedule 9
 - tools 92
- direct transfer 64

- Direct Transfer 125
- DirectTransfer ... 126, 163, 164, 171
- Distributor Authorizing License 84
- distributors
 - copyright protection 11
 - third party..... 2
- driver
 - types 34
 - uninstall..... 105
- Dynamic Link Libraries..... See DLLs
- EndCrypKey 87, 164, 181, 205, 233
- error codes..... 99, 103
- errors
 - 200 to -299 range 168
 - and EndCrypKey function 87
 - and pre-defininf DLL functions 96
 - contact CrypKey support 43
 - general..... 167
 - required file 85
- example.exe*..... 10, 13, 34
- ExplainErr 85, 164, 167, 184
- ExplainErr2 164
- failure code 85
- features 11, 32, 39, 49, 62
 - additional 62
 - and levles..... 63
 - and options 63
 - anti-tampering..... 117
 - EasyLicense 65
 - security..... 110
 - unlock..... 250
- FloatingLicenseSnapshot.. 101, 163, 164, 177, 187
- FloatingLicenseSnapshot2. 190
- Floppy Transfer 125
- functions..... 127
 - Action menu 138
 - summary 163
 - syntax..... 163
- GetAuthorization 51, 164
 - call 118
 - faux compare 118
 - mandatory call..... 86
 - positive value 86
 - priority of call 119
 - returns 0 86
 - returns negative value... 86
- GetAuthorization2..... 200
- GetDrivePermanentSerialData 165, 202
- GetLevel..... 165, 203
- GetNetHandle 165, 204
- GetNumCopies..... 165, 205
- GetNumMultiUsers..... 166, 206
- GetOption 166, 206
- GetRestrictionInfo..... 166, 207
- GetSiteCode..... 51, 88, 166
- GetSiteCode2..... 166
- hard drives
 - cloning..... 65
 - serial numbers..... 2
- InitCrypKey 86, 87

priority of call	199	definition	250
InitCrypkey()		encrypt	118
mandatory call	119	temporary	10
installing		network	
license on server	131	administrator	40, 48
method	21	concurrent users	57
Installing		configure licenses	40
a COM Object	94	configure SKG	77
procedure	23	drive share mode	17
utilities	40	installations	35
keys		licensing	39
developer	249	transfer licenses	39
encryption	3	users	12
for InitCrypkey() function		view license usage	40
.....	250	options	11, 62, 63
Master and User	8	transfer license	21
temporary	8	password	
used in CKChallenge()		and User Key	9
function	249	for product	10
KillLicense	166	from CrypKey	24
levels	62	number	9, 93, 250
change for your customer		secret numbers	92
.....	79	Readme file	31
choose features	63	README.hlp	33
defined in licenses	249	ReadyToTry	166
increase or decrease	74	ReadyToTryDays	166, 224
of privileges	2	ReadyToTryRuns	166, 226
range	62	RegisterTransfer ..	88, 126, 167
libraries		run modes	
32-bit	35	silent and verbose	105
link to application	89	SaveSiteKey	54
location	35	SDK	
new in CrypKey 7	131	new features	4
runtime	91	standard features	2
static	13	SetNetHandle	167, 204, 232
Win32	36	setupex.exe	19, 47, 103
license			
fixed	21		
floating	39		
temporary	9		
trial period	8		
Master Key	250		

- setupe.xco*..... 103
- silent mode..... 105
- Site Code 9
- Site Key..... 9, 250
- software protection..... 64
- Stealth..... 3, 34, 109
 - configuration 116
 - directory 111
- support
 - issues..... 2
 - platforms 13
 - technical..... 1
 - website..... 13
- system requirements..... 1, 12
- tampering
 - anti 117
 - file 126
 - product..... 117
- technical support..... 13
- testing
 - SDK..... 23
- TransferIn..... 89, 127
- TransferOut..... 89, 126
- trial period 3
- unlimited
 - number of redundant
 - servers 39
 - usage option 57
- upgrade..... 15
 - from current version 33
 - process 15
 - with SDK 18
- User Key
 - encrypt 118
 - temporary 10
- verbose mode 105
- web site
 - CrypKey 1
- Windows Program Manager. 49