

Sentinel™ UltraPro™ 1.0

Developer's Guide



Copyright © 2004, SafeNet, Inc. (Baltimore)
All rights reserved.

All attempts have been made to make the information in this document complete and accurate. SafeNet, Inc. is not responsible for any direct or indirect damages or loss of business resulting from inaccuracies or omissions. The specifications contained in this document are subject to change without notice.

SafeNet, Sentinel, SuperPro, and UltraPro are either registered trademarks or trademarks of SafeNet, Inc. Microsoft, Windows, Windows 95, Windows 98, Windows ME, Windows NT, Windows 2000, Windows XP, Windows Server 2003, and Internet Explorer are either trademarks or registered trademarks of Microsoft Corporation in the United States and other countries. Java is a trademark of Sun Microsystems, Inc. in the United States and other countries. All other trademarks are the property of their respective owners.

CONFIDENTIAL INFORMATION

Sentinel UltraPro is designed to protect your software applications from unauthorized use. The less information that unauthorized people have regarding your security system, the greater your protection. It is in your best interest to protect the information herein from access by unauthorized individuals.

Part Number 000015-001, Revision A
Software versions 1.0 and later

Revision	Action/Change	Date
A	Initial Release	September 2004

SafeNet Sales Offices

Australia +61 3 9882 8322	Brazil +55 11 6121 6455	China +86 10 8266 3936	Finland +358 20 500 7800
France +33 1 41 43 29 00	Germany +49 18 03 72 46 26 9	Hong Kong +852 3157 7111	India +91 11 26917538
Japan +81 3 5719 2731	Japan (Tokyo) +81 3 5719 2731	Korea +82 31 705 8212	Mexico +52 55 5575 1441
Netherlands +31 73 658 1900	Singapore +65 6297 6196	Taiwan 886-2-27353736	UK +44 1932 579200
UK (Chertsey) +44 1932 579 200	U.S. (Massachusetts) +1 978.539.4800	U.S. (New Jersey) +1 201.333.3400	U.S. (Virginia) +1 703.279.4500
U.S. (Irvine, California) +1 949.450.7300	U.S. (Santa Clara, California) +1 408.855.6000	U.S. (Torrance, California) +1 310.533.8100	

International Quality Standard Certification



The Beijing, China; Irvine, California, U.S.A; and Rotterdam, The Netherlands facilities are certified to the latest, globally-recognized ISO 9001:2000 standard. The certificate number is: CERT-02982-2003-AQ-HOU-RAB Rev 3.

European Community Directive Conformance Statement



This product is in conformity with the protection requirements of EC Council Directive 89/336/EEC. Conformity is declared to the following applicable standards for electro-magnetic compatibility immunity and susceptibility; CISPR22 and IEC801. This product satisfies the CLASS B limits of EN 55022.

FCC Notice to Users



Sentinel UltraPro has passed the FCC Self-authorization process of Computers and Computer Peripherals. FCC Part 15 Class B Specifications.

This equipment has been tested and found to comply with the limits for a class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

In order to maintain compliance with FCC regulations, shielded cables must be used with this equipment. Operation with non-approved equipment or unshielded cables is likely to result in interference to radio and TV reception. The user is cautioned that changes and modifications made to the equipment without the approval of manufacturer could void the user's authority to operate this equipment.

Acknowledgments

Sentinel UltraPro makes use of certain third-party software. Please refer to *Acknowledgments.pdf* under the *Manuals* directory for details.

Contents

Preface xi

Welcome to Sentinel UltraPro!	xi
Where to Find What You Need	xi
Conventions Used in This Guide	xiii
Technical Support	xiv
Export Considerations	xvi
We Welcome Your Comments On Documentation.....	xvi

Part 1: Sentinel UltraPro Basics 1

Chapter 1 – Protecting Software With Sentinel UltraPro..... 3

A Reality Check on Software Piracy.....	3
Sentinel UltraPro For Marketing Success.....	4
How Does Sentinel UltraPro Secure Your Software?	5
Abundant Ways to Market Your Product	8
Security to Count On	10
Robust and Convenient Licensing Solutions	13
Streamlined Project Management.....	14
Licensing and Your Channels of Distribution	15
Flexibility that Your Customers Prefer	16

Chapter 2 – The Sentinel UltraPro Components 17

Sentinel UltraPro Key	18
Sentinel Driver	18
Sentinel Protection Server.....	18

Sentinel UltraPro Toolkit.....	21
Sentinel UltraPro API.....	23
The Protection Methods - API Elements and Shell.....	24
Sentinel Protection Installer.....	24
The Field Activation Methods.....	24
Chapter 3 – The Sentinel UltraPro Toolkit.....	25
The Toolkit Tools.....	25
The Key Status Pane.....	28
About Projects, Designs, and Elements	29
Chapter 4 – The Sentinel UltraPro Key	33
The UltraPro Key Terminology	33
The UltraPro Key Features	36
Attaching a Key.....	37
Special Cases for Attaching Keys.....	38
Ordering and Returning Keys	41
Part 2: Designing and Implementing Protection	43
Chapter 5 – Planning Application Protection	45
Factors You Must Consider Before Protecting.....	45
Shell Versus API Elements.....	46
Which API Element to Choose?	48
Guarding Applications Against Time Tampering	56
Allowing Field Activation Or Not?.....	59
Which Access Mode To Use?	60
How to Set an Access Mode?	64
Specifying a User Limit	67
License Sharing Behavior.....	70
Whether Authorizing Distributors or Not?.....	71

Chapter 6 – Using Shell and Quick Shell.....	73
The Shell Methodology	73
Shell Versus Quick Shell	75
Using Quick Shell	76
Using Shell.....	79
Where to Go Next?.....	86
 Chapter 7 – Using API Elements	 89
About the API Elements-Based Protection	89
Adding Full License	91
Adding Short License	94
Adding String	96
Adding Boolean	97
Adding Counter	98
Adding Integer	100
Where to Go Next?.....	102
 Chapter 8 – Designing the Field Activation Strategy.....	 103
What Is Field Activation?	103
Client Activator Versus Field Activation Utility	107
Generating the Field Activation Objects.....	110
The Field Activation Actions.....	111
 Chapter 9 – Tips on Maximizing Application Security	 117
Basic Types of Attacks	118
Geared to Provide Unique Solutions.....	119
Tips and Tricks	119
 Chapter 10 – Implementing the Application Protection.....	 129
Prototyping Design Using the Build Button	130
Viewing Design in Key Layout	133
Final Steps of Implementing Protection	135

Part 3: Programming Keys and Generating Licenses . 137

Chapter 11 – Programming Product Keys 139

Setting Up to Program Product Keys.....	139
Programming a Product Key.....	142
Where to Go Next?.....	147

Chapter 12 – Programming Distributor Keys..... 149

Adding Value Through Distributors.....	149
Using the Distributor Keys Option	152
Updating Distributor Keys in the Field	154
Where to Go Next?.....	155

Chapter 13 – Generating Licenses And Updating Keys 157

Receiving the Locking Code from Your Customer.....	157
Generating a License Code	158
Sending the License Code to Your Customer.....	159

Part 4: Deployment 161

Chapter 14 – Deploying Protected Applications..... 163

Redistributables for Customers And Distributors	163
Deploying the Sentinel Driver	165
Deploying the Sentinel Protection Server.....	166
Deploying on Novell NetWare Platforms	167
Deploying the Configuration File.....	170
Deploying the Field Activation Utility	171
Deploying the Sentinel Client Activator	172
Deploying the Data Protection Driver	173
Deploying the Key Manager Application	174
Deploying the System Administrator’s Help	175

Appendix A – Glossary..... 177

Index 197

Preface

Welcome to Sentinel UltraPro!

Thank you for choosing Sentinel UltraPro, a software protection system that uses the Sentinel UltraPro hardware key to:

- Protect your applications from piracy.
- Implement different license models for different customers.
- Secure your product revenue.

Where to Find What You Need

This guide is a complete reference for Sentinel UltraPro version 1.0 or later. It is primarily aimed at the following categories of a developer¹:

You Could Be a...	Recommended References
Manager or New User You want to understand the features and capabilities of Sentinel UltraPro.	<input type="checkbox"/> Part I of this guide <input type="checkbox"/> Release notes

1. We use the term *developer* throughout our documentation to denote a software development company or individuals who use Sentinel UltraPro keys to protect their applications.


You Could Be a...	Recommended References
Application Programmer You want to plan, design, and implement the application protection.	<ul style="list-style-type: none">❑ Release notes (installation and feature summary)❑ Toolkit Help❑ Part I and Part II of this guide
Key Programmer You want to program the hardware keys for your customers and distributors.	<ul style="list-style-type: none">❑ Part III of this guide❑ Toolkit Help
Builder You want to prepare the product for release, including deploying the redistributables.	<ul style="list-style-type: none">❑ Part IV of the guide❑ Sentinel Protection Installer Help
Sales Distributors You want to generate license codes and program hardware keys for your customers.	<ul style="list-style-type: none">❑ Help included with the Key Manager application
Customers and Helpdesk For users who want to learn how to use the hardware key and the redistributables, like the Sentinel Driver, Sentinel Protection Server, configuration file.	<ul style="list-style-type: none">❑ System Administrator Help

Tip: Try video tutorials on the Help menu!

For Windows platforms, we have included video tutorials under the **Help** menu. You can use them for quick and interactive learning of the Toolkit.

Conventions Used in This Guide

Please note the following conventions used in this guide:

Convention	Meaning
<i>Courier</i>	Denotes syntax, prompts and code examples. Bold Courier type represents characters that you type; for example: logon .
Bold Lettering	Words in boldface type represent keystrokes, menu items, window names or fields.
<i>Italic Lettering</i>	Words in italic type represent file names and directory names; it is also used for emphasis.
	Denotes a warning. This icon flags any potential pitfalls that we think that you may need to be careful of.
<OS drive>	The root drive on your system where your operating system is installed.
<installdir>	The path where the software, in context, is installed.
<Personal folder>	<ul style="list-style-type: none">❑ The default path for the Personal folder on Windows 2000/XP/Server 2003 systems is: <OS Drive>\Documents and Settings\user name❑ The default path for the Personal folder on a Windows NT system is: <OS Drive>\Winnt\Profiles\user name\Personal

Technical Support

We are committed to supporting Sentinel UltraPro. If you have questions, need additional assistance, or encounter a problem, please contact Technical Support using the information given below:

Technical Support Contact Information

Customer Connection Center (C³)	
<i>http://www.safenet-inc.com/support/index.asp</i>	
Americas	
Internet	<i>http://www.safenet-inc.com/support/index.asp</i>
E-mail	techsupport@safenet-inc.com
United States	
Telephone	(800) 959-9954
Fax	(949) 450-7450
Europe	
E-mail	EUTechSupport@safenet-inc.com
France	
Telephone	0825 341000
Fax	44 (0) 1932 570743
Germany	
Telephone	01803 (7246269)
Fax	44 (0) 1932 570743
United Kingdom	
Telephone	0870 7529200
Fax	44 (0) 1932 570743

Technical Support Contact Information (Continued)

Pacific Rim	
E-mail	techsupportpacrim@safenet-inc.com
Australia and New Zealand	
Telephone	(61) 3 9882 8322
Fax	(61) 3 9882 0588
China	
Telephone	(86) 10 8851 9191
Fax	(86) 10 6872 7342
India	
Telephone	(91) 11 2691 7538
Fax	(91) 11 2633 1555
Taiwan and Southeast Asia	
Telephone	(886) 2 27353736
Fax	(886) 2 27352383

Tip: Check our Web site for updates

Please visit <http://www.safenet-inc.com/support/index.asp> for the most up-to-date information about Sentinel UltraPro, downloads, FAQs and technical notes.

Export Considerations

We offer products that are based on encryption technology. The Bureau of Industry and Security (BIS) in the U.S. Department of Commerce administers the export controls on our commercial encryption products.

Rules governing exports of encryption can be found in the Export Administration Regulations (EAR), 15 CFR Parts 730-774, which implements the Export Administration Act (“EAA” 50 U.S.C. App. 2401 et seq.).

An Important Note

BIS requires that each entity exporting products be familiar with and comply with their obligations described in the Export Administration Regulations. Please note that the regulations are subject to change. We recommend that you obtain your own legal advice when attempting to export any product that uses encryption. In addition, some countries may restrict certain levels of encryption imported into their country. We recommend consulting legal counsel in the appropriate country or the applicable governmental agencies in the particular country.

We Welcome Your Comments On Documentation

To help us improve future versions of the documentation, we want to know about any corrections, clarifications or further information you would find useful. When you contact us, please include the following information:

- The title and version of the guide you are referring to.
- The version of the software you are using.
- Your name, company name, job title, phone number, and e-mail address.

Send us e-mail at: **techpubs@safenet-inc.com**

Part 1

Sentinel UltraPro Basics

- ❑ Introduction to Sentinel UltraPro
- ❑ Getting Familiar With the UltraPro Components
- ❑ About the Sentinel UltraPro Toolkit
- ❑ About the Sentinel UltraPro Key

Chapter 1

Protecting Software With Sentinel UltraPro

In this chapter we will assess how software piracy threatens your profits and understand how Sentinel UltraPro can curb widespread piracy and add value to your software distribution.

A Reality Check on Software Piracy

Software piracy hurts the bottom-line of your business. Every year a huge share of revenue is lost due to piracy—affecting your profits and research and development prospects.

Software piracy can occur in many forms, varying from malicious counterfeiting to violation of the license agreement by users who may be unaware they are doing so (for example, too many clients using the application at one time, unreported installations and exchange of software disks among peers).

Software protection not only effectively secures against piracy but can also enhance product versatility with flexible licensing models. You can use new avenues for distributing your applications and ultimately improve *return-on-investment*. Moreover, software protection must be simple to implement so your schedules are not burdened with lengthy training and programming time. Read on to know how Sentinel UltraPro can do all this and much more!

Sentinel UltraPro For Marketing Success

Sentinel UltraPro is latest in the family of Sentinel hardware keys—world's #1 security keys that protect your applications from unauthorized use.

You can use Sentinel UltraPro for preventing software piracy and boosting revenue by increasing the availability of your software to new marketing segments—that might be interested in buying selective/full features of your software at attractive prices. Using Sentinel UltraPro, you can:

- Lease the software for certain period and extend the lease later if desired.
- Provide perpetual license for your application without being concerned about the licensing violations, such as the number of users exceeds the number of licenses bought, unauthorized installations, and so on.
- Sell date-limited and feature-limited software for increasing the product usage/trial rate among the potential customers.
- Protect multiple applications and modules with a single key.
- Provide stand-alone and network licensing to customers with small setups or large enterprises.
- Activate and renew applications/features, increase demo limits, and convert demos to full versions remotely.

Note: What is a license?

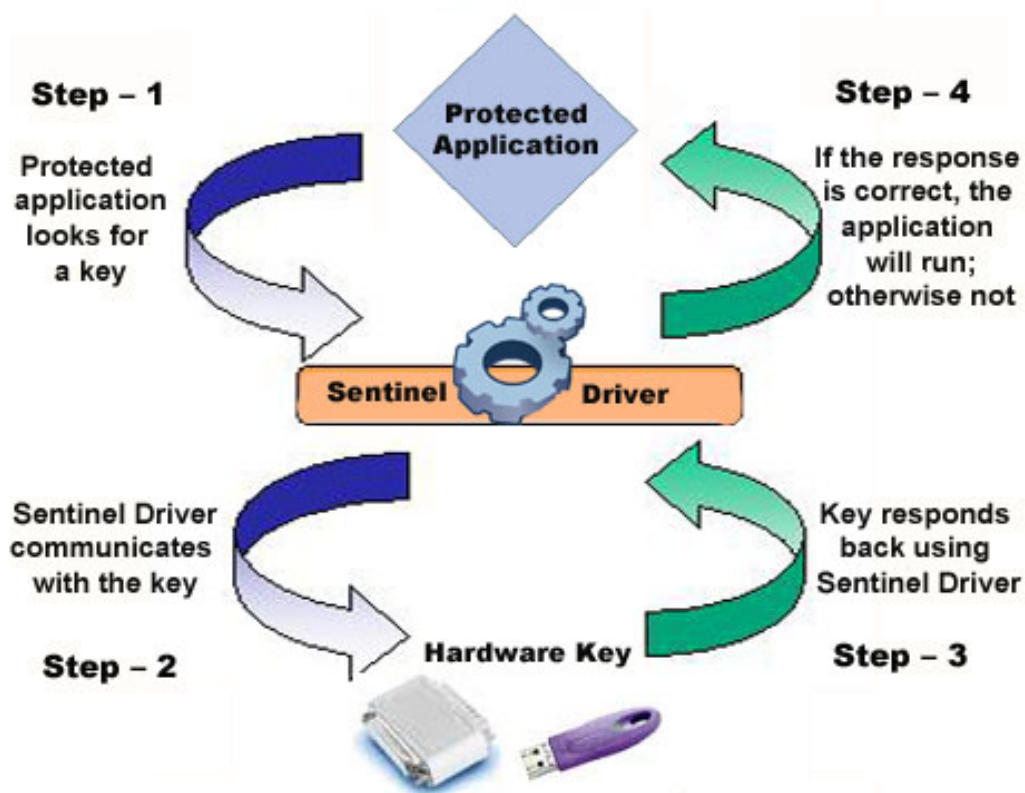
A license authorizes a user to access your application using the key. It represents the license agreement between you and your customer. Sentinel UltraPro ensures that the terms and conditions agreed upon by both the parties (whether the application will stop running after a specific number of days, it can only run on a specific computer, and so on) are met.

How Does Sentinel UltraPro Secure Your Software?

Sentinel UltraPro offers hardware-authenticated protection for your applications. The heart of the protection is contained in the encryption-based UltraPro key with highly customizable memory. The key can be programmed with several data elements, such as the encryption algorithms, activation passwords, integers, strings, Boolean flags and counters.

The following steps illustrate how it protects your software and revenue:

1. Design a protection strategy in the UltraPro Toolkit using the elements mentioned earlier.
2. Add the protection code to your application—either by directly adding the UltraPro API functions or by wrapping the Shell layer.
3. Program a key with the strategy you prepared. The key then becomes a necessary component for running your application.
4. On runtime, the application checks for the key's presence in order to run successfully. For example, by verifying some user data on the key. The Sentinel driver (an interface to the key) communicates with the key attached to a parallel/USB port (see the diagram page 6).
5. The key returns a response to the driver, which communicates back to your application. If a correct response is obtained, the user is permitted to run the application. If for any reason an expected response is not returned—may be because the key is not attached or has been tampered with—access to the application is denied. As a result:
 - ❑ The protected application can be copied by many users, but will not run beyond the number of users allowed.
 - ❑ Since the application can be programmed to check for the key periodically, it is impossible to remove the key while the application is running.

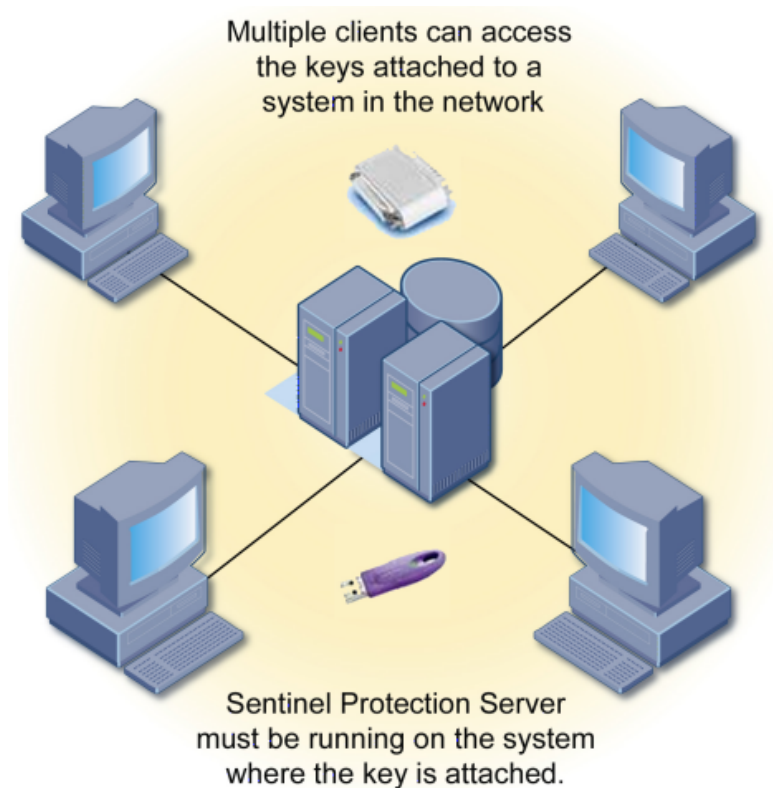


Communication Between the Application, Sentinel Driver, and Key

Network Applications Access a Key Concurrently

In the client-server environment, the application communicates with the key using the Sentinel Protection Server running on the system where the UltraPro key is attached (see the diagram page 7).

Typically, the license request is directed to a specific system or broadcasted within the subnet looking for a Sentinel Protection Server with an available license. The Sentinel Protection Server manages the communication between the application and key for obtaining, maintaining, and releasing a license.



A Single Key Can Serve Multiple Clients on the Network

Abundant Ways to Market Your Product

With Sentinel UltraPro's versatile features, you can securely market your product in alternative forms and charge accordingly. Using the flexible licensing solutions, you can expand your market along with successfully combating the software piracy.

A single application can be packaged in different ways to suit different requirements. For example, customer A may require a 30-day demo, while the customer B wants a perpetual license to run your application. Sentinel UltraPro offers you features using which you can create different licensing schemes with minimum efforts. Let's take a look at the features available:

- You can create time-limited and execution-limited versions of your application. This is especially useful in cases when you want to offer evaluation and *try-and-buy* versions of your software.
- Once the trial/demo application had expired, you can increase the demo limit or upgrade it to fully-licensed versions in the field—without having to ship a new hardware key or visit the customer's site. All this can be done using simple methods, like an e-mail, a fax, or a phone call. This process is known as *field activation*.
- The UltraPro keys are available for stand-alone and network environments. Each key contains a factory-programmed *hard limit* (ranging from 1, 2, 3, 5, 10, 25, 50 to unlimited) to restrict the number of users accessing the application. Typically, the stand-alone keys come with one hard-limit. If desired, you can program an intermittent *user limit* to set the number of users less than the hard limit.
- You can protect multiple applications using one key. The keys are available in different memory sizes, providing enough space to accommodate diverse security strategies, controlling several features and applications. A key can be programmed with a design.

Package Your Application in Different Forms

Demos, Leasing and Conversions

- ❑ Set expiration dates (For example, lease your application up to 31 March 2006)
- ❑ Count number of executions (For example, 30 free trials of a feature)
- ❑ Trial periods with a date (For example, 44 days trial software)



Bundle Application and Features

Vendors can create *package suites* of multiple applications protected with a single key.

Network Floating License Limits

Limit the number of users running the protected application in a network.

Named License Limits

Allow access to specific logins and users.



Site-based Licensing

Allow access to the application users within a network. Provide license monitoring features!

Share Licenses

Share licenses on the basis of user name, MAC address, or seat.



Hassle-Free Remote Activation

Demos and features can be renewed, converted to full-featured applications in the field.

You can create *try-and-buy* offers supported using commonly used methods, like an e-mail, a fax, or a phone call.

Security to Count On

Developers from diverse industries have consistently relied upon our hardware-based anti-piracy solutions. Sentinel UltraPro scores much higher than any other software or hardware-based products available in the market mainly due to the robust feature set combined with a versatile key.

Most Flexible Read/Write Memory

Sentinel UltraPro combines multiple encryption algorithms with read/write memory in one key. The encryption algorithms bring the *challenge-response* functionality into play and provide superior query/response protection to your applications (see the diagram on page 11). The algorithms can be combined with activation passwords, lease dates, demo options, time tampering checks, and so on.

A key can also be programmed with multiple data types to provide several types of fixed and variable responses—for unique and advanced protection strategies. You can store fixed user data, like serial numbers, user names, strings, or codes to control the feature access, program flow or application functions. Such data can be read by your application to verify the key is still attached or not. The key can also be programmed with counters to restrict the number of executions.

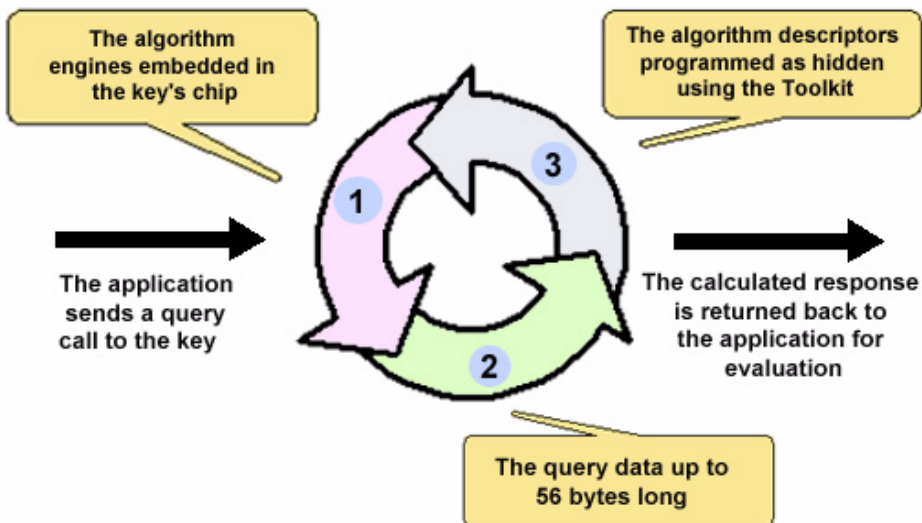
Proven Query/Response Protection

The UltraPro key provides a way of scrambling up to 56 bytes of data according to an internally stored, standard algorithm engines and the algorithm descriptors you define (both the Full License and Short License API elements provide algorithm descriptors).

Unlike writing and reading values to/from the key, which simply provides a data storage facility, the query/response mechanism transforms data sent to the key in a way that is unique for your particular UltraPro key.

Your application sends a query that consists of the bytes of data to be scrambled and the key returns a response that contains an identical number of scrambled data bytes. The resulting data (response) is unique for your par-

ticular UltraPro key and the algorithm descriptor used. The *to-and-fro* of queries and responses is enabled via the query functions you add in your source code. You can query the key as many times as you want with query data as long as 56 bytes. Of course, random query values in your code—and lots of them—make your application more secure.



How a Query is Processed?

The query/response protection is very reliable due to the logic embedded in the key. This makes your strategy a moving target dependent on the following building blocks:

- The query data is sent by the application is encrypted.
- The algorithm seeds you define are programmed as “hidden” into the hardware and cannot be read/written by your application or a debugging software. Their modification requires the overwrite passwords—assigned to each developer by your key vendor. The 32-

bit long passwords are beyond any hit-and-trial activity, aiming to break your application's security.

- The encryption engines are burnt into the key's chip—hence are difficult to be reversed or duplicated.

Uses of the Query/Response Method

At the most basic level, the query/response mechanism can be used to verify the presence of the key by sending “known” query values to the key and checking for “known” response values from the key. Using this mechanism, the protected application would contain a table of query values and a table of matching response values. If the correct response was received for a query, then it is assumed that the correct UltraPro key is present. It is a good practice to encrypt both the query and response tables to prevent a hacker from obtaining the data.

A better use of the query/response mechanism is to encrypt and decrypt critical data in a protected application. The data can be encrypted using a symmetric public encryption algorithm contained within the protected application. The application would send a query to the UltraPro key and use the response as a seed to the symmetric encryption algorithm that would be used to decrypt the critical data. If the correct key is present then the application would get the critical data it needs and will run normally.

Built-In Passwords in the Key

The UltraPro hardware keys are customized for each developer, which means another developer cannot reprogram your keys. The ability to program UltraPro keys is protected by a set of passwords: the *write* password and two *overwrite* passwords. These passwords are hidden and burnt into the chip for maximum security. An UltraPro protected application NEVER exchanges these passwords for locating the key on run-time.

One-Time Update Feature Always On

For activating applications/features in the field, additional security features are implemented. As a result, the one-time update feature is always preva-

lent to ensure that a license code is only applied once. Any subsequent attempts to apply the same license code will fail.

Anti-Time-Tampering Measures

To guard the security of your leased and time-bound applications, anti-time tampering measures are provided. You can choose to deny access to the application for any major violations or tolerate some minor violations before finally disabling the application.

Robust and Convenient Licensing Solutions

Sentinel UltraPro provides you two methods to protect your applications: Shell and API elements:

***Get-Set-and-Go* Protection With Shell**

Shell is the most popular way to protect applications—quickly and easily. In Shell, a protective layer is wrapped around your application's executable. Hence, it is automatic and does not require the source code of your application.

Customized Protection With API Elements

API elements offer you the most robust protection with immense scope for customization. You can implement a strategy of your choice right into your application's source code using the Sentinel UltraPro API. You can create secure and unique protection schemes commensurate with the techniques you adopt. Here are its few highlights:

- You can program query/response algorithms, activation passwords, lease dates, execution counters, integers, strings, Boolean into the key.
- API elements are driven by the high-level UltraPro API meant for quick prototyping and implementation.

- The Toolkit generates a code sketch and a header file that greatly simplifies the integration of the API with your source code.
- Sample designs are included to guide you on using the various elements.

Tip: Use Shell and API elements together

For maximum protection, you can apply Shell layer over the application (executables and DLLs) already protected using API elements.

Refer to Chapter 5, “Planning Application Protection,” on page 45 for details.

Streamlined Project Management

The UltraPro Toolkit offers an intuitive graphical interface with short learning curve.

It follows a 3-tier framework of *Project*, *Design*, and *Elements* for convenient strategy management. This framework is used throughout the Toolkit—from designing stage to programming keys. Furthermore, you can create, remove, duplicate, and share your strategies.

Refer to Chapter 3, “The Sentinel UltraPro Toolkit,” on page 25 for details.

Licensing and Your Channels of Distribution

Key Manager Application for Your Sales Distributors

Sentinel UltraPro comes with two versions of the Key Manager. One is integrated with the UltraPro Toolkit and meant to be used on the developer's end for programming distributor and product keys and generating license codes.

A separate installer has been provided to set up the stand-alone Key Manager application for your distributors and key manufacturing personnel. It offers the restricted functionality of programming product keys and generating license codes only. Your distributors will require a *.dst* file and a distributor key to operate. The key can be programmed with a counter value to keep track of how many keys are programmed and how many license codes are generated by the distributor. This implies that you can allow your distributors to perform sales tasks without risking your protection strategy or revenues.

Licensing at the Need of the Hour

The task of preparing a protection scheme is almost effortless and highly business-oriented—making it easy for you to tackle the *time-to-market* gap. Many of the licensing settings—such as, the number of days, number of executions, and user-limit—can be modified at the last stage when you or your distributor are programming the keys. This makes strategy designing a *one-time activity* and allows you to offer customized solutions on demand.

Flexibility that Your Customers Prefer

Sentinel UltraPro offer truly convenient licensing solutions for your customers.

Convenient Deployment of UltraPro Redistributables

Using the Sentinel Protection Installer, the deployment of a UltraPro protected application is rather simple. It offers you two choices on how to deploy the UltraPro redistributables:

- You can directly ship the Sentinel Protection Installer to your customers. They can run the installer to install Sentinel Driver and/or Sentinel Protection Server without any assistance. A copy of this installer is also available at our Technical Support Web site (<http://www.safenet-inc.com/support/index.asp>).
- If desired, you can integrate the Sentinel Driver and Sentinel Protection Server components along with your application's installer. Refer to the Sentinel Protection Installer's Help for details.

Licensing Means Freedom

Unlike the other contemporary licensing methods that spell a disaster on the customer's site due to lack of mobility or tedious processes of transferring licenses, Sentinel UltraPro offers portable and hassle-free licensing to your customers. The hardware keys are easy-to-carry, small, durable and widely-compatible. Furthermore, with Sentinel UltraPro they can make use of advanced licensing schemes for customized budgets and needs.

Chapter 2

The Sentinel UltraPro Components

In this chapter we will learn about the following components of the Sentinel UltraPro protection system:

- “Sentinel UltraPro Key” on page 18
- “Sentinel Driver” on page 18
- “Sentinel Protection Server” on page 18
- “Sentinel UltraPro Toolkit” on page 21
- “Sentinel UltraPro API” on page 23
- “The Protection Methods - API Elements and Shell” on page 24
- “Sentinel Protection Installer” on page 24
- “The Field Activation Methods” on page 24

For the list of platforms supported by each component, refer to the Release Notes.

Sentinel UltraPro Key

The UltraPro key is the heart of your protection. It eliminates unlicensed use by locking an application to a physical key. Henceforth, your customers will always require the key to run your protected application.

It is powered by on-board encryption algorithms that enable query/response-based protection for your application. The key contains customizable memory that can store variety of elements—such as, strings, booleans, integers and counters—for strengthening your protection strategy.

The UltraPro keys are also available for the stand-alone and network environments. It is available in both the parallel and USB form-factors, and different memory-sizes. Refer to Chapter 4, “The Sentinel UltraPro Key,” on page 33 for details.

Sentinel Driver

Sentinel driver is the device driver for communicating with the keys, It must be distributed with all your protected applications.

Note: USB driver is WHQL certified for Windows 2000/XP/Server 2003

The Sentinel USB driver is Microsoft Windows Hardware Quality Labs (WHQL) certified for Windows 2000/XP/Server 2003. This allows Windows Logo compliance for applications that use the Sentinel USB keys.

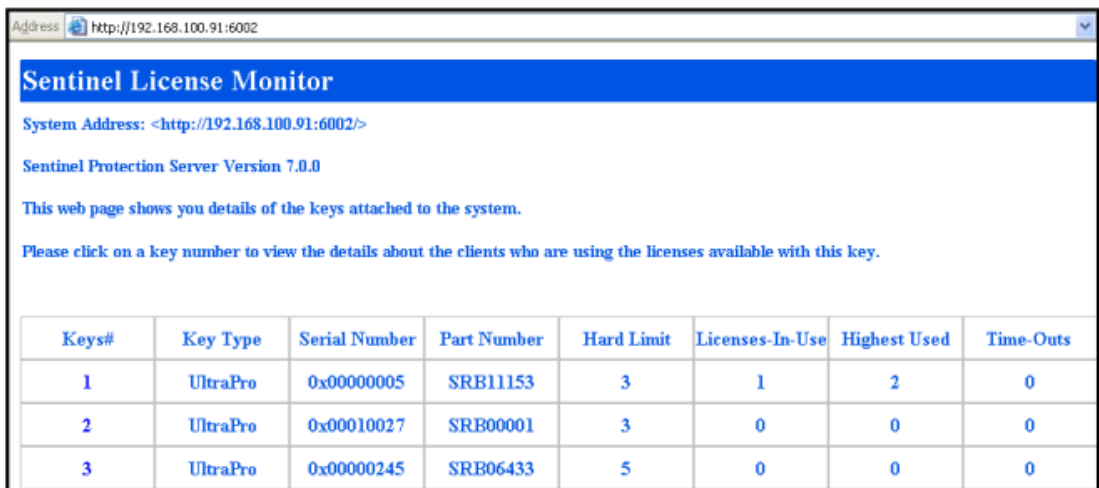
Sentinel Protection Server

Sentinel Protection Server is the license manager of your networked applications. It maintains the license information of the keys attached to the system. It is typically installed on the networked systems where the key is attached—allowing concurrent access to an application using a single key.

- The Sentinel Protection Server enforces a limit on the number of applications that can be run. It issues licenses based on the conditions you have stipulated, like the hard limit, user limit and sharing criteria.
- It tracks the number of licenses in-use for each key. It will not grant new licenses once the key's limit has been exhausted.
- Sentinel Protection Server supports the NetBEUI, TCP/IP, IPX, and SAP protocols for licensing. The HTTP protocol is used for fulfilling the Sentinel License Monitor requests from clients on LAN/WAN.

About Sentinel License Monitor

Sentinel License Monitor shows detailed information of the Sentinel keys attached to a system and the clients accessing them via a Web browser. For example, your customer could use the License Monitor to determine whether or not enough licenses were purchased, based on license demand.



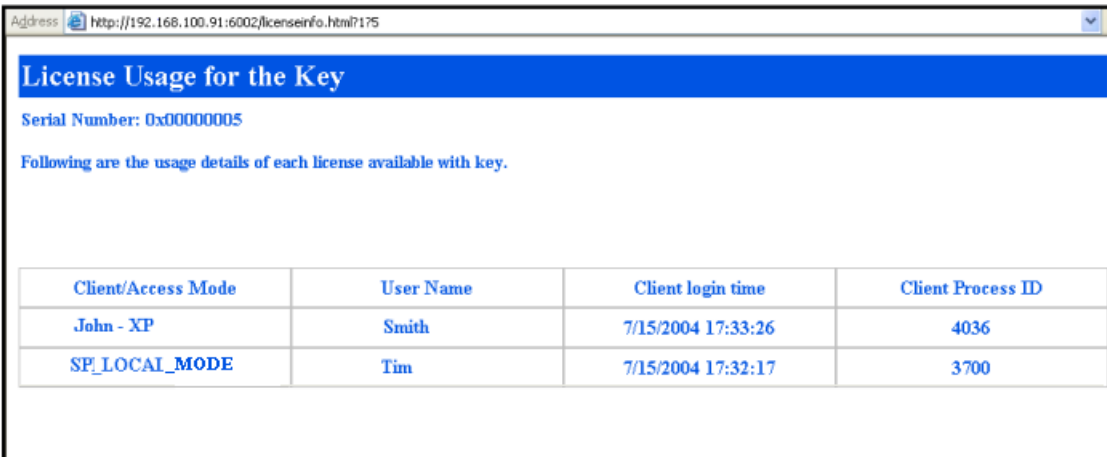
The screenshot shows a web browser window with the address `http://192.168.100.91:6002`. The page title is "Sentinel License Monitor". Below the title, it displays the system address, version (7.0.0), and instructions. A table lists three keys with their details.

Keys#	Key Type	Serial Number	Part Number	Hard Limit	Licenses-In-Use	Highest Used	Time-Outs
1	UltraPro	0x00000005	SRB11153	3	1	2	0
2	UltraPro	0x00010027	SRB00001	3	0	0	0
3	UltraPro	0x00000245	SRB06433	5	0	0	0

Sentinel License Monitor Displays the Keys Details

The following details will be shown about the keys connected to a system:

- Serial Number, Part Number and hard limit of the keys attached.
- Number of licenses currently in use.
- Highest number of licenses issued by a key.
- Number of time-outs—the sessions that collapsed without properly releasing the license—recorded by the Sentinel Protection Server for clients using licenses from a key.
- Information about the clients (such as, the user name, login time, and client process ID) who have currently obtained licenses from the key.



The screenshot shows a web browser window with the address bar displaying `http://192.168.100.91:6002/licenseinfo.html?175`. The page title is "License Usage for the Key". Below the title, it shows the "Serial Number: 0x00000005". A message states: "Following are the usage details of each license available with key." Below this message is a table with four columns: "Client/Access Mode", "User Name", "Client login time", and "Client Process ID". The table contains two rows of data.

Client/Access Mode	User Name	Client login time	Client Process ID
John - XP	Smith	7/15/2004 17:33:26	4036
SP_LOCAL_MODE	Tim	7/15/2004 17:32:17	3700

Sentinel License Monitor Displays the License Details

Tip: Customize Sentinel License Monitor's look-and-feel

You can customize the *look-and-feel* of the Sentinel License Monitor by modifying the Sentinel Protection Server class files. Refer to the readme available in the *Sentinel License Monitor* directory in your installation.

Sentinel UltraPro Toolkit

Toolkit is a Java application using which you can perform the various tasks of protecting your applications without getting exposed to the underlying details of the UltraPro system and the key. You will be using the Toolkit for performing the tasks described below.

Designing the Application Protection Strategy

You can create a protection strategy using Shell and API elements in Toolkit. Refer to the topic “The Protection Methods - API Elements and Shell” on page 24 for more information.

As a part of your strategy, you can also define the remote update actions for activating applications/features remotely. The first four chapters of Part 2 discuss the application protection strategy in detail.

Prototyping and Implementing the Strategy

You will be using the Toolkit in the ways described below for prototyping and implementing the application protection strategy:

- After creating a strategy, you can prototype it for a key using the **Build** button of the Toolkit.
- Since the Toolkit implements the Shell protection automatically, no additional programming effort is required.
- Toolkit also generates a header file and a protection plan (in the form of *code sketch*) that assists you implementing protection in the application's source code. The API Explorer allows you to evaluate the various API functions before you call them into your application.

Refer to Chapter 10, “Implementing the Application Protection,” on page 129 for more details.

Programming Keys and Generating License Codes

When you are ready with your protection strategy, you can program the UltraPro keys that will be shipped along with your application. Refer to Chapter 11, “Programming Product Keys,” on page 139 for details.

If desired, you can allow your distributors to program keys and generate license codes for your customers. To do so, you need to provide a *.dst* file and a distributor key to your distributors. Refer to Chapter 12, “Programming Distributor Keys,” on page 149 for more details.

To update keys in the field, you can generate license codes in Toolkit. Refer to Chapter 13, “Generating Licenses And Updating Keys,” on page 157 for details.

Viewing the Key’s Memory and Layout

Toolkit includes many user-friendly options, like Key Layout, Memory View, Element View and Memory Meter to help you in understanding your strategies graphically.

Managing Your Strategies

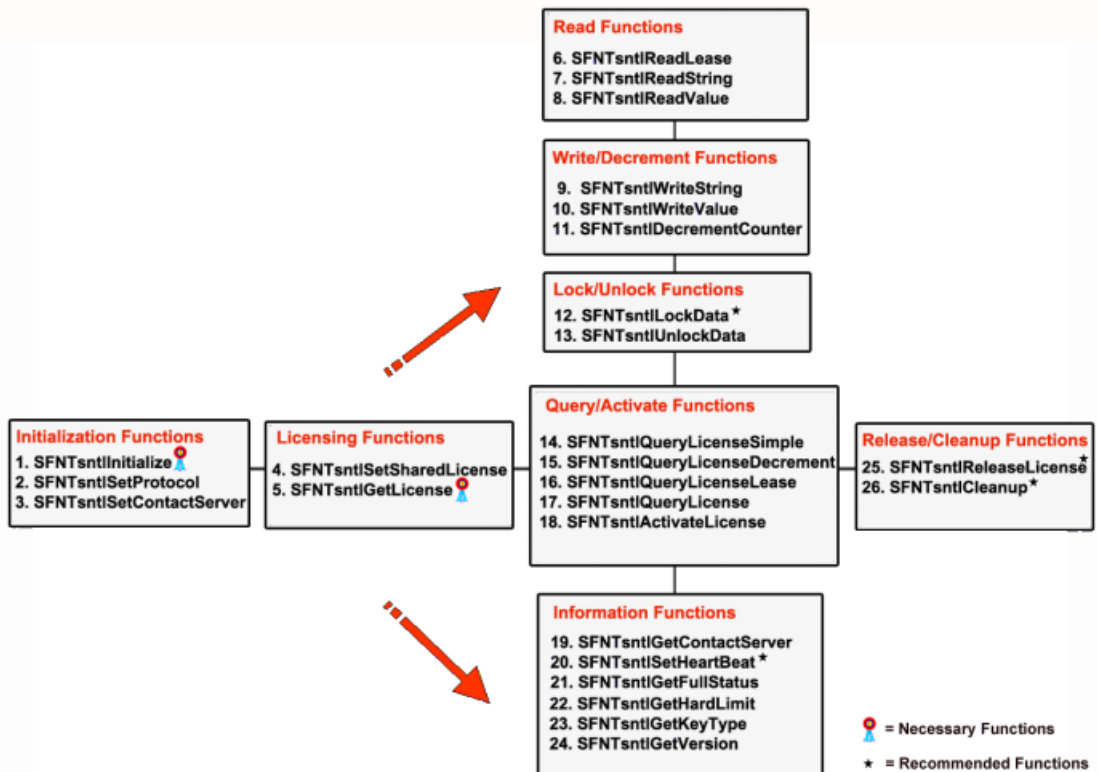
The entire Toolkit is based on the *project-design-element* hierarchy. As a result you can create, modify, import, export and reuse the strategies with *point-and-click* ease!

Refer to Chapter 3, “The Sentinel UltraPro Toolkit,” on page 25 to learn more about the UltraPro Toolkit.

Sentinel UltraPro API

The Sentinel UltraPro API is a set of functions calls used to communicate between your application and the UltraPro key. These API functions can be used for calling the key, verifying its presence, obtaining a license, reading/writing certain data on the key, sending queries and evaluating responses and so on.

The Sentinel UltraPro API provide a high-level interaction with the key and drastically reduce the efforts involved in implementing several popular license models.



The Sentinel UltraPro API Collection

The Protection Methods - API Elements and Shell

Sentinel UltraPro provides two types of protection methods: API elements-based and Shell. The protection method determines how and where the software locks are implemented.

.When you choose API elements, you create a design strategy using the API elements and add software locks (API functions to verify the presence of the key) directly into your application's source code. You control the amount and location of the locks.

In Shell, a protective layer is wrapped around your application. The Shell layer makes no changes to your application's source code. All software locks and communication with the hardware key (such as checking and verification) is handled by Shell.

Refer to the topic “Shell Versus API Elements” on page 46 for details.

Sentinel Protection Installer

The Sentinel Protection Installer is a dual installer of the Sentinel driver and Sentinel Protection Server. These components are required by the customers using your protected applications.

Refer to the Sentinel Protection Installer Help for details on how to deploy the redistributables.

The Field Activation Methods

Sentinel UltraPro provides you two field activation methods—the Field Activation Utility and the Sentinel Client Activator—that give you with the capability of updating applications in the field.

Refer to Chapter 8, “Designing the Field Activation Strategy,” on page 103 for details.

.

Chapter 3

The Sentinel UltraPro Toolkit

In this chapter we will briefly discuss about the Sentinel UltraPro Toolkit, including the *Projects-Designs-Elements* hierarchy. Refer to the Toolkit Help to learn everything about it.

Note: Have you installed the Toolkit?

This guide assumes you have already installed the UltraPro Toolkit. If not, please refer to the release notes for details on installation.

The Toolkit Tools

The entire Toolkit has been organized under the following tools/screens:

- Quick Shell
- Protection Manager
- Key Manager
- API Explorer

Quick Shell

Using Quick Shell, you can protect Win32 executables and DLLs in fast and easy manner. It employs the Shell method for protection and lets you to protect a single file at a time.

It also offers popular license management controls, like an expiration date, trial days, and an execution count. It is typically used for getting started with the UltraPro Toolkit and the Shell method. See “Using Quick Shell” on page 76 for details.

Protection Manager

The **Protection Manager** screen is your main workspace for designing and prototyping the protection strategies. Here you can:

- Create projects, designs, and elements.
- Prototype a design using the **Build** button.
- Add the field activation actions for updating keys remotely.
- View a virtual layout of your design.

See “About Projects, Designs, and Elements” on page 29 for details.

Key Manager

In the **Key Manager** screen, you can program keys for your customers and distributors and generate license codes for updating keys remotely.

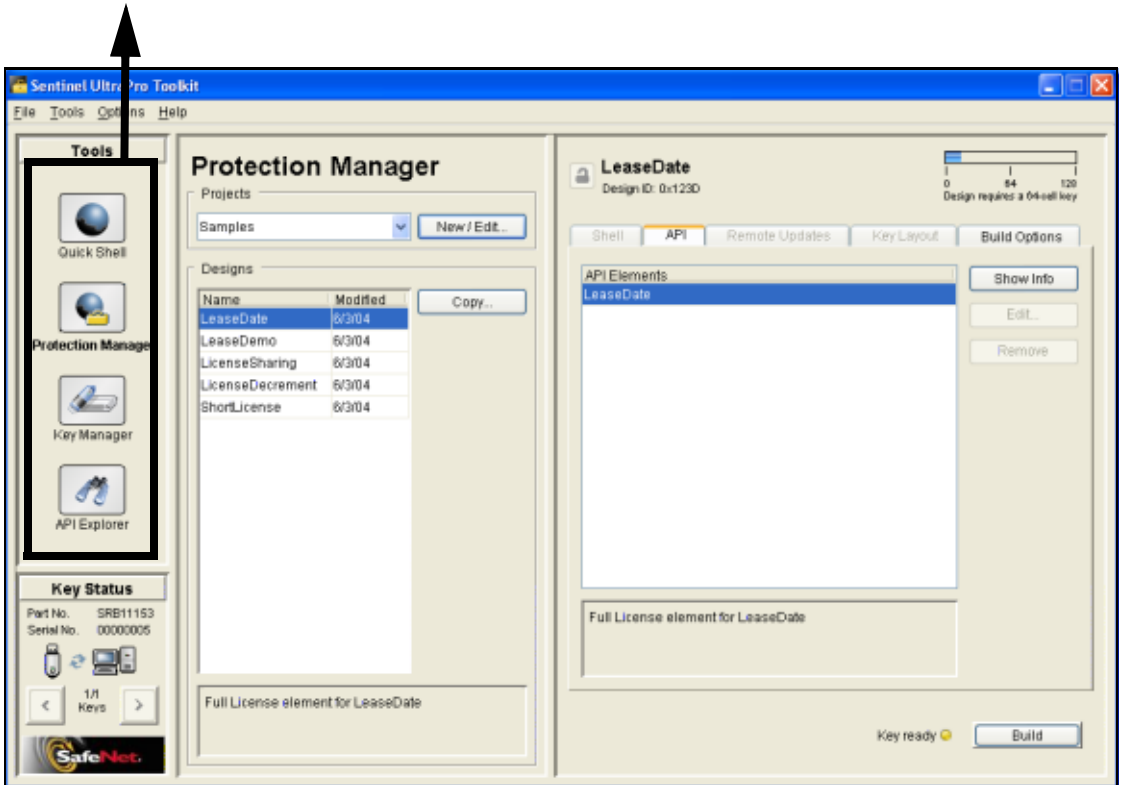
See the Part 3 - Programming Keys and Generating Licenses of this guide for details.

API Explorer

In the **API Explorer** screen you can:

- Evaluate the UltraPro API functions prior to adding them to your source code.
- View the graphical layout of a design in the key memory (Element View).
- View the low-level memory layout of a design in the key memory (Memory View).

Navigate the Toolkit screens using the Tools icons



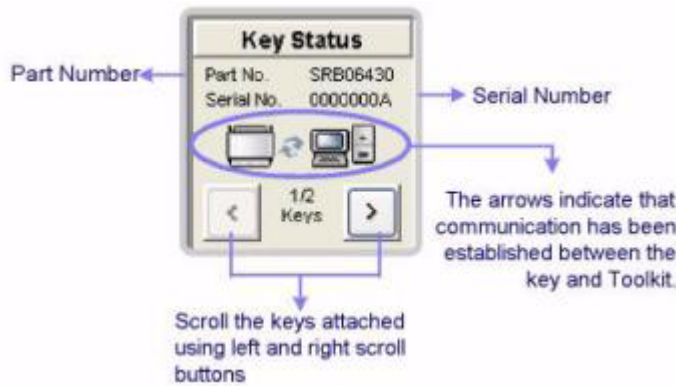
The Protection Manager Screen of the Toolkit

Tip: Getting started with the Toolkit

The fastest way to get started with Toolkit is to take a tour of it using the video tutorials provided under the **Help** menu. Afterward, you can start by configuring an UltraPro key; followed by creating a design using the **Protection Wizard**. For complete details, refer to the Help.

The Key Status Pane



The **Key Status** pane displays the details of the keys attached to the system where the Toolkit is running. It can list up to five keys at a time.






The Key Status Pane

Note: Ignored keys will not be shown
The **Key Status** pane will not show the ignored keys (whose details were not provided in the **Key Configuration** dialog box when the key was attached for the first-time). However, you can re-configure an ignored key using the **Settings** option under the **Tools** menu.

The clips described below will help you in identifying the key details:

Clip	Description
	Unable to access the key! Either no key is attached or the key has been removed.
	Accessing a Sentinel UltraPro USB port key.

Clip	Description
	Accessing a Sentinel UltraPro parallel port key.
	The key is not recognized by the UltraPro Toolkit.
	<p>Accessing a key in the API Explorer screen.</p> <p>The Key Status pane will not be refreshed in the API Explorer screen unless a license is acquired by calling the SFNTsntlGetlicense function.</p> <p>Note that the host computer will change as per the access mode set using the SFNTsntlSetContactServer function.</p>

About Projects, Designs, and Elements

The Sentinel UltraPro Toolkit is based on the *Projects-Designs-Elements* hierarchy, which streamlines your project management efforts. As a result, you can easily create, duplicate, and move your protection strategies for multiple products and across different systems.

What Is a Design?

A design refers to the strategy you prepare to protect your applications against software piracy. This strategy is made up of the elements described below.

What Are Elements?

An element is the basic unit of a protection strategy (design).

A design can contain one or more elements; however a key can be programmed with only one design. Corresponding to every element, a Toolkit Cell Address is generated that maps the element in the key programmed with the design. Using elements, you can achieve the following goals:

- Protect your application by manipulating the data programmed in the key.
- Pack and license your application in various forms, such as demo, full-featured, limited featured, and so on.

Types of Elements

A design can contain the following types of elements:

Shell

The Shell elements are used for automatically protecting the Win32 executables and DLLs.

API

The API elements allow you to create customized protection strategies for your applications. You program a key with a design and add the corresponding UltraPro API functions into your application's source code.

Given below is a list of the API elements—see Chapter 5, “Planning Application Protection,” on page 45 for more details:

- Full License
- Short License
- String
- Boolean
- Counter
- Integer

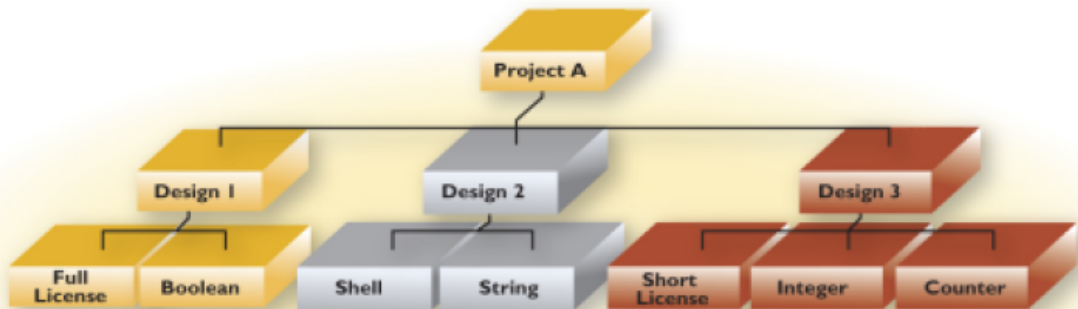
What is a Project?

A project is a container of designs. It comprises of one or more designs made up of elements. It is especially useful to have a project when you want to maintain designs for various products, versions, and releases.

A project can be easily duplicated and shared to facilitate management of your protection strategies.

For example, you may use Sentinel UltraPro to protect various products.

You create a project for that product—for example Product A. Various versions/releases of Product A can be your designs—for example Design Alpha, Design Beta, and so on. Ultimately, each design can consist of different elements.



The Project-Design-Element Hierarchy

The “One Key:One Design” Relationship

A key can be programmed with only one design. The programmed key carries a design ID that is used by the `SFNTsntlGetLicense` function for obtaining a license.

A Design Has Various Uses

While an element is the basic unit of a protection strategy, it is a design that is used for multiple purposes. For example:

- A product key is programmed with a design.
- Your distributor is given a *.dst* file and a distributor key specific for a design.
- To update your keys in the field, you add field activation actions for a design.
- A design can be used for protecting different applications.
- A design can be imported/exported for backup and sharing purposes.

Chapter 4

The Sentinel UltraPro Key



This chapter explains the UltraPro key terminology and features and procedures for ordering and returning the keys.

The UltraPro Key Terminology

Term	Description
Cell	<p>A memory location on the key that holds 16-bit values. The UltraPro keys come in various memory sizes, organized as cells.</p> <p>Think of a cell as being a container (memory location) for protection data you want to program in the key. Cells have addresses that represent their location on the key, much like street addresses represent the location of houses in a neighborhood.</p> <p>Since the Toolkit writes the values on the key there is no need for you to locate a free cell to write. Also, the Toolkit arranges the elements in a best-fit manner on the key, so the need to manually rearrange elements seldom arises.</p>

Term	Description
Hard Limit	<p>The factory-programmed limit that defines the maximum number of licenses that can be obtained from the key is called the hard limit.</p> <p>Sentinel UltraPro keys come with 1, 2, 3, 5, 10, 25, 50 or unlimited hard limits.</p> <p>In case of a stand-alone application, typically key with one hard limit is connected to the user's workstation allowing access to the application running locally.</p> <p>In case of a network application, clients in the subnet access one key (having suitable hard limit) attached to a system within the network where the Sentinel Protection Server runs.</p>
User Limit	<p>Though you can't alter the hard limit burnt into the integrated circuit of the key, you can program an intermediate limit into the key, which is known as the user limit.</p> <p>Refer to the topic "Specifying a User Limit" on page 64 for more details.</p>
Product Key	<p>Product keys are shipped to your customers with your protected application, providing access to the application.</p> <p>The product keys are programmed using the Make Keys option of the Toolkit/Key Manager application. Refer to Chapter 11, "Programming Product Keys," on page 139 for details.</p>
Distributor Key	<p>Distributor keys are given to your sales distributors, allowing them to perform activation and update functions for customers when they sell your application. Distributor keys must be connected to the system where the Key Manager application will be installed and run.</p> <p>See Chapter 12, "Programming Distributor Keys," on page 149 for details.</p>
Reserved Cells	<p>Refers to the cells in the UltraPro key that contain fixed, factory programmed key information (such as the serial number, developer ID and passwords) or are written by the UltraPro Toolkit for implementing your protection strategy. These cells are fixed and cannot be moved to any other location in the key.</p>


00	01	02	03	04	05	06	07
08	09	0A	0B	0C	0D	0E	0F
10	11	12	13	14	15	16	17
18	19	1A	1B	1C	1D	1E	1F
20	21	22	23	24	25	26	27
28	29	2A	2B	2C	2D	2E	2F
30	31	32	33	34	35	36	37
38	39	3A	3B	3C	3D	3E	3F
40	41	42	43	44	45	46	47
48	49	4A	4B	4C	4D	4E	4F
50	51	52	53	54	55	56	57
58	59	5A	5B	5C	5D	5E	5F
60	61	62	63	64	65	66	67
68	69	6A	6B	6C	6D	6E	6F
70	71	72	73	74	75	76	77
78	79	7A	7B	7C	7D	7E	7F
80	81	82	83	84	85	86	87
88	89	8A	8B	8C	8D	8E	8F
90	91	92	93	94	95	96	97
98	99	9A	9B	9C	9D	9E	9F
A0	A1	A2	A3	A4	A5	A6	A7
A8	A9	AA	AB	AC	AD	AE	AF
B0	B1	B2	B3	B4	B5	B6	B7
B8	B9	BA	BB	BC	BD	BE	BF
C0	C1	C2	C3	C4	C5	C6	C7
C8	C9	CA	CB	CC	CD	CE	CF
D0	D1	D2	D3	D4	D5	D6	D7
D8	D9	DA	DB	DC	DD	DE	DF
E0	E1	E2	E3	E4	E5	E6	E7
E8	E9	EA	EB	EC	ED	EE	EF
F0	F1	F2	F3	F4	F5	F6	F7
F8	F9	FA	FB	FC	FD	FE	FF

Reserved Memory  Read/Write Memory 

The Memory Layout of a 256 Cell Key

The UltraPro Key Features

The Sentinel UltraPro key is available in different memory sizes, including the 128 bytes (64 cells) key. Your choice depends upon the number of applications you are protecting and elements you are including in your protection strategy. Contact your Sentinel key vendor to choose the key that suits your need best.




The Hardware Key Developers Prefer!

Proprietary algorithm engines for excellent query-response protection

Available for stand-alone and network users


Long-lasting data retention capability and wide compatibility

Can contain variety of data such as, counters, integers, strings and boolean flags...



**Sentinel™
ULTRAPro™**

Choose the memory size of the key depending on number of applications you are protecting and your strategy.



The Sentinel UltraPro Key Highlights

Each key has some reserved cells, which contain sensitive information. For maximum security, certain data is burnt into the circuit (factory-programmed) of the key—making it inaccessible for any kind of manipulation.

The 25-pin parallel port keys connect to a parallel port located on the back of a computer. The USB keys connect to a USB port located on the back or front of a computer or on a USB hub.

The parallel port keys are completely transparent, allowing normal computer-printer communication. During operation, the keys use current from all possible signals-pins to power their internal logic. Hence very little power is required to operate the unit.

The state-of-the-art USB keys are convenient to use. The USB keys not only provide simple and quick installation for a user, but are designed for maximum durability and *ease-of-use*—enabling them to be inserted and removed repeatedly. The universal serial bus is a low cost, plug and play, standardized interface included on virtually all personal computers built since early 1997. All future PCs are certain to have USB ports, while the widespread availability of other interfaces remains uncertain.

Note: Sentinel USB keys are not supported on Windows NT 4.0

Please note that Sentinel USB keys are not supported on Windows NT 4.0. You may consider shipping parallel port keys to Windows NT 4.0 customers.

Attaching a Key

The key must be connected to your workstation for performing the several application protection tasks in Toolkit.

The keys come in two forms: parallel port or USB. The type you received in your package depends on what you specified when you placed your order.

You need not worry about installing Sentinel driver on your system where the UltraPro Toolkit will be run. This is because the Toolkit setup program automatically installs that for you.

On your customer's computer, the keys will be attached in the same manner described below. However, you need to ensure that the latest Sentinel driver is installed on the system before your customer attaches the key.

Attaching a USB Port Key

1. Locate an available USB port on your computer.

2. Attach the key to the USB port. Make sure it is securely and tightly connected. Look to see that the LED on the key is illuminated to verify whether the key has been plugged-in properly.

Attaching a Parallel Port Key

1. Locate an available parallel port on your computer. The key can be attached to any parallel port on your computer, as the Sentinel driver automatically polls each port to locate the key.
2. Attach the key to the parallel port connector and tighten the screws to connect the key securely to the port.

As soon as the key is detected by the Toolkit, the **Key Configuration** dialog box will appear. Provide the required information from the password card to configure the key.

Special Cases for Attaching Keys

Given below are tips for attaching Sentinel UltraPro keys under specific circumstances:

Connecting Multiple Parallel Port Keys

Multiple UltraPro keys can be attached to the same parallel port; this is called *cascading* or *daisy-chaining*. You must be aware of the following information before you attach multiple keys:

- Sentinel parallel port driver supports up to 5 keys attached to a port. Keys can also be connected to up to three parallel ports on the same computer. The Sentinel driver automatically polls all ports when looking for a key.
- Multiple USB port keys can be attached using a USB hub. Sentinel USB port driver supports up to 32 keys attached to a hub.

- The Sentinel Protection Server works in conjunction with the Sentinel USB and parallel drivers. It can support a maximum of 10 keys, but is limited by the type of keys used.
- The UltraPro keys can also be cascaded with other Sentinel keys that support cascading. Make sure the UltraPro keys are the last keys in the chain (farthest from the computer).

We expend great effort to ensure our products are compatible, transparent and convenient to use. While UltraPro keys can be cascaded with keys from other companies, it may cause compatibility issues and is not recommended.

**Warning about using the NetSentinel keys**

There is one exception to this rule. If you are cascading Sentinel UltraPro keys with NetSentinel keys, the UltraPro keys must be located before any NetSentinel keys in the chain. NetSentinel keys should be the last keys in the chain. If the UltraPro keys are located behind NetSentinel keys, they will not be recognized by the Sentinel driver, and thus the protected application will not run.

When You Have Only One Parallel Port

If your computer has only one parallel port, you may need to temporarily remove any existing parallel port devices (such as a Zip drive or printer) in order to connect the key. These devices may be reconnected to the key's outside connector after you have installed the key. The parallel port key is capable of working with most Zip drives, tape backups and other bi-directional devices.

Support for User-Configured, Parallel Port PCI Cards

A feature has been added to the Sentinel System Driver for Windows NT based operating systems that will allow it to recognize add-in PCI parallel port adapters that have been configured by the operating system. You need not enter the port addresses assigned to those adapters.

Recommendations for Attaching a 25-Pin Key

If you are using a 25-pin key, we recommend you attach the key directly to the parallel port without using an extension cable between the computer and the key. However, you may use a cable to connect a printer or other parallel device to the key.

Please be aware that not all combinations of cables and printers are compatible with the UltraPro key—contact our Technical Support if you encounter a compatibility problem.

If necessary, reconnect any other parallel port devices to the outside connector on the key. We recommend using a shielded printer cable if you are connecting a printer to your computer through the key.

Using Cables

Due to the large variety of cables currently on the market, we do not recommend a specific brand or type of cable for use with the UltraPro key. However, we do recommend the following:

- Cables should not be longer than 6 feet in length.
- Cables should be shielded.
- Do not use ribbon cables.
- Cables must be straight-through; that is, they must have all pin signals wired through to the connectors on either end of the cable.

Note: See the troubleshooting section of the Toolkit Help

You can find some more useful information about the UltraPro key in the Toolkit Help's Troubleshooting and FAQs section.

Ordering and Returning Keys

The UltraPro Toolkit comes with one hardware key. You need to order additional keys that you will program and then ship with your protected applications.

Ordering Additional Keys

Contact your Sentinel sales representative to order additional keys. Be ready with the following information before ordering keys:

- What key memory size do you want?
- What key format you want: parallel or USB?
- What is the hard-limit of the key you want?

Returning Keys

Occasionally, you may find that you need to return a key for exchange or repair. If you suspect a technical problem, call our Technical Support using the information provided page xiv. The support representative will work with you to rule out resolvable software and/or configuration problems. If the problem cannot be resolved, the RMA (Return Material Authorization) department will assign you an RMA number over the phone. To ensure proper handling is acknowledged for the returned keys, you must obtain a RMA number prior to shipping the products.

Packaging the Keys for a Return

After you have obtained an RMA number and are ready to package the keys for shipping, please read and follow these packaging guidelines:



Beware of electrostatic charges

Electrostatic charges can damage the hardware keys. We strongly recommend following these guidelines at all times to prevent damage to your keys.

- Install an electrostatic-dissipating mat as a work surface, and make sure the mat is properly grounded.
- Wear grounding wrist or ankle straps while handling the keys.
- Use packaging materials designed to avoid electrostatic charge during shipment. Plastic that does not generate static (“cold plastic”) is typically pink in color. You may also use “conductive plastic”, which is designed to drain off static.

Part 2

Designing and Implementing Protection

- ❑ Planning Application Protection
- ❑ Using Shell Protection
- ❑ Using API Elements-Based Protection
- ❑ Designing The Field Activation Strategy
- ❑ Implementing Protection - Tips and Techniques

Chapter 5

Planning Application Protection

This chapter helps you in planning your application protection. Even though the exact steps of protecting applications are given in subsequent chapters, here you will do the indispensable groundwork.

Factors You Must Consider Before Protecting

- Do you prefer Shell or API elements to protect your applications?
- If using API elements, which one will you choose?
- Which access mode you will choose and how do you wish to set it? This is a useful decision for deploying your application in both the stand-alone and network environments.
- Will you use the field activation process for updating keys remotely?
- Will you specify a user limit in your strategy? Furthermore, how would you like to share the licenses issued against a user limit?
- How to watchguard your leased applications against time-tampering?
- Would you like to authorize your sales distributors for programming the product keys, modifying the design settings, and activating applications in the field?

Shell Versus API Elements

Sentinel UltraPro offers you two methods for protecting your applications: Shell and API elements. Here is a quick snapshot of both the methods.

Comparing Shell and API Elements

Shell	API Elements
Methodology	
<p>In this method, a protective layer is wrapped around your application's executable file. This layer encrypts the executable and handles all the communication with the key—making it almost impossible to run the application without the hardware key.</p> <p>The Shell makes no changes to your application's source code—so there is no need to recompile your application.</p>	<p>Requires you to add the UltraPro API functions into your source code. This way you can customize the application protection by controlling the amount and location of the API calls in your code.</p> <p>You begin by creating a design made up of API elements—Full License, Short License, Integer, and so on. You then add the UltraPro API to your source code and compile and link it with the UltraPro client library. Finally, the hardware key is programmed with the design.</p> <p>You may also apply the Shell wrapper around your protected application for extra security.</p> <p>The API elements can be categorized into two major categories:</p> <ul style="list-style-type: none">❑ Algorithm-Based Elements The Full License and Short License elements are algorithm-based elements and let you utilize the query/response-based protection along with popular licensing options. See the topic “Proven Query/Response Protection” on page 10 for details.❑ Data Elements String, Boolean, Integer (8-bit, 16-bit and 32-bit), and Counter are the data elements that can be used for supplementing and strengthening your main protection strategy (created using a Full License or Short License element). The possibilities of using the data based elements are almost endless—you can use them for storing the application data and so on.

Comparing Shell and API Elements (Continued)

Shell	API Elements
Licensing Options	
<p>Using Shell you can effortlessly include the following licensing options:</p> <ul style="list-style-type: none"><input type="checkbox"/> Execution Count<input type="checkbox"/> Expiration Date<input type="checkbox"/> Trial Days<input type="checkbox"/> Cheat Counter<input type="checkbox"/> Access Mode<input type="checkbox"/> User Limit<input type="checkbox"/> Sharing Criteria<input type="checkbox"/> Encrypt data files	<p>Most of the licensing options offered by Shell are also available with Full License. In addition, a set of powerful UltraPro API functions is also provided to build robust and tailor-made protection.</p>
Time and Efforts Consumption	
<p>The Shell certainly yields quick results. The entire work of protecting an application is done by the UltraPro Toolkit and may take just about half-an-hour!</p> <p>The Shell is most desirable when you do not have time or desire to modify the source code.</p> <p>See Chapter 6, "Using Shell and Quick Shell," on page 73 for details.</p>	<p>API elements-based protection requires a little more time and effort than Shell.</p> <p>However, the Toolkit simplifies your task to the greatest extent by generating a header file and code sketch.</p> <p>See Chapter 7, "Using API Elements," on page 89 for details.</p>

Which API Element to Choose?

Given below are few tips that will help you in deciding which API elements to use for your protection strategy. Ultimately, it is the number of elements included in your design that decide the memory size of the UltraPro keys you will be ordering.

You Must Use a Full License or Short License Element

If you are using API elements to protect your application, you must include at least one algorithm-based API element—Full License or Short License—in your design. Both the elements allow you to program algorithm words (descriptors) and activation passwords in a key. You can then make use of the query functions for enabling the query/response-based protection for your applications. Please refer to the topic “Proven Query/Response Protection” on page 10 to learn more about it.

Comparing the Full License and Short License Elements

Option	Full License	Short License
Algorithm Words with Activation Passwords	✓	✓
User Limit	✓	✓
Execution Count	✓	x
Expiration Date	✓	x
Trial Days	✓	x
Cheat Counter	✓	x
Number of Cells*	8 to 9	4 to 5

*Depends on whether a user limit is specified or not. If a user limit is not specified, 8 and 4 cells will be used for Full License and Short License, respectively.

Full License Offers Numerous Licensing Options

The Full License element allows you to program various licensing controls in the key along with the algorithm words and activation passwords.

You can provide limited number of executions (using the execution count control), lease your application (using the expiration date control), or rent it for certain period (trial days subjected to a fixed expiration date). For combating time tampering, you can specify the cheat counter value.

With dynamic marketing and business needs, it is quite probable that you might want to switch from a particular licensing model to another, without having an intent or resources to prepare a protection strategy again. With Full License, you can attain ultimate flexibility in converting licensing models without recreating a strategy. Full License offers robust licensing for your application and is suitable for almost all the scenarios—whether you are shipping your application in demo, trial or leased form, or delivering a fully-licensed version with perpetual licensing.

The algorithm words enable query/response protection in such a way that as soon as a particular control becomes invalid (for example, the lease expires or execution count reaches zero), the associated algorithm is deactivated. When the algorithm is inactive, response data is same to the query data and application does not run. At that stage, the activation passwords can be used for activating the algorithm.

Ready-to-Use Query API Functions

To simplify the process of implementing protection, an intelligent set of query API has been provided that save your time and effort in designing procedures and mechanisms in your applications's code. You just need to evaluate the response returned to control licensing of your application.

Here is a brief description of the query API functions relevant for a Full License element:

Comparing the Various UltraPro Query Functions

Function	Suitable Scenario
SFNTsntlQueryLicenseSimple Queries an algorithm. If the algorithm is active, a valid response must be returned for the query data sent.	Full License <ul style="list-style-type: none">❑ When a Full License element—without any user controls—is included in a design.❑ This function can be used more frequently than others because it checks the validity of the algorithm faster than other query functions (because it ignores the leasing and demo checks). Short License <ul style="list-style-type: none">❑ When a Short License is included in a design. This is usually the case of a fully-licensed (perpetual licensing) application.
SFNTsntlQueryLicenseLease Checks the validity of the expiration date, trial period and cheat counter in addition to querying a Full License algorithm.	When a Full License element, containing an expiration date, trial days and cheat counter, is included in a design. This is usually the case of a leased application/features. You should use it when you do not want to decrement the execution count, but want to perform the leasing and time tampering check.
SFNTsntlQueryLicenseDecrement Decrements the execution counter value by the specified amount in addition to querying a Full License algorithm.	When a Full License element—containing just an execution count—is included in a design. This function ignores the leasing check but ensures that the execution count is decremented. This is usually the case of a demo/execution-limited application/features.

Comparing the Various UltraPro Query Functions

Function	Suitable Scenario
SFNTsntlQueryLicense Checks the validity of all the controls viz. expiration date, trial period, cheat counter and execution counter in addition to querying a Full License algorithm.	<p>You should call this function when you are using all the controls—execution count, expiration date, cheat counter and trial period—provided by a Full License element.</p> <p>You should use it at least once in your code to verify the licensing checks, related to leasing, demo and time tampering.</p>

For best results, you should call the various query functions in a customized fashion in your code. For example, if you have used a Full License element with all the controls in your protection strategy, you could call the various API in the manner described below:

```
int main ()
{
/* Initialize the API Packet */
status = SFNTsntlInitialize ();

/* Obtain a license */
status = SFNTsntlGetLicense();

/* Verify the leasing controls (expiration date, trial
   days and cheat counter) and decrement the execution
   count value by one */
status = SFNTsntlQueryLicense();

/* Your Application Code */
{
.....
.....
}
```

Continued...

```
/* Frequently check the algorithm (like, after every 5
   minutes) by calling SFNTsntlQueryLicenseSimple */
status = SFNTsntlQueryLicenseSimple ();
.....
.....

/* Periodically perform the leasing checks (like, after
   every 2 hours by calling SFNTsntlQueryLicenseLease */

status = SFNTsntlQueryLicenseLease () ; /* The
   frequency can be more like 2 hours */
.....
.....

return 0;
}
```

See the topic “Tips and Tricks” on page 119 for information on how to create a secure protection strategy using the Query API.

Short License is Suitable for Fully-Licensed Applications

Short License is suitable if you are shipping your application normally (not time-limited or execution-limited). Like Full License, it makes use of the query/response mechanism to protect your applications.

It occupies a block of 4 to 5 cells consisting of algorithm descriptors and activation passwords and a user limit (optional).

You can program a user limit, if desired depending on your customer’s requirements. They may want it for certain number of users (specify a user limit) or for unlimited number of users (do not specify a user limit).

Data-Based Elements Strengthen Your Main Protection Strategy

Apart from the algorithm-based elements (Full License and Short License), you can also make use of the following data-based elements to create a customized protection strategy:

- **Counter**

A data-based API element is used to count down from a pre-programmed value. The value is decremented each time your application calls the `SFNTsntlDecrementCounter` API function. It uses a counter word in the key.

- **String**

String is a data-based API element that can contain up to 50 ASCII characters. Two characters occupy a cell.

- **Boolean**

Boolean is a data-based API element that can contain a true (1) or a false (0) flag. A single cell can accommodate 16 boolean elements.

- **Integer**

A data-based element that allows you to program the following type of integer values in the key:

- ❑ 8-bit integer: A value between 0 and 255. It occupies half a cell in the key.
- ❑ 16-bit integer: A value between 0 and 65,535. It occupies one full cell in the key.
- ❑ 32-bit integer: A value between 0 and 4,294,967, 295. It occupies two consecutive cells in the key.

Uses of the data-based elements are enormous; varying from storing the critical application data, encryption seeds to reading/writing information like, passwords and IDs. For example, a Counter element can be used to control specific functions within your application. If you associate a counter with the Save button control, you can code the application so that when the counter reaches zero, the Save button will no longer be available, preventing

the user from saving their work and making the application unusable in a practical sense.

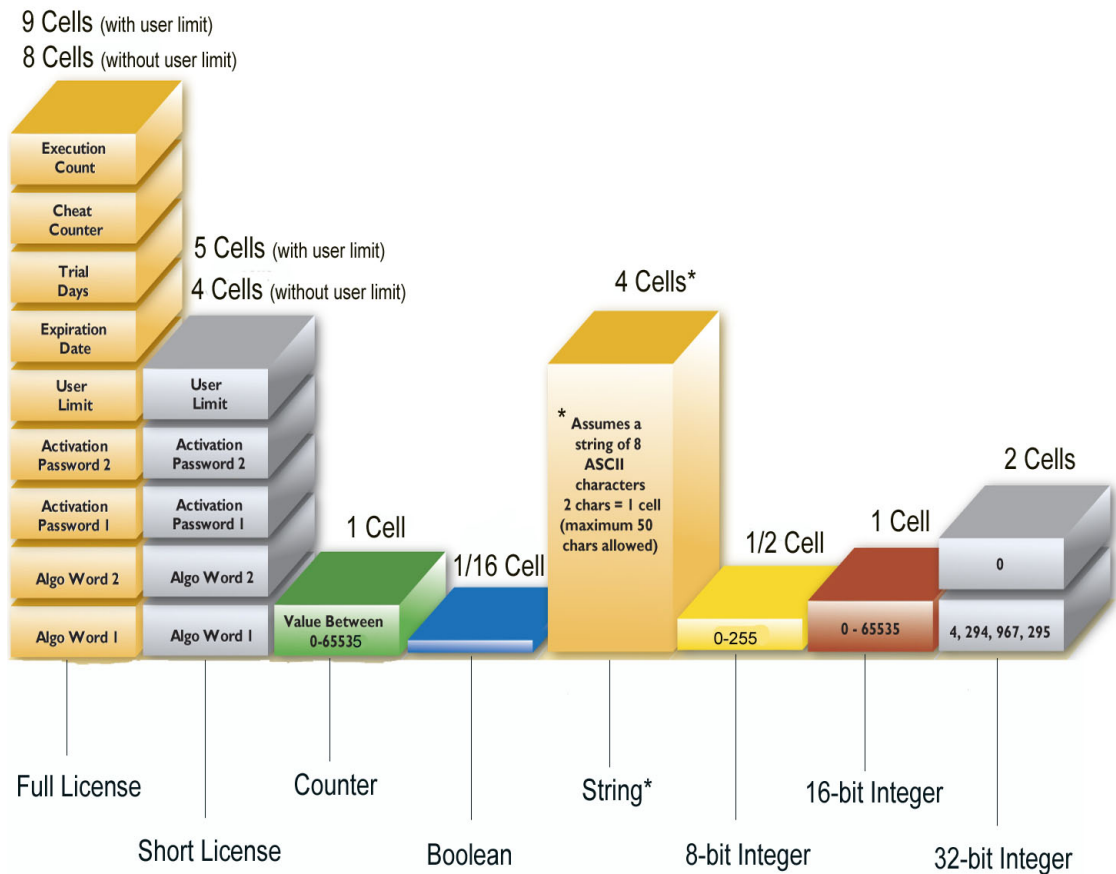
Tip: Combine a Counter with algorithm and activation passwords

If you want to create a protection strategy that combines a Counter with an algorithm, you should make use of the Full License element. When the execution counter reaches zero, the associated algorithm is automatically deactivated. As a result, the `SFNTsntlQueryLicenseDecrement` and `SFNTsntlQueryLicense` functions will return responses same as queries. Furthermore, a pair of activation passwords is also associated automatically using which the algorithm can be activated in the field.

A host of UltraPro API functions is also provided to allow manipulation of the data-based elements.

UltraPro API Functions For the Data-Based Elements

Function	Description	Suitable For
SFNTsntlReadString	Reads a string programmed into the key.	<input type="checkbox"/> String
SFNTsntlWriteString	Writes a string into the key.	<input type="checkbox"/> String
SFNTsntlDecrementCounter	Decrements the counter value by the specified amount.	<input type="checkbox"/> Counter
SFNTsntlReadValue	Reads the element values from the key.	<input type="checkbox"/> Boolean <input type="checkbox"/> Integer <input type="checkbox"/> Counter
SFNTsntlWriteValue	Writes element values into the key.	<input type="checkbox"/> Boolean <input type="checkbox"/> Integer
SFNTsntlLockData	Locks an unlocked element using the Write Password.	<input type="checkbox"/> Boolean <input type="checkbox"/> Integer
SFNTsntlUnlockData	Unlocks a locked element using the Overwrite Passwords.	<input type="checkbox"/> Boolean <input type="checkbox"/> Integer <input type="checkbox"/> Counter



A Quick Summary of the API Elements

Guarding Applications Against Time Tampering

Sentinel UltraPro allows you to protect your applications against time tampering attempts. The key stores the last known good date and time (LKDT) which is updated each time a query call is made by the application using the SFNTsntlQueryLicense and SFNTsntlQueryLicenseLease functions.

A time tampering condition is detected whenever the LKDT programmed into the key is ahead of the system’s current date and time. This indicates that the system clock has been tampered with. As a result, the following categorization of the time tampering condition is made:

Categorization of the Time Tampering Condition

Scenario	Time Tampering Category
The time gap is less than 90 minutes i.e., (LKDT minus Current Time) < 90 minutes	No time tampering condition is assumed. This leeway of 90 minutes is allowed to take account of the normal day-to-day working conditions in which the system time may slightly vary across the systems.
The time gap is more than 90 minutes but less than 30 days i.e., 30 days > (LKDT minus Current Time) > 90 minutes	Minor time tampering condition exists.
The time gap is more than 30 days i.e., (LKDT minus Current Time) > 30 days	Major time tampering condition exists.

Please note that simple time zone changes and daylight savings time changes do not cause time tampering. Also, the LKDT is always stored in the GMT (Greenwich Mean Time) system time to ignore the time zone changes.

Use of the Cheat Counter Value

To combat against the minor time tampering conditions, you can program a cheat counter value. This counter value will decrement by one each time minor time tampering is detected on a system. During this check, the LKDT stored in the key is updated to reflect the latest time.

The counter provides a kind of grace period to the application user till it reaches zero; and the minor tampering attempts are tolerated as long as the cheat counter is not exhausted. However, once the cheat counter reaches zero, the associated algorithm will be deactivated immediately. In case of major time tampering, the algorithm is deactivated immediately regardless of the cheat counter value.

You can program a cheat counter with the Full License and Shell elements. Please make sure that when you are customizing your protecting by calling UltraPro API into your source code, you should regularly call the `SFNTsntlQueryLicenseLease` or `SFNTsntlQueryLicense` functions to expose the time tampering attempts. You should also take appropriate actions, deciding the control/access to the application, once the algorithm is deactivated. The decision of locking out the application completely, or restricting some particular features (such as, the option to save) is completely yours.

In case of Shell protection, the Shell layer automatically makes checks in background to detect time tampering. Once the cheat counter reaches zero, the algorithm is deactivated and access to the application is denied.

Note: Time tampering condition in case of network applications

Please note that in case of client-server environment, when the UltraPro key is attached to a networked system, the time tampering conditions will be validated against the server-time (the system where the Sentinel Protection Server is running) and not the client-time (where the protected application is running).

Resetting the Cheat Counter

Using the field activation process, you can reset the cheat counter value remotely and restore normal access to the application. The customer will

send the locking code of the key to you/your distributor, who can generate a license code to update the key. The relevant field activation commands are:

- **Increment time tamper counter**
Increments the cheat counter value by the desired amount.
- **Update last known good date and time**
Updates the LKDT date written into the key.

In addition, the following field activation actions are also provided:

- **Decrement time tamper counter**
Decrements an existing cheat counter value in the key.
- **Decrement time tamper counter to zero**
Decrements the cheat counter value to zero.

Due to the one time update feature of the key, the license code cannot be applied more than once to correct a tampered system. However, you can always generate more license codes to update keys.

For details on the field activation process, please refer to Chapter 8, “Designing the Field Activation Strategy,” on page 103.

Allowing Field Activation Or Not?

Shipping your protected application and its corresponding key(s) to customers in the field doesn't end your control over the key and your software. With Sentinel UltraPro, you can perform a number of functions on keys already in the field, including activating and updating applications, setting or clearing bits, incrementing or decrementing counters, resetting the time tampering limits and so on.

Field activation enables you to ship your application in an unusable state, and provide a means to activate it. The activation process is protected by encryption algorithms and passwords pre-programmed into the key. It allows you to support field upgrades and control feature access.

To be able to update keys in the field, here are the main steps:

- Generate field activation objects (DUSAFE and UUSAFE).
- After adding elements in a design, define the field activation actions and commands that can be performed on those keys within your protection strategy.
- Choose the method for field activation. You can choose between Sentinel Client Activator and the Field Activation Utility or, create your own utility using the field activation functions.

Refer to Chapter 8, "Designing the Field Activation Strategy," on page 103 for details.

Which Access Mode To Use?

Whatever protection method you use—Shell or API elements—you need to decide the access mode you wish to use.

An access mode defines the boundaries within which your application will look for the key. For example, whether it will look for all the computers within the network, a specific system on the network or the local computer. For network communication, Sentinel Protection Server must be running on the system where the hardware key is attached.

Once a correct key is found by your application, various software locks are performed to acquire, maintain and release the license. You should use a mode that provides the best performance and makes the least demand on network resources.

Given below are the various access modes available with Sentinel UltraPro:

Sentinel UltraPro Access Modes

1. SP_STANDALONE_MODE
Accesses the keys attached to the local workstation directly—without using the Sentinel Protection Server. There is absolutely no need of shipping the Sentinel Protection Server along with the protected application. However, Sentinel driver must be present on the system.
2. SP_DRIVER_MODE
<div><input type="checkbox"/> Accesses the keys attached to the local workstation directly—without using Sentinel Protection Server. It is not necessary to ship the Sentinel Protection Server along with the protected application.</div> <div><input type="checkbox"/> Compared to SP_STANDALONE_MODE, this mode provides license management capabilities.</div>
3. SP_LOCAL_MODE
<div><input type="checkbox"/> Accesses the keys attached to the local workstation using the Sentinel Protection Server.</div> <div><input type="checkbox"/> If a key is not present locally—an error message is returned. Follows the client-server mode of communication. Hence, the Sentinel Protection Server must be installed on the local system.</div>

Sentinel UltraPro Access Modes

4. SP_BROADCAST_MODE
<ul style="list-style-type: none"> ❑ When invoked, the application broadcasts the license request in the subnet looking for a Sentinel Protection Server with available licenses. If none of the protection server in the subnet has a free license, an error is returned. ❑ Follows the client-server mode of communication. Hence, the Sentinel Protection Server must be shipped along with the protected application. ❑ Keep in mind, broadcast messages require additional network resources and result in a longer total time from application start-up to key acquisition. If network resources and timing is an issue for you, you may want to consider using any other suitable mode.
5. SP_ALL_MODE (3+2+4) – Default
<ul style="list-style-type: none"> ❑ It performs the same function as using SP_LOCAL_MODE followed by SP_DRIVER_MODE followed by the SP_BROADCAST_MODE mode. ❑ The Sentinel Protection Server is essential for network operations. ❑ Use this for maximum flexibility when you have a limited idea about the customer's setup. Since this mode is set by default, you need not call the SFNTsntlSetContactServer function in your source code.
6. SP_SERVER_MODE (3+4)
<ul style="list-style-type: none"> ❑ It performs the same function as using SP_LOCAL_MODE followed by SP_BROADCAST_MODE. ❑ The Sentinel Protection Server handles all the communication and manages licensing. Hence, it must be shipped along with your protected application.
7. Directed Call to a System (Name/IPaddress/IPXaddress)
<ul style="list-style-type: none"> ❑ You can direct the SFNTsntlSetContactServer function to a specific system (by its name, IP or IPX address) where the Sentinel Protection Server is running and the key is attached. ❑ Please note that if due to any reason, the system is not found or the key is not detected, the application will not send a broadcast message to the network. ❑ This mode provides the most controlled mode of operation. It prevents the application from making a search by sending a broadcast message to the entire network. Hence, this mode saves both time and network resources used in the license acquisition process. ❑ Since you cannot assume the system name of your customers in advance, you may alternatively instruct them to set the value in the configuration file or in the NSP_HOST environment variable.

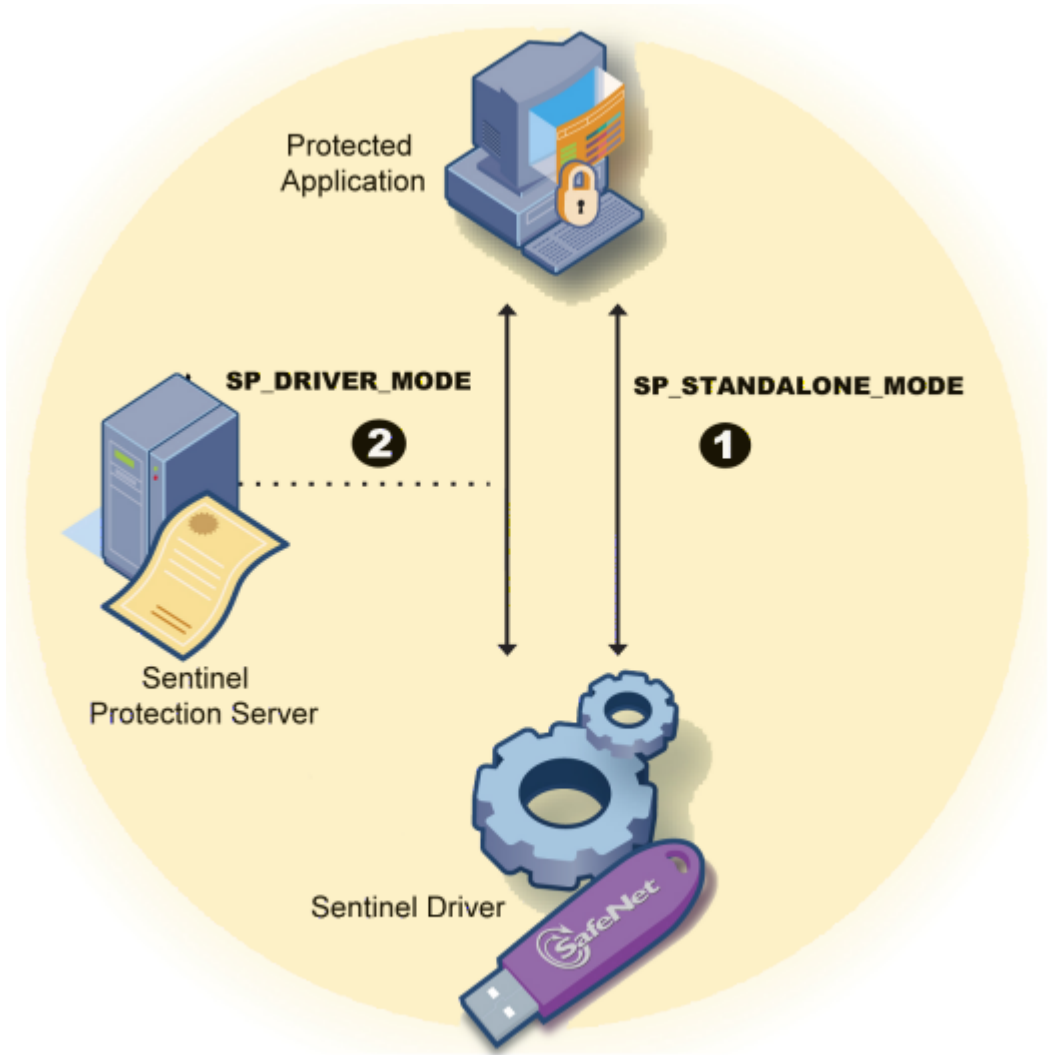
SP_STANDALONE_MODE Versus SP_DRIVER_MODE

The SP_STANDALONE_MODE mode does not use the Sentinel Protection Server to contact the key, nor does it respond to its presence on a system. SP_DRIVER_MODE, on the other hand, notifies the protection server of the license status (if a protection server is present on the system).

For example, imagine a protected application is being run using a UltraPro key with 1 hard limit. Now, if the Sentinel Protection Server is run on the system, it will always detect a free license when the access mode is SP_STANDALONE_MODE because the protection server is not notified of the license already issued. Whereas, in case of SP_DRIVER_MODE no extra license will be available.

Hence, we recommend using SP_DRIVER_MODE instead of SP_STANDALONE_MODE to manage licenses.

Refer to the diagram given on the next page for more clarity.



SP_STANDALONE_MODE Versus SP_DRIVER_MODE

How to Set an Access Mode?

In general, there are three methods to set an access mode viz., the `SFNTsntlSetContactServer` function, configuration file and the `NSP_HOST` environment variable. However, there are some differences in how an access mode is set for an API elements protected and Shell protected application.

For the API Elements Protected Applications

There are three ways to set an access mode for applications protected using UltraPro API directly:

- **Using the `SFNTsntlSetContactServer` Function**

You can call the `SFNTsntlSetContactServer` function in your application code. Refer to the Toolkit Help for detailed description of the function.

- **Using the Configuration File**

We have included a configuration file (*sntlconfig.xml*) that you can ship to your customers to set an access mode, heartbeat and protocol on each client system to guide the protected application.

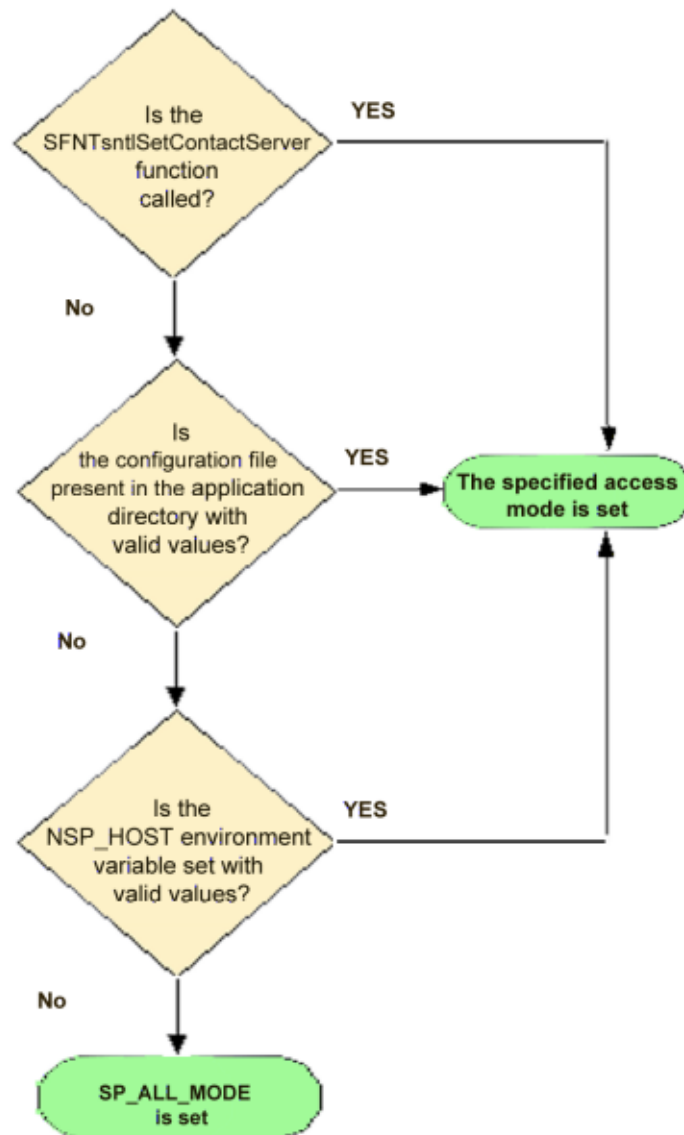
- **Using the `NSP_HOST` Environment Variable**

The `NSP_HOST` environment variable is set by your customers on the client systems where your application will be run on.

Note: About the `SP_STANDALONE_MODE` mode

Please note that the `SP_STANDALONE_MODE` access mode cannot be set using the configuration file or the `NSP_HOST` environment variable.

An access mode set through code will override an access mode set via the other two methods described below. We recommend setting an access mode through code whenever possible. See also the diagram, “How `SFNTsntlGetLicense` Works” on page 53.



Setting an Access Mode for an API Elements-Protected Application

Setting an access mode via NSP_HOST or the configuration file is optional, and unnecessary if you have set the access mode through code. However, for maximum performance system administrators may choose any of these methods. If none of the three methods is used, the application will run in SP_ALL_MODE mode.

Detailed instructions for editing the configuration file and setting the NSP_HOST environment variable can be found in the Sentinel UltraPro System Administrator's Help.

For Quick Shell and Shell Protected Applications

Toolkit provides you three access modes to choose from while you are specifying settings for a Shell element. The table below compares priority of those three choices vis-a-vis the other two methods:

Access Modes For Shell Protected Applications

Choices	Equivalent To	Comments
All Modes* (Default)	SP_ALL_MODE	Best suited for both stand-alone and network applications, specially when you are not sure whether the key will be present on the same system or on a different system within the subnet.
Server modes	SP_SERVER_MODE	Best suited for network applications. The Sentinel Protection Server is essential for communicating with the key.*
Standalone	SP_STANDALONE_MODE	Best suited for stand-alone applications. Preferred when you are sure that the key will be attached on the same system. This mode does not require you to ship the Sentinel Protection Server.**

* This mode will be overridden if a Sentinel Protection Server name—the system name/IP address/IPX address/NetBIOS name of the workstation where the key is attached—is set using the configuration file and the NSP_HOST environment variable. Barring this specific case, any access mode set in the configuration file or NSP_HOST environment variable will never have a priority over this choice.

** This will not be overridden under any case.

Specifying a User Limit

A limit, less than the hard limit of the key, which defines the maximum number of users of a protected application, is known as user limit.

The UltraPro keys come with standard hard limits (1, 2, 3, 5, 10, 25, 50 or unlimited); having a user limit helps in applying a license limit less than that. Typically, this option is used with network applications.

For example, if the key's factory-programmed limit is 25 and you need to allow for maximum 15 licenses, then stipulating a user limit equal to 15 meets your requirements exactly.

Elements That Allow Specifying a User Limit

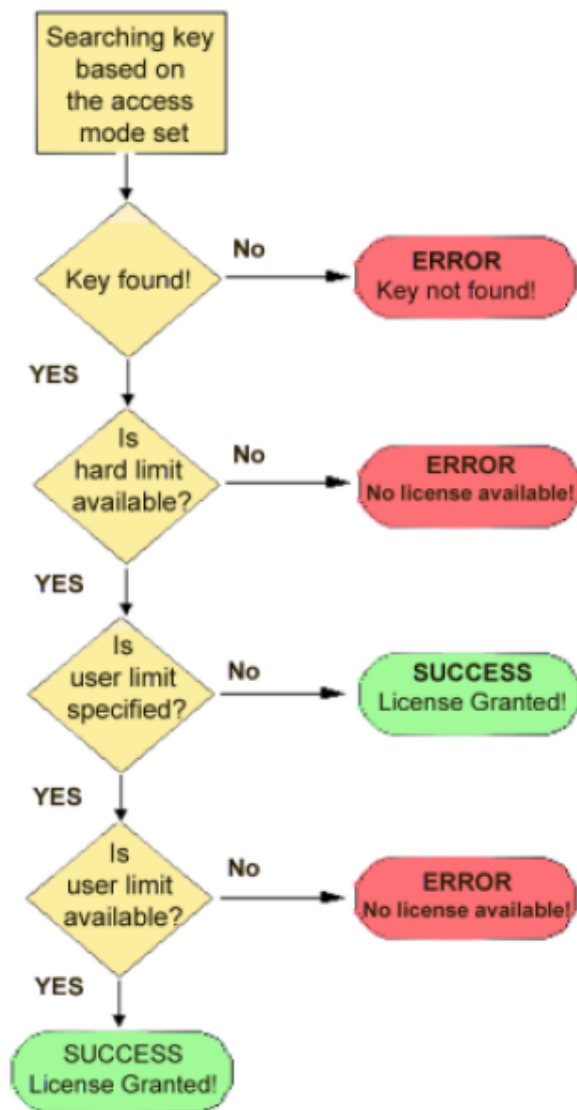
You can specify a user limit while adding the following elements:

- Shell
- Full License
- Short License

The Process of Obtaining a License

To fully-understand the user limit concept, you must learn how a license is obtained by the protected application. The flow chart on the next page describes how a license is obtained. Here is a summary:

- The UltraPro key will be searched for on the basis of the access mode.
- Once a correct key—containing the desired developer ID and design ID—is found, the availability of the hard limit is verified. If the hard limit is not available the license request fails.
- If the hard limit is available, the presence of a user limit is verified. If the user limit is not programmed, a license is granted.
- If the user limit is programmed but exhausted, the license request fails. Else, a license is granted.



The Decision Points for Obtaining a License

Please note that above description assumes that the user limit licenses are not shared. For details on user limit license sharing, refer to page 70.

Note: Display a customized message whenever a license is not available
It is up to you as to how you want to handle a client request when a user limit is unavailable. You may also guide your users for verifying the license usage/availability status using the Sentinel License Monitor.

When Multiple User Limits Are Specified

Usually a key will contain only one user limit specified using any of the element mentioned earlier. However, if you are protecting multiple applications using one key—each for same/different number of users—then multiple user limits may exist. In that case, the availability of licenses for all the application users will be dependent upon the hard limit. There must be a hard limit available—regardless of the user limit availability—before a license can be obtained.

For example, assume the hard limit of your UltraPro key is 20 licenses. If you are protecting three applications with a single key, you could use the following user limits:

- Set the license limit for application A to 10
- Set the license limit for application B to 7
- Set the license limit for application C to 10

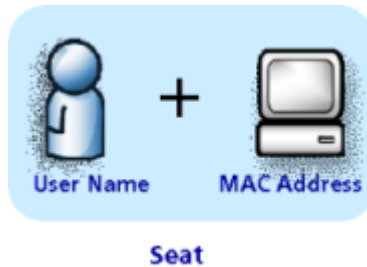
Notice that the total number of user limits (27) is greater than the hard limit (20). This means, if all the licenses for Application A are being used, only 10 hard limit is left.

If all seven licenses are being used for Application B, only three hard limit is left for Application C. Thus, even though Application C has 10 user limit available, only three licenses will be issued. This is because the hard limit is obtained first, then the user limit.

License Sharing Behavior

Sentinel UltraPro allows running multiple instances of a protected application using one license. This is applicable to all types of users (stand-alone or network) as long as the license request originate from the same *seat* (represented by a user and MAC address combination). Therefore, when a user invokes an additional instance of a protected application, the Sentinel Protection Server grants it automatically without deducting an extra hard limit from the key.

This kind of license sharing is always on and cannot be turned off.



What is a Seat?

Furthermore, if you specify a user limit in a strategy, the licenses issued can be shared too! Refer to the topic “Sharing a User Limit License” on page 70.

Sharing a User Limit License

If you have specified a user limit in your protection strategy, you can allow sharing the licenses issued on the basis of user name, MAC Address or their combination (i.e. a seat). By default, user limit license is shared for a seat.

For Shell Elements

You can specify a user limit and allow license sharing using the options available under the **Networking** tab while you add/edit a Shell element.

For API Elements

You can specify a user limit for the Full License and Short License elements. When you are calling UltraPro API in your source code, use the `SFNTsntlSetSharedLicense` API function to specify the sharing criteria. If this function is not called, a license will still be shared for the license requests originating from the same seat.

Note: The complete API documentation is available in the Toolkit Help

Please refer to the Toolkit Help for viewing the Sentinel UltraPro API documentation.

Whether Authorizing Distributors or Not?

If desired, you can authorize your sales distributors for doing the various licensing and key programming activities. A separate Key Manager application is provided using which your distributors can:

- Program keys for your customers.
- Configure Sentinel Client Activator for supporting field activation.
- Generate license codes for updating keys in the field.

You need to provide them the following items to operate:

- The Key Manager application
- A distributor (.dst) file
- A distributor key

Note: The .dst file and distributor key are a matched pair

The .dst file and the distributor key are programmed for a particular design. A distributor will not be able to use that key with any other design and vice-versa.

The details of programming keys for your distributors are provided in Chapter 12, “Programming Distributor Keys,” on page 149.

Chapter 6

Using Shell and Quick Shell

In this chapter we will:

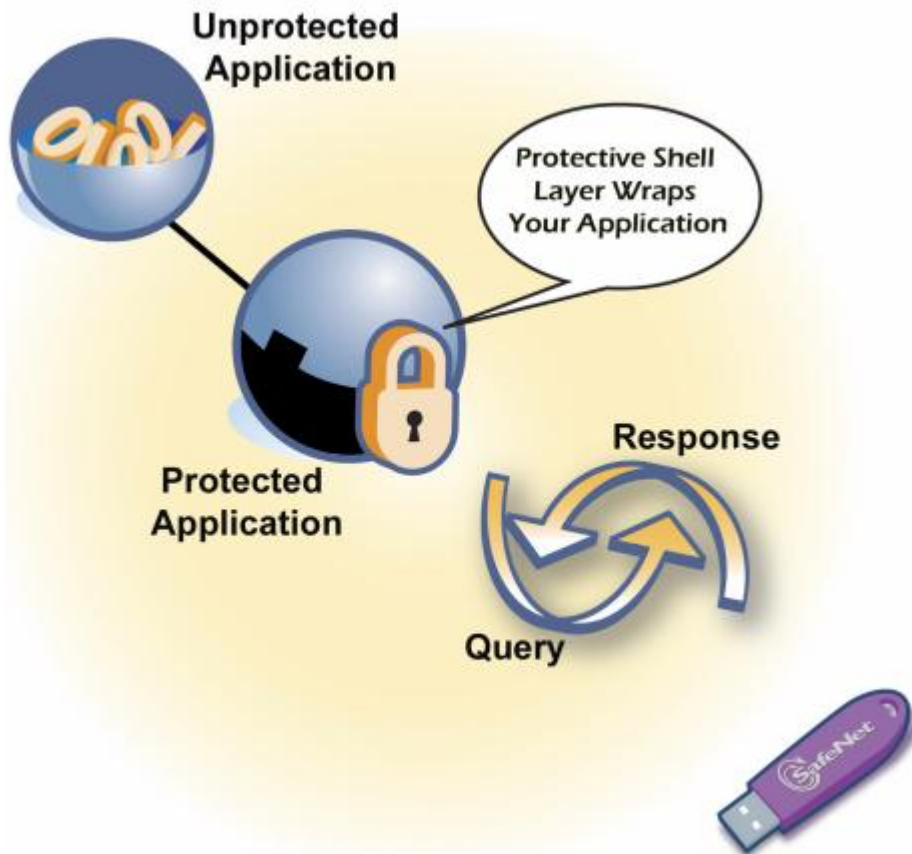
- Discuss the Shell methodology and features.
- Describe how to protect your executables using Shell and Quick Shell.

The Shell Methodology

Shell is the fastest and easiest method of protecting your executables. The entire process of protecting an application—including programming the key—can take as little as half-an-hour.

In this method, a protective Shell layer is wrapped around your application's executable while the application is strongly encrypted and concealed under it. When a user starts the application, the layer acts as a shield and takes control to verify whether all the license conditions are successfully met or not. For example, if the correct UltraPro key is not attached or has been tampered with, the user sees an error message and the application does not run.

At runtime, the Shell layer consistently makes query calls to check the validity of all conditions you stipulated. So, if the key is missing at any time, the application shuts down immediately.



Shell Protection In Action

Shell is Automatic and Secure

Shell is automatic because it does not require you to modify the source code of your application. You just require a copy of your application's executable file to protect the application. You need not bother to add query/response values to your source code—the Toolkit does everything for you! What's more, you can customize your protection by choosing your own algorithm

values. Shell is secure because of the numerous and random checks it makes that foil hacking attempts.

Shell Offers a Range Of Licensing Options

Using Shell's intuitive user interface, you can effortlessly incorporate the following exciting licensing options in your protected application:

- Expiration date for leasing your application.
- Trial period for offering days-limited demos.
- Execution count to limit the number of times the application can be run.
- Cheat counter value to detect malicious time tampering on a system and deny access to the application after certain attempts. Later, using our field activation process, you can resume access to the protected application—without shipping a new hardware key or visiting the customer's site.
- User limit to restrict the number of users running the protected application. The licenses issued against the user limit can be shared on the basis of user name, MAC address or seat.
- Encrypt the external files associated with your protected application. These files can be associated both during the:
 - Shell time (for example, overlays, data files, DLLs or file types other than the executable).
 - Run-time (for example, the files generated by the protected application).

Shell Versus Quick Shell

Quick Shell, shown on the introductory screen, is a gateway to the UltraPro Toolkit. We recommend it for new users who want to get familiar with UltraPro right-away. You can use it for quickly protecting an executable with basic controls, like expiration date and execution count.

Consider using the **Shell** option under **Protection Manager** to make use of advanced options, like protecting multiple files at one time, encrypting data files, and customizing the Shell algorithm values.

Both methods yield secure and convenient protection.

Using Quick Shell

Attach and Configure a Key

Before you proceed with Quick Shell, make sure that you have attached and configured a key. The exact instructions are provided in the Help. As soon as the key is configured properly, the **Prepare Key** button is enabled.

The screenshot shows the 'Quick Shell' configuration window. At the top, a title bar is followed by the text: 'Quick Shell is a fast and simple method to protect your executables. It offers a limited set of features and only a basic level of protection. For advanced Shell protection, try Shell available under Protection Manager.'

The window is divided into three main sections, each with a numbered tab on the left:

- 1 Licensing options:** This section contains two radio buttons. The first is 'Protect my application without any licensing limit.' with the label '(Perpetual licensing)' to its right. The second is 'Protect my application with the licensing limit I choose from below.' with the label '(Limited licensing)' to its right. Below the second option are three checkboxes: 'Expiration date:' with a text box containing 'No expiration', 'Trial period (days):' with a text box containing 'No trial period', and 'Execution count:' with a text box containing 'Unlimited'. A 'Prepare Key' button is located at the bottom right of this section.
- 2 Application settings:** This section contains a 'Source:' label followed by a text box and a 'Browse...' button. Below that is a 'Destination:' label followed by a checked checkbox 'Different from source', a text box, and a 'Browse...' button. At the bottom left is an 'Access mode:' label followed by a dropdown menu showing 'All modes'. At the bottom right is the text 'Application not defined' with a red error icon and a 'Make Shell' button.
- 3 Make key:** This section contains the text 'Toolkit is not ready to program the hardware key.' and a 'Make key' button.

At the bottom of the window, there are two buttons: 'Reset' on the left and 'Save...' on the right.

The Quick Shell Screen

Choosing the Licensing Options

You need to decide whether you want any licensing options or not.

Without Licensing Options - Perpetual License

You can protect your application without any licensing limit (such as, an expiration date or an execution count). As a result, the protected application will never cease to run as long as the key is attached.

1. In the **Quick Shell** screen, select the **Protect my application without any licensing limit** option.
2. Click **Prepare Key**. When complete, move to the next step of providing the application settings.

With Licensing Options - Limited License

In addition to protecting your application, you can choose how long you want the license to last.

1. In the **Quick Shell** screen, select the **Protect my application with the licensing limit I choose from below** option.
2. Specify value(s) for the desired controls. You can choose from an expiration date, trial days, and an execution count. If you specify two or more options, the application will expire as soon as any limit is reached.
3. Click **Prepare Key**. When complete, move to the next step of providing the application settings.

Choosing the Application Settings

Please note that the options discussed below will remain disabled unless you have completed the last step successfully.

1. In the **Source** field, type the path of the unprotected file or, use the **Browse** button to select the file.

To avoid overwriting the original file, the **Different from source** check box is selected by default. The protected file will be created at the same path with “_SHELLED” suffix.

2. Specify an access mode for your application. By default, **All modes** is chosen. It is best suited for both stand-alone and network applications.
3. Click the **Make Shell** button to apply the Shell protection. When complete, you can move to the next step of programming the key.

Programming Key and Saving the Design

Please note that the options discussed below will remain disabled unless you have completed the last step successfully.

1. Click the **Make Key** button to program the selected key (shown in the **Key Status** pane).
2. Click the **Save** button to save the Quick Shell design. For detailed instructions, on saving a Quick Shell design, refer to the Help.

You can add more elements in the saved design. You can edit its settings under the **Shell** tab on the **Protection Manager** screen. Or, use the saved design for programming more keys.

The **Reset** button will clear the settings you provided (without saving them).

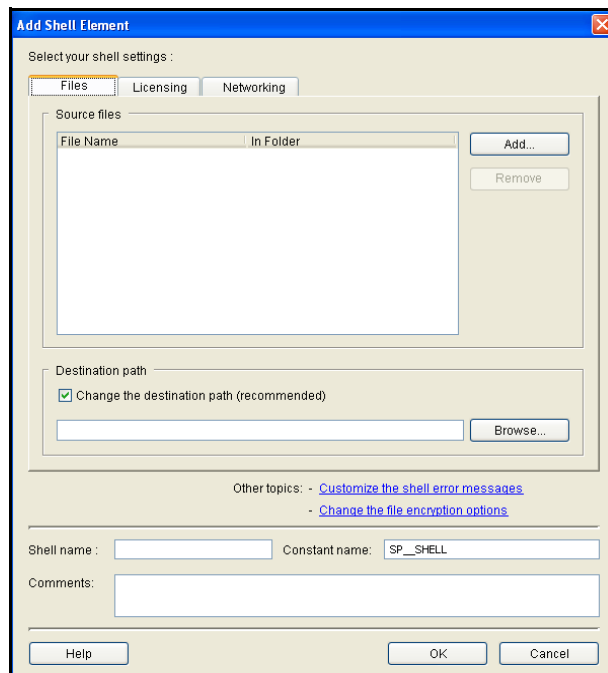
Using Shell

You can add Shell elements under the **Shell** tab on the **Protection Manager** screen. The steps below assume that you have already created a project and a design in which the Shell element will be added.

Adding Files

To begin, you must add file(s) for Shell protection. It is a compulsory step.

1. In the **Protection Manager** screen, select the design under which the Shell element will be added.
2. Click the **Shell** tab.
3. Click the **Add** button. The **Add Shell Element** dialog box appears.



The Add Shell Element Dialog Box

4. Under the **Files** tab, click the **Add** button to add the source files. You must add at least one *.exe* or a *.dll* to proceed.

In general, you can add as many files as you like, from various directories, including any data files. The Shell layer will be applied to *.exes* and *.dlls*. All other files will be encrypted at Shell-time and can only be decrypted on run-time using the protected application.

You can modify the file encryption settings using the **Change the file encryption** option described on page 83.

5. Type or browse the path of the destination directory in the edit box under the **Change the destination path** check box (selected by default). This is the destination path for writing the output files.

We recommend you to always specify a different path for generating the output files. If you do not perform this step, the output files will overwrite the original files once the **Make Shell** button is clicked.

Tip: See the Troubleshooting topic of the Toolkit Help

The Troubleshooting and FAQs section of the Toolkit Help includes more information on what types of files can be protected using Shell.

Providing Licensing Settings

You may like to modify the licensing settings under the **Licensing** tab. Otherwise, the default values will be applied.

1. Click the **Licensing** tab in the **Add Shell Element** dialog box.
2. Specify licensing settings using the information provided in the following table:

Licensing Settings for a Shell Element

Option	Default Setting	Can the Value be Overridden During Key Programming?
Expiration Date	No expiration.	Yes
Trial Days	Unlimited trials.	Yes
Cheat counter	Zero – implies that access to the application will be denied when minor time tampering is detected for the first-time on the system.	Yes
Execution Count	Unlimited number of executions.	Yes
Algorithm 1 and 2*	The Toolkit randomly generates the values that keep the algorithm active. Though, you can provide your own values, we recommend you to accept the default algorithm values. Note that if the values you specify make the algorithm inactive, then the Toolkit will prompt you to specify appropriate values.	No**
Activation Password 1 and 2*	The Toolkit randomly generates the values. However, you can provide your own values, if desired.	No

* This option will be shown if the **Show low-level algorithm data for design elements** check box on the **Miscellaneous** tab of the **Settings** menu is selected.

** At the time of programming keys, you/your distributor can make the algorithm inactive. As a result, the application will not run on the customers' site unless the algorithm is activated in the field. See the Chapter 11, "Programming Product Keys," on page 139 for more details.

Note: When you choose multiple execution controls...

If you specify two or more execution controls—the expiration date, trial days, and execution count—the application will expire as soon as any of these limits is reached.

Whatever licensing settings you specify here will be programmed in your product keys unless overridden during the key programming. To allow overriding these values, select the **Override** check box and provide a maximum value beyond which the value cannot be modified. This way you/your distributors can easily modify the element values without re-creating an element or a design.

Providing Network Settings

You can provide the networking settings for your application, such as user limit, access mode and the license sharing criteria under the **Networking** tab. Otherwise, the default values will be applied.

- 1. Click the **Networking** tab in the **Add Shell Element** dialog box.
- 2. Choose options using the information provided in this table:

Network Settings for a Shell Element

Option	Default Setting	Can the Value be Overridden During Key Programming?
User limit	No user limit – the hard limit programmed into the key will decide the number of users for a key on the customer's site.	Yes
Access modes	All Modes – equivalent to SP_ALL_MODE.	No
Sharing Criteria	Seat sharing	No

Note: Access mode for Shell and Quick Shell protected applications
For details on setting the access modes for Shell and Quick Shell protected applications, refer to the topic “For Quick Shell and Shell Protected Applications” on page 66.

Customizing the Shell Error Messages

You can customize the Shell run-time error messages, so that you can display message text of your own choice. This is an optional step and the default text messages will be shown if you do not modify them.

1. Click the **Customize the Shell error messages** link in the **Add Shell Element** dialog box. The **Edit Shell error messages** dialog box appears.
2. In the edit box, write your message text. It can contain up to 200 characters. You must review the existing messages before deciding to modify them. Afterward, you may select an error message from the list for modification.
3. If you are not satisfied with your message text, click **Restore** to accept the original message. Clicking **Restore All** will replace all the customized messages with their original text strings.
4. Click **OK**. You return to the **Add Shell element** dialog box.

Changing the File Encryption Options

You can specify choices for encrypting the external files—other than the executables and DLLs—such as, the data files. Your protected application will automatically and transparently decrypt these files at run-time, as needed.

When not-in-use these files remain encrypted. If your application creates one of these files, it will be decrypted only if the correct UltraPro key is being used to run the application.

This step is optional for adding a Shell element and by default all files—other than *.exes* and *.dlls*—you selected under the **Files** tab will be encrypted.

Note: Sentinel Data Protection Driver required on Windows 98 and ME

If you are protecting data files for Windows 9x applications, your customers need to install the data protection driver in addition to other redistributables. See “Deploying Protected Applications” on page 163 for details.

1. Click the **Change the file encryption options** link in the **Add Shell Element** dialog box. The **Shell file encryption settings** dialog box appears.
2. Select the **Encrypt/decrypt the following files at run-time** check box.
3. Type the file extensions in the edit field. This field can contain up to 255 characters.

You can filter files using the wildcard symbols asterisk (*) and semicolons (;). The 8.3 naming convention is followed, meaning that the file name and extension cannot exceed 8 and 3 characters, respectively. File filters, like *.* , *name.* are not acceptable. Given below is a description of the valid conventions:

Conventions for Specifying Data Files

Convention	Example
<i>*.extension</i>	<i>*.txt</i> will encrypt/decrypt files, like <i>settings.txt</i> , <i>install.txt</i> and so on.
<i>name*.*</i>	<i>safe*.*</i> will encrypt/decrypt files, like <i>safeobj.htm</i> , <i>safenet.cpp</i> and so on.
<i>name*.extension</i>	<i>safe*.txt</i> will encrypt/decrypt files, like <i>safeobj.txt</i> , <i>safenet.txt</i> and so on.

4. The Toolkit automatically generates an encryption seed. If you want to provide your own encryption seed, select the **Specify my own encryption seed** check box and edit the value. Please note that if the encrypted data files are shared by multiple applications, all the applications must use the same encryption seed. The seed can consist of two strings of 14 hex characters each (forming a 112-bit long seed).
5. Click **OK**. You return to the **Add Shell element** dialog box.
6. Click **OK** again to move back to the **Protection Manager** screen where you can find the Shell element added under the **Shell** tab.

The Final Steps for Shell Protection

After having specified your choices in the **Add Shell element** dialog box. You must now perform the following final steps:

Use the Make Shell Button

Click the **Make Shell** button. This will wrap the Shell layer around your executables and DLLs and encrypt the data files (if any). This may take time depending upon the number of files you have added. Once the Shell layer is applied, all the licensing and networking you defined are implemented to the application (output files at the destination path).

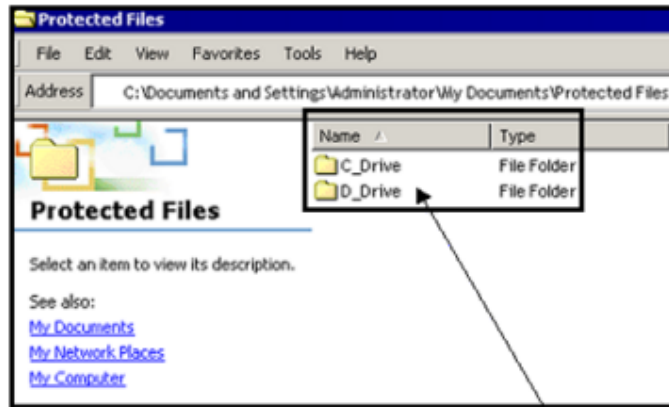
Where Can I Find the Output Files?

If you had cleared the **Change the destination path** check box under the **Files** tab, your original files were overwritten during the Shell process. Your source files are replaced by the output files.

Otherwise, your protected files were written at the path specified in the **Change the destination path** edit box.

Please note that if you had selected multiple files from different disk drives on your system, then the protected files will be copied to the individual directories identified by the appropriate drive names and path names. See an example on the next page for understanding.

This is to ensure that files carrying same name, even if selected from different path, are not overwritten during the Shell process and their source path can be tracked easily.



Protected files are written into a replicated directory structure at the destination path you chose.

An Example: Destination Path for Files Selected From the C: and D: Drives

Where to Go Next?

After protecting files using Shell, you can decide your next step from any of the following topics:

- **Add API Elements**

If you want to add API elements, refer to Chapter 7, “Using API Elements,” on page 89.

- **Add Field Activation Actions**

If you want allow field activation for the application protected using the Shell element, you can field activation actions under the **Remote Updates** tab. Please refer to Chapter 8, “Designing the Field Activation Strategy,” on page 103 for details.

- **View the Key Layout**

You can use the **Key Layout** tab to view how does Shell elements fits

in the key. Refer to Chapter 10, “Implementing the Application Protection,” on page 129 for more information.

■ **Use the Build Button**

If you are through with the application protection task, and don’t want to add any more elements, you can click the **Build** button to update your design settings and program the key with your design. However, before that you must set the options provided under the **Build Options** tab. Refer to the topic “Prototyping Design Using the Build Button” on page 130 for details.

■ **Program Keys in Key Manager**

If you are through with the application protection, you can proceed with programming keys for your customers and distributors using the **Key Manager** screen. The Part 3 of this guide provides details on the each topic.

■ **Configure the Sentinel Client Activator Wizard**

If you are through with protecting your application and want to make use of the Sentinel Client Activator to allow field activation, refer to the Sentinel Client Activator documentation available at the following path: *<installdir>\SafeNet Sentinel\UltraPro\<UltraPro version>\Sentinel Client Activator*. Also refer to the topic “Client Activator Versus Field Activation Utility” on page 107 of this guide.

Chapter 7

Using API Elements

In this chapter we'll be getting familiar with the various API elements offered by Sentinel UltraPro.

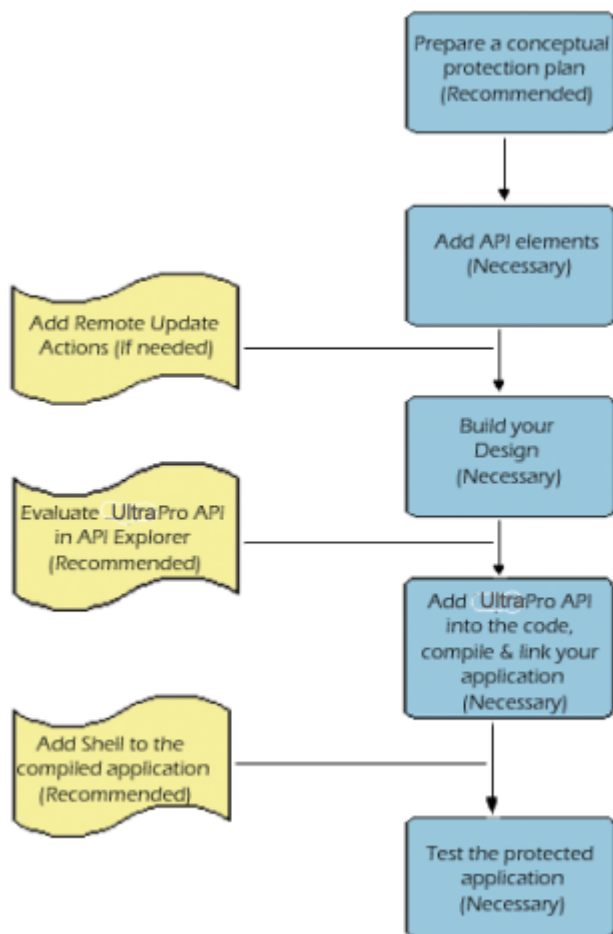
About the API Elements-Based Protection

Under the API elements based protection, you add software locks—API functions to verify the presence of the key—directly into your application's source code. You control the amount and location of the locks. The frequency of software locks within your application, and the action taken if no key is found, is left up to you. The more locks you add to your application, the more difficult it will be for potential hackers to break your application's protection. It allows you to take advantage of a number of protection techniques, both basic and advanced, using those that work best for you.

Because you must understand the API calls used to support the protection strategy you have designed, and manually add them to your code, using API elements may take longer than Shell. It is most commonly used when:

- You want to have control over the protection techniques used to secure your application.
- You have access to the source code and understand the API functions.

The diagram below identifies the various stages involved in the API elements based protection:



Stages of the API Elements-Based Protection

Tip: **Where are the API elements located in the Toolkit?**

After creating a design using the **Protection** wizard in Toolkit (can be without any elements), navigate to the **Protection Manager** screen, and click the **API** tab. To create an API element under the design, click **Add**.

Adding Full License

Full License is an algorithm-based element that allows you to design and customize robust licensing strategies along with the powerful query/response functionality—providing you the best of the licensing strategies.

Options Available in Full License

Option in Full License	Strategies
Expiration Date	Lease applications
Trial Period	Offer days-limited demos/trial applications valid up to an expiration date
Cheat Counter	Apply anti-time tampering measures for leased/ days-limited applications
Number of Executions	Limit the application executions
User Limit	Specify and share a license limit less than the hard limit of the key (known as the user limit)
Algorithm and Activation Passwords	Create secure query/response-based protection in combination to the various licensing options.

Tip: Relevant API functions for a Full License element

Here are a few API functions that you may use for a Full License element: SFNTsntlQueryLicense, SFNTsntlQueryLicenseLease and SFNTsntlQueryLicenseDecrement.

Given below are the steps for adding a Full License element:

1. In the **Protection Manager** screen, select the required project.
2. Select the design under which the Full License element will be added.
3. Click the **API** tab.
4. Click the **Add** button. The **Add Element** dialog box appears.

5. Select **Full License**. The Full license options appear in the right-side panel. Choose the options using the information given in this table:

The Full License Settings

Option	Default Setting	Can the Value be Overridden during Key Programming?
Expiration Date	No expiration	Yes
Trial Days	Unlimited trials	Yes
Cheat counter	Zero (therefore, access to the application will be denied when minor time tampering is detected for the first-time on the system)	Yes
Execution Count	Unlimited number of executions	Yes
User Limit	No user limit (the hard limit programmed into the key decides the number of users allowed on the customer's site)	Yes
Algorithm 1 and 2*	The Toolkit randomly generates the values that keep the algorithm active. Though, you can provide your own values, we recommend you to accept the default algorithm values. Note that if the values you specify make the algorithm inactive, then Toolkit will prompt you to specify appropriate values.	No**
Activation Password 1 and 2*	The Toolkit randomly generates the values. However, you can provide your own values, if desired.	No

* This option will be shown only if the **Show low-level algorithm data for design elements** check box, available under the **Miscellaneous** tab of the **Settings** menu, is selected.

** At the time of programming keys, you/your distributor can choose to make the algorithms inactive. As a result, the application will not run on the customers' site unless the algorithms are activated. See the Chapter 11, "Programming Product Keys," on page 139 for more details.

Note: When you choose multiple execution controls...

If you choose two or more execution controls—the expiration date, trial days and execution count—the application will expire as soon as any of these limits is reached.

Whatever values you specify here will be programmed in your product keys unless modified right before key programming. If you want to allow overriding these values, select the **Override** check box and provide a maximum value beyond which the value cannot be modified. This way you/your distributors can easily modify the element values without recreating an element or a design.



Warning! Regarding the Override option

The option of overriding the element settings is unrestricted. This means that all the Toolkit and Key Manager users will be able to modify the elements settings while programming the keys. If you do not want them to use this option, you must unselect the **Override** check box. Furthermore, if you are using the existing element values in your code for some verification-purposes, then make sure that you do not allow overriding it. This is because, the value will be overridden in the key **ONLY** and not in your application's code.

6. Provide a name for this element (necessary). It can consist of 20 alphanumeric characters.
7. The constant name will be automatically generated. However, you may modify it if needed.
8. You may optionally provide **Comments** for this element.
9. Click **OK**.
10. Click the **Build** button to update your design. For more details, refer to Chapter 10, “Prototyping Design Using the Build Button,” on page 130.

Adding Short License

Short License is an algorithm-based element that is suitable for protecting full featured applications—that are not time or execution limited.

Tip: Relevant API functions for a Short License element
You can use the SFNTsntlQueryLicenseSimple function for a Short License element.

Given below are the steps for adding a Short License element:

- 1. In the **Protection Manager** screen, select the required project.
- 2. Select the design under which the Short License element will be added.
- 3. Click the **API** tab.
- 4. Click the **Add** button. The **Add Element** dialog box appears.
- 5. Select **Short License**. The Short license options appear in the right-side panel.Choose the options using the information given in this table:

The Short License Settings

Option	Default Setting	Can the Value be Overridden During Key Programming?
User Limit	No user limit (the hard limit programmed into the key alone decides the number of users allowed on the customer's site)	Yes

The Short License Settings

Option	Default Setting	Can the Value be Overridden During Key Programming?
Algorithm 1 and 2*	The Toolkit randomly generates the values that keep the algorithm active. Though, you can provide your own values, we recommend you to accept the default algorithm values. Note that if the values you specify make the algorithm inactive, then Toolkit will prompt you to specify appropriate values.	No**
Activation Password 1 and 2*	The Toolkit randomly generates the values. However, you can provide your own values, if desired.	No

* This option will be shown only if the **Show low-level algorithm data for design elements** check box, available under the **Miscellaneous** tab of the **Settings** menu, is selected.

** At the time of programming keys, you/your distributor can choose to make the algorithms inactive. As a result, the application will not run on the customers' site unless the algorithms are activated. See the Chapter 11, "Programming Product Keys," on page 139 for details on the key programming options.

Whatever user limit you specify here will be programmed in your product keys unless modified right before key programming. If you want to allow overriding the user limit value, select the **Override** check box and provide a maximum value beyond which the value cannot be modified. This way you/your distributors can easily modify the element values without recreating an element or a design.

6. Provide a name for this element (necessary). It can consist of 20 alphanumeric characters.
7. The constant name will be automatically generated. However, you may modify it if needed.
8. You may optionally provide **Comments** for this element.

9. Click **OK**.
10. Click the **Build** button to update your design. For more details, refer to Chapter 10, “Prototyping Design Using the Build Button,” on page 130.

Adding String

String is a data-based API element that can contain up to 50 ASCII characters. Each character occupies a single cell.

Tip: Relevant API functions for the String element

Here is the list of the API functions that you may use for a String element: SFNTsntlReadString, SFNTsntlWriteString, SFNTsntlLockData and SFNTsntlUnlockData.

Given below are the steps for adding a String element:

1. In the **Protection Manager** screen, select the required project.
2. Select the design under which the String element will be added.
3. Click the **API** tab.
4. Click the **Add** button. The **Add Element** dialog box appears.
5. Select **String**. The String options appear in the right-side panel.
6. Type a string. It must not exceed 50 ASCII characters.
7. Whatever string value you specify here will be programmed into your keys at the time of key programming. If you want to override it right before programming your product keys, select the **Override** check box. Also, specify the maximum number of characters that can be set at the key programming time—though it cannot exceed 50 in any case.

- 8. Provide a name for this element (necessary). It can consist of 20 alphanumeric characters.
- 9. The constant name will be automatically generated. However, you may modify it if needed.
- 10. You may optionally provide **Comments** for this element.
- 11. Click **OK**.
- 12. Click the **Build** button to update your design. For more details, refer to Chapter 10, “Prototyping Design Using the Build Button,” on page 130.

Adding Boolean

Boolean is a data-based API element that can contain a true (1) or a false (0) flag. A single cell can accommodate 16 boolean flags.

An element that can contain a true (1) or a false (0) flag.

T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

A Cell Can Accommodate 16 Boolean Flags

Tip: Relevant API functions for the Boolean element

Here are a few API functions that you may use for a Boolean element: SFNTsntlReadValue, SFNTsntlWriteValue, SFNTsntlLockData and SFNTsntlUnlockData.

Given below are the steps for adding a Boolean element:

- 1. In the **Protection Manager** screen, select the required project.

2. Select the design under which the Boolean element will be added.
3. Click the **API** tab.
4. Click the **Add** button. The **Add Element** dialog box appears.
5. Select **Boolean**. The **Boolean options** appear in the right-side panel.
6. Specify a boolean value: **True** or **False**.
7. Whatever value you specify here will be programmed into the key at the time of key programming. If you want to override it right before programming your product keys, select the **Override** check box. This way you/your distributors can easily modify the element value without recreating an element or a design.
8. Provide a name for this element (necessary). It can consist of 20 alphanumeric characters.
9. The constant name will be automatically generated. However, you may modify it if needed.
10. You may optionally provide **Comments** for this element.
11. Click **OK**.
12. Click the **Build** button to update your design. For more details, refer to Chapter 10, “Prototyping Design Using the Build Button,” on page 130.

Adding Counter

Counter is a data-based API element used to count down from a pre-programmed value. The value is decremented each time your application calls the SFNTSntlDecrementCounter API function. It uses a counter word in the key.

Tip: Relevant API functions for the Counter element

Here is the list of the API functions that you may use for a Counter element: SFNTsntlDecrementCounter and SFNTsntlReadValue.

Given below are the steps for adding a Counter element:

1. In the **Protection Manager** screen, select the required project.
2. Select the design under which the Counter element will be added.
3. Click the **API** tab. A list of the API elements already existing might be shown.
4. Click the **Add** button. The **Add Element** dialog box appears.
5. Select **Counter**. The **Counter options** appear in the right-side panel.
6. Specify a counter value between 0 and 65,535.
7. Whatever value you specify here will be programmed into the key at the time of key programming. If you want to override it right before programming your product keys, select the **Override** check box. This way you/your distributors can easily modify the counter value without recreating an element or a design. If you do not want to allow overriding this element's values at the time of key programming, move to step 10.
8. If you have selected the **Override** check box, provide the minimum value that can be written during the key programming stage. The minimum value has to be equal to or less than the original value you specified in step 6.
9. If you have specified the minimum value, also specify a maximum value that can be written during the key programming stage. The maximum value has to be equal to or greater than the original value you specified in step 6. However, it must not exceed 65535.

10. Provide a name for this element (necessary). It can consist of 20 alphanumeric characters.
11. The constant name will be automatically generated. However, you may modify it if needed.
12. You may optionally provide **Comments** for this element.
13. Click **OK**.
14. Click the **Build** button to update your design. For more details, refer to Chapter 10, “Prototyping Design Using the Build Button,” on page 130.

Adding Integer

An Integer element allows you to program the following types of integer values in the key:

- 8-bit integer: A value between 0 and 255. It occupies a half cell in the key.
- 16-bit integer: A value between 0 and 65,535. It occupies one full cell in the key.
- 32-bit integer: A value between 0 and 4,294,967,295. It occupies two consecutive cells in the key.

Tip: Relevant API functions for the Integer element

Here is the list of the API functions that you may use for a Integer element: SFNTsntlReadValue, SFNTsntlWriteValue, SFNTsntlLockData and SFNTsntlUnlockData.

Given below are the steps for adding a Counter element:

1. In the **Protection Manager** screen, select the required project.
2. Select the design under which the Integer element will be added.
3. Click the **API** tab.

4. Click the **Add** button. The **Add Element** dialog box appears.
5. Select **Integer**. The **Integer options** appear in the right-side panel.
6. Choose from the following options: **8-bit integer**, **16-bit integer** or **32-bit integer**.
7. Specify a value in the **Integer value** field.
8. Whatever values you specify here will be programmed into your keys at the time of key programming. If you want to override it right before programming your product keys, select the **Override between the range specified below** check box. If you do not want to allow overriding this element's values at the time of key programming, move to step 10.
9. If you have selected the option specified in the last step, provide the minimum value that can be written during the key programming stage. The minimum value has to be equal to or less than the original value you specified in step 7.
10. If you have specified the minimum value, also specify a maximum value. The maximum value has to be equal to or greater than the original value you specified in step 7. However, it must not exceed maximum allowed value for that integer type.
11. Provide a name for this element (necessary). It can consist of 20 alphanumeric characters.
12. The constant name will be automatically generated. However, you may modify it if needed.
13. You may optionally provide **Comments** for this element.
14. Click **OK**.
15. Click the **Build** button to update your design. For more details, refer to Chapter 10, "Prototyping Design Using the Build Button," on page 130.

Where to Go Next?

- **Use the Key Layout Option**

After having added the various elements in your design, click the **Key Layout** tab to view the virtual arrangement of the selected design in the key. For more details on using the Key Layout option, go to page 133.

- **Add Field Activation Actions**

If you want allow field activation for the application protected using the API elements, you can field activation actions under the **Remote Updates** tab. Please refer to Chapter 8, “Designing the Field Activation Strategy,” on page 103 for details.

- **Build the Design**

You must build your design once you are through with adding API elements.

- **Add UltraPro API into your Source Code**

Once you have built your design, you can proceed with adding the Sentinel UltraPro API into your application’s source code. For tips and tricks that can help you in implementing a secure solution, refer to Chapter 9, “Tips on Maximizing Application Security,” on page 117.

- **Add Shell to your Compiled Application**

You can always add Shell protection to your compiled application. However, make sure that the settings you specify in Shell—such as access mode and number of executions—do not conflict with that of Integrated protection. Therefore, always test your protected application before shipping it to customers.

Chapter 8

Designing the Field Activation Strategy

This chapter explains how to design the field activation strategy for your protected applications. Using the field activation process you can provide an opportunity to your customers to subscribe additional features and updates in the field on demand.

Tip: You could skip this chapter if...

If you are not allowing remote activation facility to your customers, you may skip this chapter. Here we'll mainly discuss the use of the **Remote Actions** tab.

What Is Field Activation?

The field activation feature of Sentinel UltraPro allows you to ship inactive applications/features to your customers in the field and provide them an option to activate remotely.

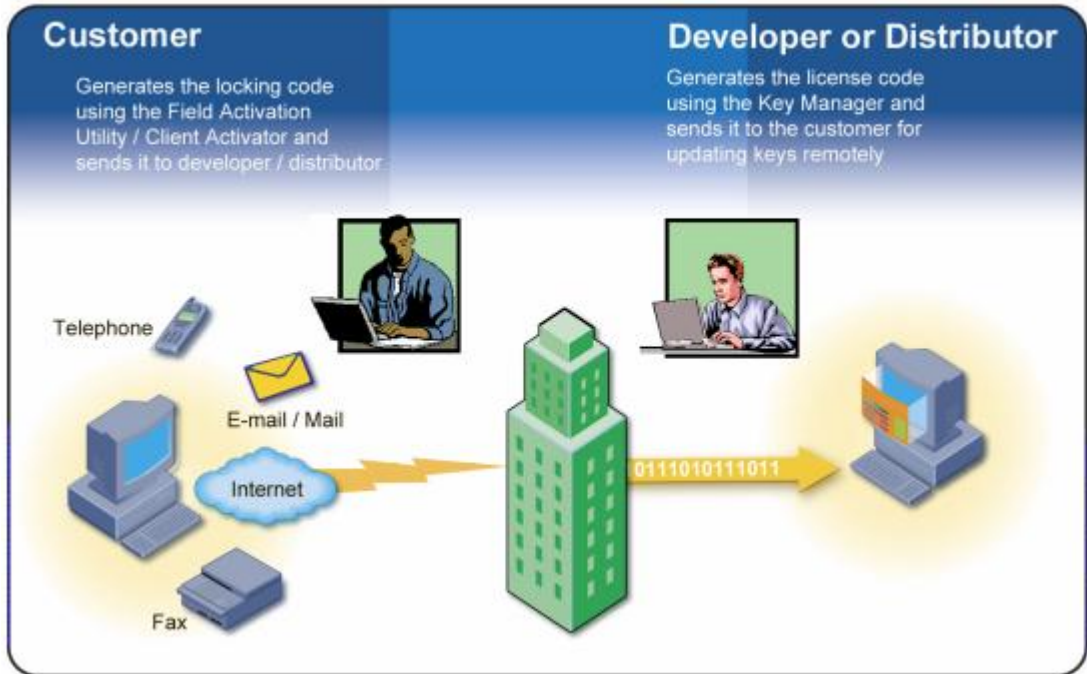
Using the field activation process you can activate applications, increase demo limits, upgrade demos to fully-licensed versions and provide access to additional features all without having to ship a new hardware key, additional software or visit the customer's site. All this can be done using an e-mail, a fax, or a phone call.

The Field Activation Process

Using the field activation feature of Sentinel UltraPro, you can ship an inactive (full/partial) application to your customers and provide an option to activate it remotely. Thus, you can control the application use from a distance and support personalized marketing programs securely with little effort.

Here is a brief description of the entire process:

1. Create a pair of field activation objects (DUSAFE and UUSAFE) in the Toolkit. See “Generating the Field Activation Objects” on page 110 for details.
2. After adding the appropriate elements in your design, define the field activation actions that can be performed on your product keys in future. See “The Field Activation Actions” on page 111 for details.
3. Ship your protected application along with the means to activate applications in the field—either the Field Activation Utility or Sentinel Client Activator.
4. Whenever your customers decides to subscribe additional features, they run the Field Activation Utility or Sentinel Client Activator to generate the locking code and sends it to you/distributor using an e-mail message, a fax, or a phone call.
5. Using that locking code, you/your distributor generate a corresponding license code to update the memory of the key. When you generate a license code, the chosen actions and commands are encrypted into the license code. See Chapter 13, “Generating Licenses And Updating Keys,” on page 157 for details.
6. You send the generated license code to your customer using an e-mail message, a fax, or a phone call.
7. Your customer applies that license code using the Field Activation Utility or Sentinel Client Activator—enabling the use of desired features.



The Field Activation Process

An Example

You can make your main application active, but make additional features inactive. This allows your customer to easily purchase and immediately activate upgrades in the field. Here is how:

Assume your product suite consists of the following applications:

- A main application—named *SceneryEditor*—that you want to run immediately at the customer's site.
- A demo of *SceneryVoice* that you want to run immediately at the customer site, but it will expire after 10 executions.

Now, you do the following:

1. Create a protection strategy and protect the main application SceneryEditor using a Short License element. By default, the application remains active, so you accept the default settings.
2. In the same design you add a Full License element that protects the SceneryVoice demo. Specify only the execution count value equal to 10 and accept the remaining default settings.
3. Generate the field activation objects—DUSAFE and UUSAFE.
4. Define the field activation actions for the design you created. You can create an action with the Activate Algorithm command for the Full License element.

You need not add any commands for the first application (protected using the Short License element) because the application is already fully licensed—your customer will be able to execute SceneryEditor as long the hardware key is attached.

5. Build the design and program the key with your design.
6. Keep the DUSAFE object with you and send the UUSAFE object, key, and the Field Activation Utility along with the protected application.
7. Your customer executes SceneryEditor normally, and evaluates the SceneryVoice. He finally decides to buy a fully-licensed copy of SceneryVoice as well.
8. He generates a locking using the Field Activation Utility and sends you an e-mail containing the locking code.
9. You generate a license code in the **Key Activator** option and send it to your customer using an e-mail.
10. Your customer reopens the Field Activation Utility and applies the license code. He can now access the fully-licensed version of the SceneryVoice. All this was done in few clicks!

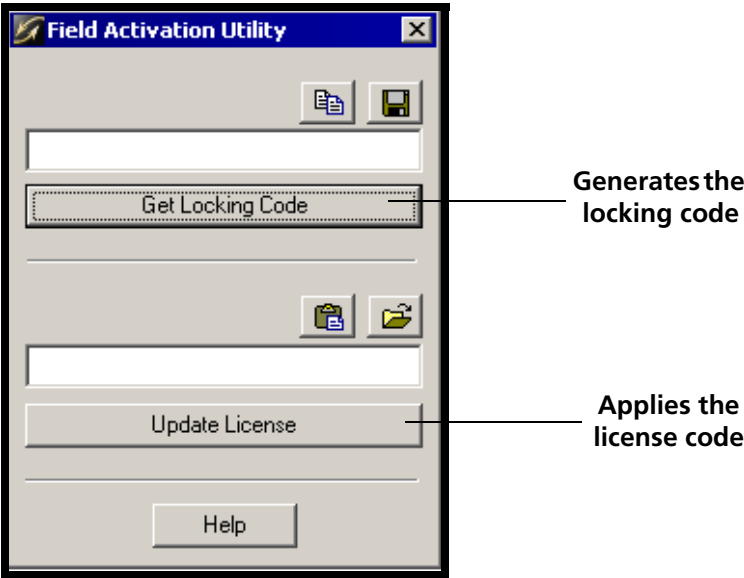
Client Activator Versus Field Activation Utility

In the table below, we compare the Sentinel Client Activator and Field Activation Utility. Even though both the methods allow you generating locking code and applying license code to update the keys in the field, both have a different *look-and-feel* and scope. Choose that suit your requirements best.

Criteria	Sentinel Client Activator	Field Activation Utility
User Interface (Windows)	Wizard-based and graphical. You can customize the user-interface, instructions, and include custom graphics, like a splash screen.	Compact and ready-to-use (requires no configuration). Look at the graphic included at the end of this topic.
Available Options	<ul style="list-style-type: none"> ❑ Available for Windows 95/98/ME/2000/XP/Server 2003. ❑ Client Activator is localization-ready. ❑ Best suited for <i>try-and-buy</i> applications that use expiration date, trial days, and execution count. ❑ Supports various activation methods, like e-mail, telephone, fax, mail, and Internet. Refer to the Sentinel Client Activator documentation for details. 	<ul style="list-style-type: none"> ❑ Best suited for activating disabled features, reactivating time-tampered applications, and converting demos into full-featured applications. It can also be used for incrementing the counter programmed into distributor keys. ❑ The locking and license code can be easily exchanged over an e-mail, telephone, mail or fax.
Integration	There is no need to install Client Activator separately on a customer's system. It can be integrated with your protected application right-before programming keys. The option is also included in the stand-alone Key Manager application, so that your distributors can configure it, if desired.	Can be shipped as an independent utility. You can even create your own field activation option using the field activation API functions.
Deployment	Since the Client Activator activation program is integrated with your protected application, your customers can use it for updating keys attached to the same system (stand-alone applications).	Can be used for updating keys attached to a stand-alone or network system, wherever the key is attached.



The Sentinel Client Activator Interface



The Field Activation Utility Interface

Using the Sentinel Client Activator

Sentinel Client Activator is an automated license installation application that is used to create a product-specific activation script for your protected application.

It is a recommended means of field activation for Sentinel UltraPro-protected applications, due to its user-friendly interface. The Client Activator also allows your customers to easily and quickly activate your product via a Web site, if you desire. Additionally, if you are going to use Sentinel Express with your Sentinel UltraPro-protected application for field activation, you must use the Client Activator.

The activation wizard helps you in building and configuring a Client Activator for a design (protecting one or many applications). The wizard collects product and publisher information that is used by the Client Activator to process a license activation request.

The Wizard allows you to choose how the product activation will be presented to your customer, and defines the methods your customer can use to activate the product (keyboard/file, telephone, fax, mail, or Internet).

To build your product-specific Client Activator, define your product and activation method(s) using the activation wizard. The wizard builds the Client Activator, which you ship with your protected application. When your customer installs the application, he or she has the option of clicking the **Try** or **Buy** button.

The **Try** button allows your customer to use the product for a specified time limit or pre-determined number of executions. The **Buy** button prompts your customer for the necessary information and completes the activation.

Refer to the Client Activator documentation for more information. See also “Deploying the Sentinel Client Activator” on page 172 for information on how to deploy it on your customers’ system,

Using the Field Activation Utility

The Field Activation Utility is a stand-alone utility you may send to your customers. It is used to first generate the locking code needed to create a

license code, and then enter the license code that updates the key with the field activation commands.

You can send it to your customers along with your protected application only if you plan to remotely update the keys and you are not using the Client Activator or any other customized method (created using the DUSAFE and UUSAFE API functions).

See “Deploying the Field Activation Utility” on page 171 for information on how to deploy it on your customers’ system,

To best understand the usage of both the methods—Sentinel Client Activator and the Field Activation utility—we recommend you to try them once.

The Client Activator is installed along with the Toolkit and Key Manager. It can be accessed from the **Make Keys** tab. See “Configuring the Sentinel Client Activator” on page 147 for details.

The Field Activation Utility can be accessed from the list of programs installed along with Toolkit.

Generating the Field Activation Objects

The field activation objects—DUSAFE and UUSAFE—form the heart of the entire field activation process. These are generated each time you change your settings in the **Generate Field Objects** dialog box in the Toolkit. To generate the field activation objects:

1. Attach and configure a key. If the key is already attached and configured, make sure it is selected in the **Key Status** pane.
2. Under the **Options** menu, click **Generate Field Objects**. The **Generate Field Objects** dialog box appears.
3. The dialog box shows configuration information about the current key you are using. In the **Secret Code** field, enter the secret code. It must be 9 to 16 characters long string, consisting of at least one numeric character. Otherwise, you may choose to auto-generate the secret code, by clicking the **Auto Generate** button. In either case,

write down the string somewhere as you will need it to update clients in future.

4. Click **Create**. An information message appears.
5. Read it carefully and click **OK**. The DUSAFE and UUSAFE objects are created in the Toolkit installation directory.



Warning about generating the field activation objects

If the objects already exist on the workstation, they will be overwritten. The objects generated each time are different even if the design details or key details are same. Hence, do remember to keep a backup of your field activation objects. If new objects are generated, and you have already distributed the old objects with your protected software, you will be unable to reprogram keys using the old objects because your DUSAFE won't match with the user's UUSAFE. In this case, be sure to create backups of your old objects so you can continue to reprogram existing keys in the field.

The Field Activation Actions

To be able to update keys in the field, you must define the field activation actions and commands that can be performed on those keys within your protection strategy.

No locking code will be generated for a key in the field, if the design you programmed into the key does not contain any remote update actions.

What Are Actions?

Actions are groups of commands. Actions allow you to group a set of field activation commands together, so you or your distributors don't have to select the commands individually when generating license codes for field activation. This is especially helpful for those people who will be generating the license codes, but are not familiar with details of the commands being used.

What Are Commands?

Commands are API function calls that describe what will be done to the key in the field. For example, the Decrement Counter command locates the counter cell on the key and decrements it by the value you specify. Actions are groups of one or more commands.

When you generate a license code, the actions and commands you select are encrypted into the license code specific to the selected key. When the license code is entered in the Client Activator or Field Activation Utility, the actions and commands are applied to the key. To be able to update keys in the field, you need to define the field activation actions that can be performed on those keys within your protection strategy. Actions and commands for activating an inactive application/feature.

Tip: License codes can be applied only once

A license code cannot be used to update one key for more than once. This is because of the one-time update feature in the key that prevents applying a license code for the next time. Furthermore, a license code cannot be applied to any other key than it is meant for. This is because the license code is tied to a particular key.

Adding a Field Activation Action

Here are the steps to add an action:

1. On the **Protection Manager** screen, select the design for which field actions are to be defined.
2. Click the **Remote Updates** tab.
3. Click **Add**. The **Remote Updates** dialog box appears.
4. Type a name for the action in the **Action Name** field. It can consist of 1 to 20 characters. The name should be concise, yet descriptive, so the people generating license codes can easily see how the key will be updated if they include this action in the license code update script.

You can create multiple actions applying different commands in the field using the steps described in this topic.

5. You may optionally include comments for the action in the **Comments** edit box.
6. Select an element from the **Elements and Commands** list. Appropriate commands for an element will automatically be shown under the **Command** list. The appropriate commands appear in the right-side panel under **Commands Options**.
7. Select a command from the **Command** drop-down list and specify appropriate values wherever required. If desired, you can allow overriding the command settings at the time of generating license codes.

The table below describes the various commands.

List of Field the Activation Actions

Command Name	Description	Overriding Allowed While Generating the License Code?	Element for Which This Command is Available
Activate algorithm	Converts an inactive algorithm to active.	No	<input type="checkbox"/> Full License <input type="checkbox"/> Short License <input type="checkbox"/> Shell
Add distributor counter	The value you want the counter on a distributor key incremented by.	Yes	<input type="checkbox"/> Full License <input type="checkbox"/> Short License <input type="checkbox"/> Shell
Bit mask AND	Value you enter is ANDed to the existing 8-bit integer value in the cell.	Yes	<input type="checkbox"/> 8-bit Integer <input type="checkbox"/> 16-bit Integer
Bit mask OR	Value you enter is ORed with the existing 8-bit integer value in the cell.	Yes	<input type="checkbox"/> 8-bit Integer <input type="checkbox"/> 16-bit Integer

List of Field the Activation Actions (Continued)

Command Name	Description	Overriding Allowed While Generating the License Code?	Element for Which This Command is Available
Decrement counter	Decrements the existing Counter element value in the key by the amount specified.	Yes	<input type="checkbox"/> Counter
Decrement counter to zero	A value does not appear when you select this command. Because it decrements the Counter element to zero. Hence, no value is necessary.	No	<input type="checkbox"/> Counter
Decrement execution counter	Decrements the existing Counter element value in the key by the amount specified.	Yes	<input type="checkbox"/> Full License <input type="checkbox"/> Shell
Decrement execution counter to zero	A value does not appear when you select this command. Because it decrements the execution count value to zero. Hence, no value is necessary.	No	<input type="checkbox"/> Full License <input type="checkbox"/> Shell
Decrement time tamper counter	Decrements an existing cheat counter value in the key.	Yes	<input type="checkbox"/> Full License <input type="checkbox"/> Shell
Decrement time tamper counter to zero	A value does not appear when you select this command. Because it decrements the cheat counter value to zero. Hence, no value is necessary.	No	<input type="checkbox"/> Full License <input type="checkbox"/> Shell
Increment counter	The value you want the selected Counter element incremented by.	Yes	Counter
Increment time tamper counter	The value you want the selected cheat counter incremented by.	Yes	<input type="checkbox"/> Full License <input type="checkbox"/> Shell

List of Field the Activation Actions (Continued)

Command Name	Description	Overriding Allowed While Generating the License Code?	Element for Which This Command is Available
Update last known good date and time	Updates the LKDT date written into the key.	Yes	<input type="checkbox"/> Full License <input type="checkbox"/> Shell
Write integer value	The value you want to write to an element. You need to enter your own value,	Yes	Integers (8-bit, 16-bit and 32-bit)
Write user limit value	Updates the user limit value already written into the key.	Yes	<input type="checkbox"/> Full License <input type="checkbox"/> Short license <input type="checkbox"/> Shell
Write boolean value	Alters the existing boolean value.	Yes	Boolean
Write execution counter	Activates the algorithm and writes the execution count by a desired amount (greater than zero).	Yes	<input type="checkbox"/> Full License <input type="checkbox"/> Shell
Write lease date	Activates the algorithm and updates the expiration date and cheat counter.	Yes	<input type="checkbox"/> Full License <input type="checkbox"/> Shell
Write string	Updates the String element value.	Yes	String

8. Click **OK** to add the action.

9. Click the **Build** button to update your design. For more details, see “Prototyping Design Using the Build Button” on page 130.

Chapter 9

Tips on Maximizing Application Security

Protecting a software application is never fail-safe. A good analogy are automobiles; they are stolen every day. However, there are many vehicles, despite their high value, that thieves avoid because they are too difficult to steal. This is generally the result of an auto manufacturer that purchased the best lock available and spent the time to integrate it properly. Otherwise, even the strongest lock can be easily defeated.

Sentinel UltraPro provides the best software protection system available today. However, like the auto manufacturer, you must take the time to properly implement the system or it will be bypassed.

The goal of any software protection strategy is to make the cost of defeating it much more than purchasing the software legitimately. Once this is in place, users get more value from buying the software than stealing it. You should provide a security strategy that fits in with the value of the software itself—the higher the value of the software, the more time you should spend protecting it.

This chapter provides you tips on maximizing your protection strategy. If you are using an API-based protection strategy, keep in mind the following guidelines to ensure your strategy is effective.

Note: Knowledge of UltraPro API is required

Creating your own customized protection scheme requires you to understand the API functions described in the Toolkit Help.

Basic Types of Attacks

Before you can plan a good protection strategy, you need to understand the basic methods used to break software. Then you protect against the most likely avenues of attack.

Attack the Hardware

This method is extremely difficult, time-consuming, and requires very expensive equipment to do. The UltraPro key provides excellent protection and it is extremely unlikely you will be attacked in this manner.

Attack the SafeNet Code

Generally, the hacker modifies the SafeNet-supplied components so that they return correct answers to the application without the key being plugged in. To combat such attacks, SafeNet components have mechanisms in-place to prevent their replacement and modification.

This method is generally avoided by all but the most experienced attacker because of its difficulty. However, you can also protect against this method with several of the tips described in this chapter. This method is usually referred to as a *record and playback* attack.

Attack the Application

Since most software developers do not write security code everyday, this is often the easiest target. A poorly protected application may only require a few quick changes to operate without the key attached. Fortunately, by using many of the tips in this chapter this can also be made extremely difficult.

Geared to Provide Unique Solutions

At SafeNet we are often asked why we do not provide specific code examples to describe the protection techniques. The reason is that every protection strategy needs to be unique. This prevents attackers from building generic mechanisms to break applications. Instead, each protection strategy must be recognized, analyzed, and overcome on an individual basis. It also means that a poor strategy will not weaken a good strategy. Each must be tackled on its own.

Sentinel UltraPro is designed from the ground up to operate uniquely for each customer.

- Each customer has their own set of passwords so only they have the capability to program the key.
- Each unique Part Number contains a unique seed. This makes the key respond differently for each developer using this system.
- Each design programs a unique seed into the license (or algorithm). Without knowing that seed, even SafeNet cannot predict the responses obtained by querying a key.
- Each key can be programmed with unique activation passwords. This makes it impossible for the code to unlock a license on one key to be used on another key.

Tips and Tricks

We have provided many tips and tricks you can implement in your application. Each tip protects against specific type of attack—and multiple methods will complement each other.

Use Shell to Encrypt Your Executable

Combine your API elements-based custom protection with Shell and add an extra layer of protection. The Shell encrypts your final executable, which makes it difficult to disassemble or debug your application.

Even if the attacker manages to overcome the difficult task of removing the Shell, the application inside is still protected—due to two strong layers of protection. Here are the steps to use this trick:

1. Begin by adding a Shell element in the Toolkit to protect your executable.

See the topic “Using Shell” on page 79 for detailed instructions on using Shell protection.

2. Before you build your design using the **Build** button, specify your choices under the **Build Options** tab. Do remember to select the **Include Shell elements** check box along with other settings.
3. Click **Build**. The resulting design header file will contain constants for accessing the license used by the Shell and a query/response table for the license. Now, both the Shell and your application will access the same license on the UltraPro.
4. Add the UltraPro API function calls to your source code that check the license first and recompile your application.
5. Now, create a Shell element again—this time with the API-protected compiled application.

Use the Query API

The algorithms-based query/response protection is the primary method of securing an application. Simply storing data in the key memory is easy to mimic even if it is encrypted. Once you know what the memory should contain, you can modify the application to return those values instead of actually reading them from the hardware key.

Using the UltraPro Query API on an algorithm descriptor programmed into the key allows your application to issue a nearly infinite amount of unique challenges. This mechanism becomes the backbone of your protection strategy since it is extremely difficult to duplicate the correct responses. You can implement this tip by adding a Full License or a Short License to your design

and then using the Query API to challenge the algorithm stored in the license.

However, just using the API once in your application is not enough; there are many other tips, like given below, you should endeavor to employ. See “Ready-to-Use Query API Functions” on page 49 for details on the Query API.

Create a Large Query/Response Table

If your application only knows a few challenges to issue to the key, then it can still be emulated. However, a large table will take a long time to use every possibility; thereby increasing the time taken to emulate every possible challenge.

Tip: Where is the query/response table?

The query/response table is written into the design header file when you build a design containing at least one algorithm-based element. ¹

Split Large Table

You must split the large table into several smaller tables. This places the tables at multiple locations inside your protected application and makes it difficult to find them all.

Query Frequently

If you rely on a single call at the beginning of your code, it is relatively easy for a skilled hacker to isolate the call and defeat your protection. Another potential problem with querying only once is that a user could remove the key after starting the application. The key could then be used to run another copy of the application.

1. The query/response table will be generated for a Shell element when you select the **Include Shell elements** check box under the **Build Options** tab.

Query Randomly

You must design the protection strategy to have the application pick the challenge from the query/response table randomly. This makes it difficult to anticipate what the challenge will be. Once you choose a challenge, use this one challenge repeatedly for some period of time (such as each time the program is run, or once a day). If your application uses a different challenge every time, then it will cycle through all available challenges in a faster time frame. This speeds up the time it takes to listen to every possible challenge.

Add Noise to your Query Checks

Generate random queries and then dismiss the results. This generates a large amount of unused data among the useful data. Anyone trying to record your communications with the key will need to record large amounts of data and have trouble deciphering what is meaningful.

Generate New Tables Frequently

Each time you create an update to your application, regenerate the query / response table. If an attacker has been able to record all the challenges used by your program, the update will suddenly require this work to be repeated. If you have used the tips discussed above, it will likely be time consuming so the illegitimate user is stuck using outdated software.

Use Encryption to Protect Data

You can use the response from a UltraPro Query API call as the seed for an encryption algorithm. The correct decryption seed can only be generated with the UltraPro key attached. Thus, any amount of data can only be opened if the correct UltraPro key attached.

It is recommended that you only use industry standard algorithms such as AES (Advanced Encryption Standard). AES has withstood intense scrutiny from the cryptography experts and the U.S. government. It is trusted by many organizations and has a proven track record. Home-made algorithms, on the other hand, often have hidden insecurities that make them easy to break.

The AES algorithm requires a 128-bit seed for its encryption/decryption. This translates easily into an 16-byte query to the key. The 16-byte response can be fed directly into the AES algorithm to protect the data. You can increase the complexity of the algorithm by continually feeding your seed back into the Query API for each data packet you encrypt. The resulting file will require knowledge of an entire series of query responses, not just a single 16-byte value.

You can use encryption to protect your application in many ways.

Encrypt the Query/Response Table

You should encrypt the query/response table, making it extremely difficult to find and use the table by looking at the code. If you only decrypt challenges as you use them, then the attacker never sees the table in a fully decrypted form.

Encrypt Data Elements on the Key

Rather than storing your data on the key in clear text, you can encrypt it so the data cannot be easily viewed or modified.

Encrypt Critical Data Used by the Application

All applications at some point require data to operate. Encrypt important data files or constants used by your program so it will only operate properly with the key attached.

Decentralize Your Security Checks

Decentralizing the security checks throughout the code is a good practice. This requires each place in the code to be modified in order for the application to run without the key. Restricting them to a few places can lead to easy detection and elimination subsequently. Given below are more tips:

Use In-line Functions in Place of a Centralized Function

Creating a single function call that checks the key and then making function calls throughout the code does not decentralize. Instead, only the security check function must be modified to operate without the key.

Use Many Different Security Checks

If you use the same security check in many places of the code, then you can find each check by searching for patterns.

Tip: Place the security checks in hard-to-trace operations

For example, if you scatter you check throughout a series of database operations, it can be extremely time consuming to trace the calls.

Scatter the Security Checks

Security checks typically consist of multiple steps: calling the key, evaluating the returned value, and acting on the evaluation results. Instead of putting the entire strategy at one place in your code, you should disperse the check all over to make the relation between them not obvious. A security check is harder to break if its code components are physically separated into different sections of the application instead of being located together.

Use Multiple Threads to Your Advantage

If the work of an individual security check happens over multiple threads, then tracing through the operation can be complicated. This makes debugging the code very difficult.

Use Returned Values as Variables

One effective technique to hide security checks in a high-level language is to use returned values to control application flow. With this method, a value returned by the key becomes a logical pointer or selection key to the next execution step or subroutine. This makes analysis of your code more difficult.

Another way to use a returned value is to add it to the value of a variable so the sum is the desired value of the variable. If the variable is used in other parts of the code, then that code is dependent on the call to the hardware key.

For example, suppose that at some point in your application you want a variable to contain the value 13. Assume that one of the query strings you send to the key returns the decimal number 12,345.

- Set the variable to -12,332.
- Send the query.
- Add the response to the variable.

If the correct key is attached, the variable will contain the proper value. In actual practice, this technique is most effective if the mathematics behind the correct value is more complicated than simple addition.

Checksum Your Code

You should adopt the practice of making a checksum of your critical data. You should also verify the validity of the application itself and any DLLs or shared libraries that it uses. This helps detect if the code or data has been modified and identify when your application has been tampered with.

Simple addition based checksums are easy to anticipate the final result. If you modify the application by increasing a single byte in the executable file by 10, you can make the checksum correct by lowering another byte by 10. This is easily done by finding another byte than is not important. CRC algorithms are harder to anticipate the results, but still can be done. The best algorithms to checksum your code are industry standard one-way hash algorithms such as MD5 or SHA-1. These generate a 128-bit or 160-bit hash value you can compare against.

Avoid comparing the result of a checksum to a previously calculated value. Use the checksum result to perform an action in your code. You can make the application behave erratically or cause an error if the wrong value is calculated.

Dealing With Missing Hardware Keys

If no hardware key is attached to the computer or in network when a protected application is run, an error is returned by the SFNTsntlGetLicense API function. If a connection is established, but the key is later removed, subsequent API functions will return errors. Refer to the Toolkit Help for exact status codes.

If your application detects that the Sentinel UltraPro key is not present, it is up to you to decide what action you want to take. Typically, you should not shut down your application because of a single unexpected response.

Instead, repeat your query; if the response is still wrong, then you can take action. Possible actions include:

- Display a message and wait for the user to respond. This method does not prevent users from running the application, but it makes doing so extremely annoying, especially if the application queries the hardware key frequently.
- Shut down the application after a predetermined number of failed queries. (However, only under the most extraordinary circumstances should you terminate your application without allowing the users to first save their work).
- Allow the application to appear as if it is functioning properly, while in fact it is not. (Be very careful if you use this method; less drastic actions should be considered first.)
- Display a critical error message and tell the user to contact your technical support department.

These are just some suggested actions; you can implement any combination of them to suit your needs. Remember, other events, such as network transmission errors or parallel port contention problems, can also cause your application to detect a hardware key problem. Since these are almost always innocent events, you should design your strategy to be as forgiving of them as possible, while still maintaining protection integrity.

Change your Strategy

Finally, as you perform software updates to your application in the field, devote time to change and improve your security checks. The longer the checks exist in the field, the more time there is to attack the mechanism. Eventually, even the toughest checks might be overcome. However, if you continually update your checks, then you can stay one step ahead. Consider this action part of a planned maintenance to keep your security at its peak level.

Note: Contact our Technical Support for expert advice

For more personalized assistance in integrating the security checks in your application, please contact our Technical Support using the information given on page xiv.

Chapter 10

Implementing the Application Protection

When you have finished designing your protection strategy—defined what type of application protection to use and all your elements are in place—you are ready to implement your strategy.

The first step in implementing your strategy is to prototype your design with a key using the **Build** button. During the prototyping, the Toolkit writes your strategy to the key, generates the design header file and code sketch.

The last step in implementing your strategy is to add the appropriate Ultra-Pro API functions to your source code when using API elements to protect your application.

In this chapter, we will be covering the following topics:

- “Prototyping Design Using the Build Button” on page 130.
- “Viewing Design in Key Layout” on page 133.
- “Final Steps of Implementing Protection” on page 135.

Prototyping Design Using the Build Button

When you are building a design, you are actually prototyping a master key with all of the elements you included in your protection strategy. During the build process, the following tasks are performed:

- The key is programmed with your design settings.
- A header file will be generated/updated for the design, depending on the **Header file generation** option you have selected.
- The code sketch will be generated/updated for the elements existing in your design.

Note: **Building a design is a required stage**

When you build your design, you are committing to your strategy. Before moving over to the **Key Manager** screen, always ensure that the design has been built. Otherwise, the various options of the Key Manager will remain disabled and the “*Design not built*” error will appear. Also, always repeat the build process after editing your design settings. The build process may be repeated as many time as necessary.

If you are building the design for the first-time, remember to specify the build options discussed on page 131. This is recommended only for the first-time and subsequently the same build options will be used whenever the **Build** button is clicked.

The Design Header File

The header file contains important information about your design, such as the constants and values for the various elements, the query/response table (if you included a Full License or a Short License in your design), design ID, Toolkit Cell Addresses, Developer ID, Write Password and Overwrite Passwords.

See the topic “Setting the Build Options” on page 131 to know about the header file generation options.

The Code Sketch

When you are protecting your application using the API elements, you must manually add the appropriate API function calls to your application's source code in order to implement your protection strategy. The Toolkit provides you with the code sketch that outlines the functions you need to add to your code. It is a good reference when you have little idea on which UltraPro API functions are relevant for the particular design created.

The code sketch is provided for the most-frequently used development languages: ANSI C, Visual Basic, and Pascal. It is written into an HTML file present in the Toolkit working folder.

To view the code sketch you can either click **View** under the **Build Options** tab or navigate to the Toolkit working folder after building a design.

Note that the header file and code sketch contain confidential information about your design—make sure you keep it secure!

Setting the Build Options

1. After adding the elements and actions for a design, click the **Build Options** tab on the **Protection Manager** screen.
2. Specify the **Header file generation** option for subsequent modifications in the design. This setting will decide how often the design header file will be updated. You can select any of the following options:
 - ☐ **Automatically generate when necessary (default)**
Toolkit automatically decides and generates the header file whenever necessary.
 - ☐ **Generate for every build**
Toolkit will generate the header file every time the Build button is clicked.
 - ☐ **Ask me before building**
Toolkit seeks your permission every time the header file requires a revision.

3. Now, set the parameters for the header file.

☐ **Include Shell elements**

If you have added any Shell element in your design, you will see this check box enabled. When you select it before building the design, it will generate the code sketch and query/response table and Toolkit Cell Addresses for your Shell element(s) as well.

☐ **Programming language**

Select the development language you want to view the code sketch in.

☐ **Number of queries**

Specify the number of queries you want to make. More queries provide better security. However, this may take time in header file generation.

☐ **Query size**

Select how long, in bytes, you want the query string to be. The query string must be between 1 and 56 bytes. The longer queries provide better security. However, this may take time in header file generation.

4. Click the **Build** button to proceed. A dialog box will appear displaying the status of the various activities—key programming, header file and code sketch generation.

5. As soon as the design is built, a dialog box will appear prompting you to copy the Sentinel UltraPro libraries and design header files (for example, *upromeps.h* and *upromepsdesign.h*) at a user-defined path.

6. If desired, you can click the **Copy** button to copy the design header file an existing directory.

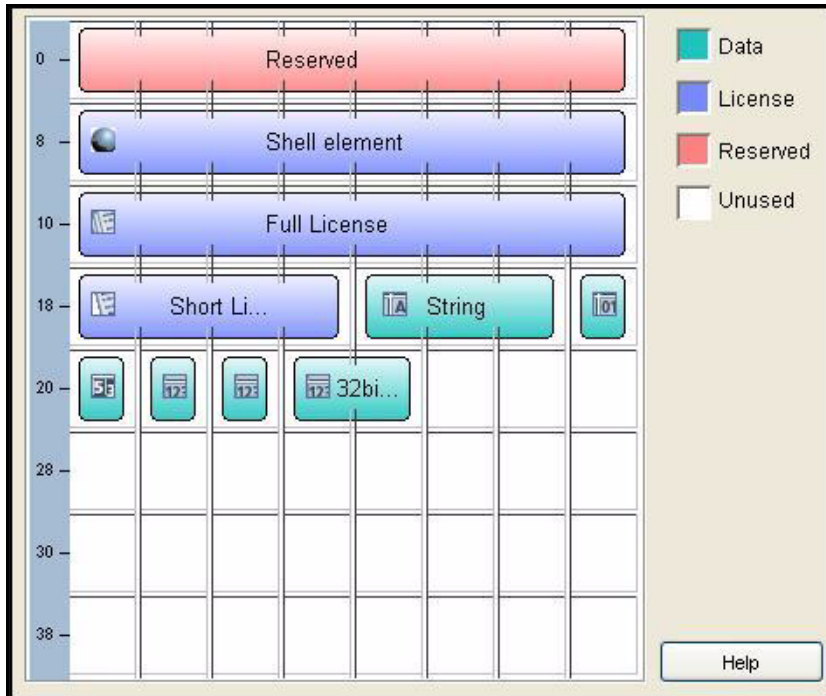
7. You can also click the **View** button to show the code sketch for this design in the (default) Web browser.

Tip: Evaluating API in the API Explorer

Before adding the UltraPro API functions in your source code, you should evaluate them in the API explorer screen. Refer to the Toolkit Help for details.

Viewing Design in Key Layout

You can use the **Key Layout** tab to view the virtual layout of your design in the key memory. It is updated automatically as you add the elements in your design and you need not build your design to produce the key layout.



The Key Layout View

Viewing Element Properties

To view the element properties in the Key Layout, double-click on the Data or License area. The **Element Properties** dialog box appears with the following information:

- Element name
- Element type
- Element value
- Toolkit Cell Addresses of the element and its controls (if any).

You can edit the existent element settings by clicking the **Edit** button.

Rearranging Elements

You can cut and paste certain elements in the Key Layout. Here are the points to be noted:

- You cannot move the reserved cells and the Shell/Quick Shell elements.
- A Full License element occupies at least a row, consisting of 8 consecutive cells of 16-bits each. It can only be pasted to an extreme left corner of a row—with the cell address equivalent to 0 MOD 8.
- A Short License element occupies at least 4 consecutive cells of 16-bits each. It can only be pasted to start from the left corner of a row, or the middle of the row—with the cell address equivalent to 0 MOD 4.
- Grouped elements (contained within a single cell), such as an 8-bit integer and Boolean, can only be moved as a group/block.

To move an element:

1. Right-click on the element you want to move. The shortcut menu appears.
2. Select **Cut**.

3. Right-click on the location in the Key Layout where you want to paste the element.
4. Select **Paste** from the short-cut menu that appears.

Final Steps of Implementing Protection

After having added the UltraPro API functions into your application's source code, you can:

- Include the design header files generated at the build-time.
- Include the UltraPro library generated at the build-time.
- Compile your application.
- If desired, you can apply the Shell protection on your executable and data files. Refer to “Using Shell” on page 79 for exact steps on using Shell.

Note: Testing your protection

At this point, when you have created your protection strategy, programmed a prototype key, and implemented the strategy (added the appropriate API and code or applied Shell), you can proceed with testing your protected application. We recommend testing your application to verify that it executes correctly with the appropriate hardware key both attached and missing.

Part 3

Programming Keys and Generating Licenses

- ❑ Programming Product Keys
- ❑ Programming Distributor Keys
- ❑ Generating Licenses for Remote Updates

Chapter 11

Programming Product Keys

This chapter describes how to program the product keys. We will be covering up the following topics:

- “Setting Up to Program Product Keys” on page 139.
- “Programming a Product Key” on page 142.

Setting Up to Program Product Keys

Once you have completed your protection strategy, including building a design, you are ready to start programming product keys to include in the final package with your protected application.

Typically, as a developer, you will not be responsible for programming the keys that will be shipped to your customers. This task is usually handled by your company’s manufacturing department as part of the assembly-line process for creating your product.

To avoid giving your manufacturing department access to your passwords and strategy—which would also give them the ability to change field activation commands or other elements in your protection strategy—we recommend providing them with the Key Manager application included with Sentinel UltraPro. For more information on how to use this application, refer to its Help. If you had installed Sentinel UltraPro using the

Complete option, you can access it from the list of programs installed on a Windows system.

You may also decide to allow your distributors to program product keys and generate license codes. In this case, you will need to provide those distributors with a pre-programmed distributor key to meter the number of keys they program/activate. You may be responsible for programming distributor keys, or this task may also be handled by your company's manufacturing department. Refer to Chapter 12, "Programming Distributor Keys," on page 149 for details on programming distributor keys.

Before you begin programming product keys, make sure you have an adequate stock of keys with your assigned Part Number. The keys you will be programming as product keys must have the same Part Number as the key used while you designed your protection strategy. If you need additional keys, please contact your Sentinel sales representative. Also, be ready with the following information before ordering keys:

- What key memory size do you want?
- What key format you want: parallel or USB?
- What is the hard-limit of the key you want?

Number of Keys You Can Program

Though you can attach as many keys as possible over the USB and parallel ports on your system. However, we recommend you to follow the limits mentioned below based on the electric current availability:

- **Number of USB keys supported**

At a time, you can program up to 32 USB keys attached on your port/hub.

- **Number of parallel port keys supported**

You can program up to five parallel port keys attached/cascaded on over a single parallel port. You can have up to three parallel ports on a system.

Connecting the Keys

As you program multiple keys, you will be taking keys on and off your workstation frequently. As a result, you will want to protect connectors from being damaged, while at the same time making it convenient to connect and disconnect keys.

To solve both issues, we recommend attaching a shielded cable with appropriate connectors for the key you are programming that reaches from the port to a work surface, preventing you from having to reach or bend over to connect or disconnect keys.

Do not remove a key while it is being programmed, as a write failure will occur. A key removed during programming can be reprogrammed.

Using Cables to Connect Keys

Due to the large variety of cables currently on the market, we do not recommend a specific brand or type of cable for use with the UltraPro key, nor do we guarantee that all cables will be compatible with the key. However, we do recommend the following:

- Cables should not be longer than 6 feet in length.
- Cables should be shielded.
- Do not use ribbon cables.
- Cables must be straight-through; that is, they must have all pin signals wired through to the connectors on either end of the cable.

Please be aware that cable connectors may only be used for a specific number of times, perhaps for as few as 100 connections. Contact the manufacturer of your cable/connector for specific information on how many insertions the cable/connector is rated for. Based on that specification, change the cable/connector on a regular basis, as needed.

Additionally, since the insertion life of the connector on the computer you are using for programming is also limited, you may want to consider using the bulk programming services offered by your Sentinel key vendor, as a solution for reliable, high-volume key programming. For doing so, you need

to provide them an *.sms* file for a design. Using the *.sms* file, a Sentinel key vendor can program keys without having access to your design details. Refer to the next topic that discusses how to create an *.sms* file for your key vendor.

Programming a Product Key

In order to program keys for customers, you require a design prototyped using the **Build** button. If this step is not completed, then “*Design not built in Protection Manager*” error will be shown in the Key Manager screen and the options will remain unavailable. To program a product key follow the steps given below:

1. On the **Key Manager** screen, under the **Make Keys** tab, select the project from the drop-down menu.
2. Select the design from the list.
3. Follow the information given in the table below to select the field activation options provided:

Specifying the Field Activation Options for Programming Keys

Check Box Label	Default	When To Select	When Not to Select
Make all the algorithms in this design inactive	Not selected.	As part of your protection strategy, you choose to make your application either active or inactive. An active application is one that is ready to run when shipped to your customer. It will always remain active, as long as the hardware key is attached. An inactive application will not run until it is activated.	Do not select this check box if: <ul style="list-style-type: none"> <input type="checkbox"/> You want to keep the algorithm active on start. As a result the protected application will run on start as long as the correct UltraPro key is attached. <input type="checkbox"/> You want to make a particular (and not all) algorithm inactive. Selecting this check box will make all the algorithms, inactive in one attempt. If desired, you can selectively de-activate algorithms in the Customize Elements dialog box.
Use unique activation passwords for each key	Not selected. (Enables only if the first check box is selected)	Select this check box if you want to use unique passwords for activating algorithms in the field. Otherwise, static passwords will be used for all the keys/customers.	<ul style="list-style-type: none"> <input type="checkbox"/> Do not select this check box if you want to use static (same passwords) for activating algorithms in the field. However, unique passwords provide more security.
Do not update keys in the field	Not selected	Select this check box if you NEVER intend to update keys in the field.	Do not select this check box if: <ul style="list-style-type: none"> <input type="checkbox"/> You plan to update keys in the field for performing some kind of field activation commands. <input type="checkbox"/> You have selected the first check box.

Note: Demo applications are a special case

Demo or metered applications are a special case, in that they are shipped as active, but usually become inactive after a specific number of executions. Select the **Make all the algorithms in this design inactive** check box if you want your customers to obtain a license code in order to execute the application. This is because that selecting this check box will make all the algorithms in your design inactive. The application will not start if the algorithms are inactive. To activate the algorithms, your customers need to send you the locking code using which you will generate a license code.

4. After specifying the field activation choices, click the **Customize Elements** button.

The dialog box will display the element details of the design you are programming keys with. In case, you had selected the **Override** check box while adding an element, you can edit the value now, for this batch of keys being programmed. Please note that the edited value cannot exceed the maximum limit stipulated at the time of creating an element. Also, the edited settings will be applied only to this batch of keys being programmed, without changing your overall strategy.

Generally the following choices will be shown, depending on the controls you selected for overriding while adding elements:

- ☐ **Active:** Using this check box you can make the algorithm active or inactive on start. If the **Active** check box is selected, the application will be shipped as active for this key only. If the check box is cleared, the application will be inactive upon shipment, requiring the user to explicitly activate the application.
- ☐ **Expiration Date:** The fixed expiration date (month, day and year) after which the application will not run.
- ☐ **Trial Days:** The number of days your protected application will run for. Please note that under no situation the application can run beyond the fixed expiration date.

- ❑ Cheat Counter: A counter limit that is decremented by one each time a minor time tampering is attempted.
 - ❑ Execution Count: The maximum number of times the application will run.
 - ❑ User Limit: The number of users who can use this application. Please note that the actual number of users are also affected by the hard limit of the key and the license sharing settings.
 - ❑ String Value: A string made of up of ASCII characters. Under no situation, this can exceed 50 characters.
 - ❑ Boolean: A true/false value.
 - ❑ Counter: A count-down value that decrements by one each time the associated feature is executed.
 - ❑ Integer: Refers to the 8-bit, 16-bit and 32-bit long Integer values.
5. Click **OK** if you do not want to create a *.sms* file. And, move to step 10.

You can write a *.sms* file for your key vendors using which your key vendors can program keys in bulk for you. To create a *.sms* file, follow the steps given below:

Note: The SMS file option is not shown

It is possible that currently the Create SMS file option is not shown to you. To have that option, click **Settings** under the **Options** menu. In the **Settings** dialog box, select the **Allow exporting .SMS file to key vendor for mass manufacturing** check box. Now, verify in the **Customize Elements** dialog box if the option of creating an SMS file is provided.

6. Click the **Create SMS File** button.
7. In the resultant dialog box, provide a pathname for the output file.
8. Click **OK** in the to create the file.
9. Click **OK** in the **Customize Elements** dialog box and move over to the next step.

10. Attach the key (or keys) to be programmed to the parallel or USB port (or both) ports/hubs.
11. Select any of the two radio buttons to specify whether you wish to program a single key or multiple keys.
12. Also, decide whether you wish to fill up the empty cells in the key with random data or not. By default the **Program unused cells with random data** check box is selected. We recommend you to keep that selected because it provides more security.
13. Click the **Make Keys** button to program the keys.

After each key is programmed, the key programming statistics are updated:

- ☐ **Pass**
The number of keys that have been successfully programmed.
- ☐ **Fail**
The number of keys that have not been successfully programmed.
- ☐ **Total**
The total number of keys you have programmed during this session, whether they passed or failed.

Verifying the Key Was Programmed Correctly

We recommend periodically checking your product keys to be sure they are being programmed correctly. To do so, attach a randomly selected key to the workstation running the Toolkit, then open the **API Explorer** screen. Now, evaluate the following API functions to view the memory view of the key.

- SFNTsntlInitialize
- SFNTsntlGetLicense

The values shown should match for all the keys. Verify the element values at the relevant cell addresses. Please note that if you had selected the **Program**

unused cells with random data check box, randomized data might also be shown.

For detailed steps on evaluating the UltraPro API refer to the Toolkit Help.

Tip: Programming failure is hardware or software related?

To determine if a programming failure is due to a software error or a hardware error, try programming another key with the same design. If the programming is successful, the error was hardware-related. If you try programming many keys, and all of them fail programming, the error is software-related.

Configuring the Sentinel Client Activator

To allow field activation using the Sentinel Client Activator, you need to configure the activation wizard. Click the **Configure Client Activator** button to launch the activation wizard. Refer to the following documentation for more details:

- To get started with Sentinel Client Activator, see its *Getting Started Guide*.
- For complete steps on using the Sentinel Client Activator, refer to the Sentinel Client Activator Developer's Guide (Online).

Note: Disable the Configure Client Activator button

By default, Sentinel Client Activator is allowed to be configured for every design. However, you can turn it off by clearing the **Associate Client Activator** check box in the **Design Properties** dialog box.

Where to Go Next?

Please refer to the following chapter/topic:

- Chapter 12, "Programming Distributor Keys," on page 149.
- "Redistributables for Customers And Distributors" on page 163.

Chapter 12

Programming Distributor Keys

This chapter describes how to program the distributor keys using the **Distributor Key** option of the **Key Manager** screen in the UltraPro Toolkit. The following topics are covered:

- “Adding Value Through Distributors” on page 149.
- “Using the Distributor Keys Option” on page 152.
- “Updating Distributor Keys in the Field” on page 154.

Adding Value Through Distributors

If desired, you can involve your sales distributors in Your distributors’ responsibilities are dependent on how much work you want to off load to them. Sentinel UltraPro allows for the following distribution models:

- The distributor can program product keys for a design.
- The distributor can generate license codes to update keys remotely.
- The distributor can configure the Sentinel Client Activator as a means to update keys in the field.

In addition to this:

- You can allow your distributors to modify the element settings—such as the expiration date, execution count, trial days, user limit, cheat counter, and string, integer, counter and Boolean values—at the time of programming product keys.
- You can allow your distributors to modify the license code settings—the values you assigned to field activation commands—at the time of generating license codes.

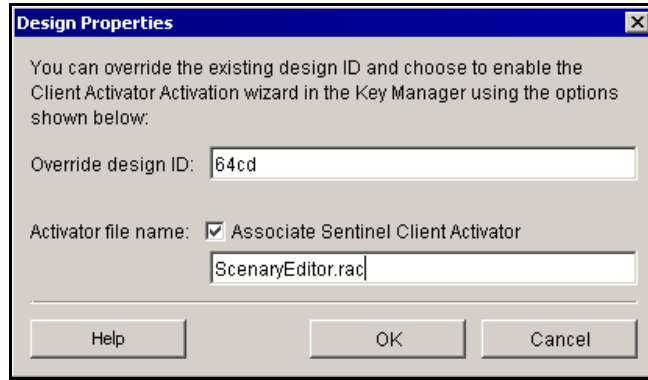
Distributors perform these tasks in much the same way that you would do. They use the Key Manager application, together with the *.dst* file and distributor key for a design.

You need to provide them with the following items:

- Prepare a design that protects the application your distributor will support in the field. If your distributor will be supporting multiple products, protected using different designs, you need to create a *.dst* file and a program distributor key for each design separately.

In case you are allowing field activation for that application, then also remember to add field activation actions in the design.

- Program a distributor key with that design. While doing so, you can also specify the metering count for certain elements in the design.
- Create a distributor file (*.dst*) for that design. Please note that the distributor key and the *.dst* file are a matched pair. Hence, for every distributor you need to program a key and generate the *.dst* file under the **Distributor Keys** tab. Your distributor can import these files in the Key Manager application by using the **Import DST File** option. Refer to the Key Manager Help for exact steps.
- To allow Sentinel Client Activator configuration for that application, you need to ensure that the Associate **Sentinel Client Activator** check box is selected in the **Design Properties** dialog box. Otherwise, the option to configure Sentinel Client Activator wizard will remain disabled under the **Make Keys** tab.



The Design Properties Dialog Box

Tip: **How to obtain the Design Properties dialog box**

In the **Protection Manager** screen, select a design from the designs list. Now, click **Design Properties** under the **File** menu to obtain the dialog.

- In addition, you need to ship all the items required by your distributors—such as, the Key Manager application’s installer, *.dst* file, the distributor key, UltraPro keys to be programmed, DUSAFE object, and various other items meant for your customers. Refer to Chapter 14, “Deploying Protected Applications,” on page 163 for a list of items that need to be shipped to your distributors and customers.

Using the Distributor Keys Option

Given below are the steps for creating a *.dst* file and programming a distributor key:

1. In the **Protection Manager** screen, select the design to program the key with.
2. Now, click the **Build** button. Be sure that a configured key is attached to your system before you click the **Build** button. For details on how to build a design, refer to “Prototyping Design Using the Build Button” on page 130.

If the design is not built, then its details will not be shown in the **Key Manager** screen and the “Design not built” error will be shown.

3. After having successfully built a design, move over to the **Key Manager** screen.
4. Click the **Distributor Keys** tab.
5. Select the project from the list (required if multiple projects are existing).
6. Select the design from the list (required if multiple designs are existing). The elements will appear with their metering options in the right-side panel.
7. Edit the element's metering settings. You can specify **Unlimited** metering options if you do not want track the activities your distributor. In other words, the distributor will be able to program unlimited product keys and generate license codes for your product. Now, move to step 10.
8. If you want to limit the metering options, select **Enter a value** from the **Metering Options** drop-down menu. You can specify metering count for the algorithm-based elements only, i.e., Full License, Short

License and Shell (which is programmed as a Full License in the key). For data-based elements, no metering options need to be provided.

The distributor key counter will be decremented as per the following conditions:

- ❑ The counter will be decremented by 1 if, an algorithm-based element is used to program keys or, the action being used to generate license code contains any of these commands: Activate algorithm, Write lease date or Write user limit value.
- ❑ The counter will be decremented by an amount equivalent to the user limit if, a user limit is also specified in any of these elements (Full License, Short License, and Shell). For example, your distributor programs a product key for a design that contains a Short License element with 5 user limit, then the Short License metering count will be reduce by 5 if a single product keys is programmed. If 10 product keys are programmed, the metering count will be reduced by 50 (10 x 5).

Note: Error will appear if insufficient count is left in the distributor key

An error will be shown while programming product keys or generating license codes if insufficient metering count is left in the distributor key. Your distributors can use the field activation process to update the distributor counter in the key.

9. Enter a value between 1 and 65,534.
10. If there are multiple algorithm-based elements, specify appropriate metering options for each element because, the decrement will be for each element.
11. Now attach a key to the appropriate port on your system. Since we recommend attaching only one key at the time of programming a distributor key, you can either remove the key you attached in step 2, or program that as a distributor key.

Note: Keys must have same Part Number

Please make sure that the key to be programmed as distributor key must have same Part Number as the key you used for building the design. Since a developer is usually provided with keys of same Part Number, a situation like this rarely arises.

12. Click the **Make Distributor Keys** button. The **Make Distributor Key** dialog box appears.
13. Click **Browse** or type a pathname to write the *.dst* file.
14. Click **Program Key**. Your key will be programmed in few seconds. And, the *.dst* file will be written at the path specified in step 13.

Updating Distributor Keys in the Field

When the activation counter on a distributor key for an element reaches zero, the distributor will no longer be able to program keys or perform updates to keys in the field. You can increment the activation counter on a distributor key (and charge for doing so) in the same way that you update product keys in the field.

For doing so, you must add the **Add distributor counter** command in the action you created for that element. Also, make sure that you generate a different pair of field activation objects (DUSAFE and UUSAFE) just for your distributors so that they are not able to update the counter you have programmed in their key without your approval.

Refer to Chapter 8, “Designing the Field Activation Strategy,” on page 103 to learn how field activation actions are added and objects are generated.

To update a distributor key in the field:

1. Ask your distributor to run the Field Activation Utility, while his distributor key is connected to his system, to generate a locking code.
2. Tell your distributor to send the locking code to you via telephone, fax or e-mail.

3. Enter the locking code in the **Key Activator** tab. The key information is extracted from the locking code.
4. Select the action that consists of the command for updating the distributor key counter for that element.
5. Click **Generate License Code**. Toolkit generates a license code—specific to the distributor’s key—based on the locking code you entered and the action you selected.
6. Send the license code back to the distributor.
7. Distributor enters the license code in the Field Activation Utility. The key is updated and the distributor can again program keys and activate application by generating the license codes.

Where to Go Next?

Please refer to the following topics in the subsequent chapters:

- “Redistributables for Customers And Distributors” on page 163.
- “Deploying the Key Manager Application” on page 174.

Chapter 13

Generating Licenses And Updating Keys

Field activation requires both you and your customer to exchange information about the key. Your customer is responsible for generating and sending the locking code to you. You are responsible for generating and sending the license code to the customer. This chapter deals with the developer's or distributor's role in field activation. The topics covered are:

- “Receiving the Locking Code from Your Customer” on page 157
- “Generating a License Code” on page 158
- “Sending the License Code to Your Customer” on page 159

Receiving the Locking Code from Your Customer

Before you can generate a license code, you must receive a locking code from your customer for the key protecting the application they want to update. If the customer has multiple copies of your application, each of which use a different key, your customer must send you a unique locking code for each key.

There are several ways that a customer can provide you with the locking code; use the method that works best for both you and your customer:

Also, your customer can save the code to a file. This file has an default name of *LockingCode.loc*, and can be loaded directly into the Key Activator option.

- Telephone
- E-mail
- Fax
- Internet

Generating a License Code

You can use the **Key Activator** tab for generating the license codes to update product keys and distributor keys in the field. Here are the steps:

1. Enter the locking code provided by your customer. To enter the locking code, you can use the following options:
 - ❑ If you copied the locking code to the clipboard, click the paste button to paste the code in the field.
 - ❑ If the locking code was saved to a file (*.loc*) click the open button to locate and open the file. The code is entered in the field automatically.

As soon as a the locking code is pasted, the design details become visible.

2. Now, select the appropriate action for generating the license code. Actions determine what elements of the application will be activated or upgraded.
3. Now, click the **Generate License Code** button. The **Customize Actions** dialog box appears.
4. You can edit the values of the commands for which you had selected the **Override** option while adding/editing the action(s). Please note that the modified commands settings will be valid for only this license

code. These modifications will not affect the actual values you chose while defining these commands.

Otherwise, the values you specified at the time of adding field activation actions will be applied.

5. Click **OK**. You are returned back to the Key Activator tab and the license code is generated.
6. To copy the license code, you can use the following options:
 - ❑ To copy the license code to the clipboard, click the paste button.
 - ❑ To save the license code to a file, click the save button and then select a file name and location for the license code file.
 - ❑ To directly e-mail the license code, click the e-mail button. This will paste the license code in a text format using the default e-mail client on your system.

Tip: License code can be used only once to update keys

The license codes generated can only be used once to update keys. This way your customers and distributors cannot get additional features or increment counters on their own.

Sending the License Code to Your Customer

Once the license code has been generated, you need to send it to your customer. You have several options for how to send it; use the method that works best for both you and your customer:

- Telephone
- Fax
- E-mail

When your customer receives the license code, they must enter it in the Client Activator or the Field Activation Utility to update or activate their key and receive their updates.

Part 4

Deployment

- ❑ What to Ship to Your Customers and Distributors?
- ❑ Tips on Deploying the Sentinel UltraPro Redistributables

Chapter 14

Deploying Protected Applications

This chapter guides you on the items that you must ship to your:

- Customers who will be using the protected applications.
- Distributors who will be programming keys and activating applications to support your protected applications.

You should be familiar with the protection strategy, so that you can choose the appropriate items for deployment.

Redistributables for Customers And Distributors

Apart from sending a hardware key programmed with your design, you should provide a few other components to your customers (see the check-list below).

If your company uses distributors to sell your products, then all the customer items must also be passed to your distributors.

Check List of the Items to Be Redistributed

No#	Component	Customer	Distributor
1.	Sentinel driver	✓	✓
2.	Sentinel Protection Server	✓	
3.	Configuration file	✓	
4.	Field Activation Utility and its Help – Your distributors may need it for updating the metering count in their keys.	✓	✓
5.	UUSAFE object for customers	✓	
6.	UUSAFE object for distributors – Your distributors may need it for updating the metering count in their keys.		✓
7.	DUSAFE object for generating license codes		✓
8.	Sentinel Client Activator files (.rac and ainst.exe) – Required only when you are using Client Activator for field activation.	✓	
9.	Data Protection Driver – Required only when you have encrypted data files in Shell for Windows 98/ME systems.		
10.	Sentinel UltraPro System Administrator Help (HTML)	✓	
11.	Key Manager's self-extracting installer		✓
12.	Design-specific distributor key		✓
13.	Design specific.dst file		✓
14.	hhupd.exe and hhactivex.dll – Needed for viewing the.chm files (on Windows), like Field Activation Utility Help and Sentinel Protection Installer Help.	✓	
15.	Msvcrt.dll – (Needed only for applications using Shell)	✓	



Items not to be sent to your customers and distributors

Never ship the project (.sup) and design (.sud) files to your customers/distributors.

Deploying the Sentinel Driver

When to Deploy?

Sentinel driver is the device driver for using the hardware keys. It must be redistributed with all kinds of UltraPro-protected applications, regardless of the strategy chosen.

Where to Deploy?

The Sentinel driver must be deployed on the system where the hardware key is attached.

- If the application is a stand-alone application, install the Sentinel driver on the system where the application is installed.
- If the application is a network application, install the Sentinel driver on the system in the network where the hardware key is attached.

How to Deploy?

You can use the Sentinel Protection Installer to deploy Sentinel driver and/or Sentinel Protection Server.

For Windows

The `<installdir>\UltraPro\<UltraPro version>\Sentinel Protection Installer` path contains the relevant files, including the merge modules and an MSI that you can use in your Windows Installer-based package. It also contains a Help file that guides you about the various methods to deploy Sentinel System Driver.

Tip: Download latest releases

Keep watching <http://www.safenet-inc.com/support/index.asp> for the latest releases of Sentinel Protection Installer. You can provide the same Web address to your customers/distributors for downloading directly.

Deploying the Sentinel Protection Server

When to Deploy?

It is the license manager for your protected applications. As a thumb-rule, it is necessary for using network applications—when clients concurrently access the hardware key(s) attached to a networked system.

The access mode you have chosen for your protected application can also help you in deciding whether you need to ship the protection server or not.

Access Mode	Requires Sentinel Protection Server or Not
SP_STANDALONE_MODE	NO
SP_DRIVER_MODE	NO
SP_LOCAL_MODE	YES
SP_BROADCAST_MODE	YES
SP_ALL_MODE	YES
SP_SERVER_MODE	YES
Directed call to a system	YES

Refer to the topic “Which Access Mode To Use?” on page 60 to understand the various access modes.

Where to Deploy?

The Sentinel Protection Server must be installed on the system where the hardware key is attached.

- If the application is a stand-alone application, install the Sentinel Protection Server on the system where the application is installed. The Sentinel driver and key must also exist on the same system.
- If the application is a network application, install the Sentinel Protection Server on a system in the network where the hardware key is attached. The Sentinel driver must also exist on the same system. If

necessary, multiple Sentinel Protection Servers can be installed in a network, as long a hardware key is there for each system.

How to Deploy?

You can use the Sentinel Protection Installer to deploy Sentinel driver and/or Sentinel Protection Server.

For Windows

The `<installdir>\UltraPro\<UltraPro version>\Sentinel Protection Installer` path contains the relevant files, including the merge modules and an MSI that you can use in your Windows Installer-based package. It also contains a Help file that guides you about the various methods to deploy Sentinel Protection Server.

On Windows NT/2000/XP/2003, the Sentinel Protection Server is installed as a system service that will start automatically whenever you boot up.

Note: Deploying Sentinel License Monitor

No additional steps are needed to deploy Sentinel License Monitor, unless you are customizing the Sentinel License Monitor `.class` files. Refer to the readme file available at: `<installdir>\Sentinel License Monitor\Help` for details.

Deploying on Novell NetWare Platforms

When to Deploy?

You need to deploy `SPNNWSRV.NLM`—the Sentinel Protection Server for Novell—when the hardware key is attached to a Novell file server. `SPNNWSRV.NLM` has a built-in Sentinel driver that allows you to directly communicate with Sentinel parallel port keys. However, it does not support the Sentinel USB keys.

Where to Deploy?

SPNNWSRV.NLM must be deployed on the system where the hardware key is attached. In order to access the key, clients can set the contact server directly (the protocol must be set to SAP, TCP, or IPX), or make a network broadcast using the *SP_BROADCAST_MODE* access mode.

Please also note the following requirements:

- Make sure that the *WS2_32.nlm* and *TCPIP.NLM* modules are loaded on the Novell file server.
- The Windows client must have IPX installed in addition to the Novell client specific to the Windows platform.
- Windows NT clients are recommended to set the IPX frametype property to Ethernet 802.2 for successful communication with the Novell server.
- Windows 9x clients require "Client for Novell NetWare" to communicate with the Novell file server.
- The *nspdns.nlm* module must be loaded before running the Sentinel Protection Server.
- To be able to resolve the IP address/host name, make sure that the following requirements are met:
 - For static IP addressing scheme: Mappings for the host names and host aliases to IP addresses exist in *SYS:ETC:HOSTS*. For example:
192.168.100.229 NOVELL_SERVER
 - For dynamic (DHCP) IP addressing scheme: You must modify *SYS:ETC:resolv.cfg* for the DHCP server entries as follows,

```
domain          domain_name  
  
nameServer      (IP Address of the DHCP Server)
```

How to Deploy?

Given below is information for loading and unloading protection server on Novell:

Loading Sentinel Protection Server on Novell

You can load the Sentinel Protection Server using either of the two options given below:

Note: The protection server file must be in the system (sys:) directory

The protection server file must be in the system (sys:) directory, otherwise the HTTP request for the Sentinel License Monitor will fail.

Using Command Prompt

1. Copy *SPNNWSRV.NLM* and the complete *\root* directory to the *sys:* directory of the file server.
2. Attach the UltraPro key to the parallel port.
3. Load the protection server using the `load sys:\spnnwsrv.nlm` command.

Using AUTOEXEC.NCF

When set through *AUTOEXEC.NCF*, the protection server runs automatically every time the file server boots up. To do so:

1. Copy *SPNNWSRV.NLM* and the *\root* directory to the *sys:* directory of the file server.
2. Type `load spnnwsrv.nlm` into the *AUTOEXEC.NCF* file.
3. Attach the key.
4. Reboot the system.

Unloading Sentinel Protection Server on Novell

To unload the Sentinel Protection Server, type `unload spnnwsrv` at the command prompt.



WARNING!

Make sure that `root` directory and its files—copied along with the Sentinel Protection Server—are neither moved or deleted. These must co-exist with the protection server executable (*spnsrvnt.exe* or *SPNNWSRV.NLM*) on the system.

Deploying the Configuration File

When to Deploy

If you want to allow your customers to set any/all of the following settings, you must ship the configuration file:

- Network protocol
- Access mode
- Heartbeat time interval

Note: Priority of the configuration file

Please note that an access mode, a protocol and heartbeat set using the `SFNTsntlSetContactServer`, `SFNTsntlSetProtocol` and `SFNTsntlSetHeartBeat` function respectively will always override the settings provided in the configuration file. Further, an access mode set using the `NSP_HOST` environment variable will be applied only if the `SFNTsntlSetContactServer` function is not called and the configuration file is not present at the appropriate location.

Where to Deploy

This file must exist in the same directory where your application is installed.

How to Deploy

This file is available in the UltraPro installation. You must include it in your application's installation program, and instruct your customers to set the values accordingly.

Deploying the Field Activation Utility

When to Deploy

The Field Activation Utility is an alternative to the Client Activator for remotely activating the applications/updating keys. It is used to first generate the locking code needed to create a license code, and then enter the license code that performs the field activation commands.

This utility needs to be sent to your customers along with your protected application only if you are planning to update keys remotely and not using the Client Activator or, any other customized option to update keys.

Where to Deploy

In order to update keys, the Field Activation Utility must be installed on the same system where the UltraPro key has been connected. This is because keys cannot be updated over network.

- If your application is a stand-alone application, the Field Activation Utility should be installed on each client system.
- If your application is a network application, the Field Activation Utility should be installed only on the system where the UltraPro key is located; it does not need to be installed on the client systems.

Be sure to modify your installation programs appropriately.

How to Deploy

For Windows Platforms

Install the following items as part of your application's setup routine:

- Field Activation executable file (*FieldActUtil.exe* – available at: `<installdir>\Toolkit`)
- *uusafe32.dll*
- Help file (*FieldActUtil.chm* – available at: `<installdir>\Toolkit\Language packs\en_US`)

Note: For using .chm files

If you are shipping a .chm file, you may also need to ship *hhupd.exe* and *hhactivx.dll*. Generally, these files reside in the \System directory of a Windows system. Otherwise, these can be downloaded from: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/htmlhelp/html/hwmmicrosofthtmlhelpdownloads.asp>.

Deploying the Sentinel Client Activator

When to Deploy?

The Sentinel Client Activator needs to be deployed on a customer's system only if you are planning to update keys remotely and not using the Field Activation utility or, any other customized option to update keys.

Note: Deploying Client Activator for distributors

Please note that the Client Activator program can be integrated with your design in the Key Manager screen (available in full Toolkit or as a stand-alone), and there is no need to distribute it separately for your distributors.

Where to Deploy

In order to update keys, the Client Activator must be installed on the same system where the UltraPro key is attached. Hence,

- If your application is a stand-alone application, the Client Activator should be installed on each client system.
- If your application is a network application, the Client Activator should be installed only on the system where the UltraPro key is located; it does not need to be installed on the client systems. Be sure to modify your install program appropriately.

How to Deploy

Include the *.rac* and *ainst.exe* files in the application's install program. Modify the install program to run *ainst.exe* during installation.

Deploying the Data Protection Driver

When to Deploy

If you have encrypted data files at run-time in Shell, then you need to deploy the Sentinel Data Protection Driver on your customer's system. Please note that it is required only for Windows 9x systems.

Where to Deploy?

It must be deployed in the */System* folder on the system where the protected application is installed.

How to Deploy

The *\Data Protection Driver* directory in the Sentinel UltraPro CD consists of the following files:

File	Description
<i>Instdrv.exe</i>	The Data Protection driver installer.
<i>Instdrv.c</i>	C source code of the <i>Instdrv.exe</i> utility for you. You can use it to customize the driver installation and registry modification procedure.
<i>Sentdata.vxd</i>	The Data Protection driver.
<i>Readme.pdf</i>	A readme file that has details on the installation.

You can either ship *Instdrv.exe* and *Sentdata.vxd* files to your customer with the following instructions on how to load the driver:

1. Copy the directory contents to a system.
2. Select **Run** from the **Taskbar** and run the *instdrv.exe* file.
3. When the message "Driver installed! Restart your system" appears, click **OK**.

4. Now, restart the system. The following files have been copied into your *System* folder and required registry entries are made:

- ❑ WINDOWS\SYSTEM\SENTDATA.VXD
- ❑ WINDOWS\SYSTEM\INSTDRVR.EXE

Otherwise, you can modify this installation program for your own installation needs, we have provided the C source code that installs this program.

You need to call the program to check for the type of operating system being used. You also need to remind customers that their computer must be rebooted after installing the data protection driver in order to load the driver.

Your installation program may call the data protection driver with the following command line options:

Option	Description
/P	Specifies the source path for <i>sentdata.vxd</i> . If not included, the installer looks for the <i>.VXD</i> file in the directory where the driver installer resides.
/U	Uninstalls the driver. The installation program returns a zero if it was successful; otherwise, it returns the error code of the last Win32 API call that had an error (if applicable).

The installation program returns a zero if it was successful; otherwise, it returns the error code of the last Win32 API call that had an error (if applicable). If the unsuccessful API call does not return an error, the installer returns a -1.

Deploying the Key Manager Application

For Windows

A self-extracting installer has been provided in the Sentinel UltraPro CD that will install the following components on a distributor's system:

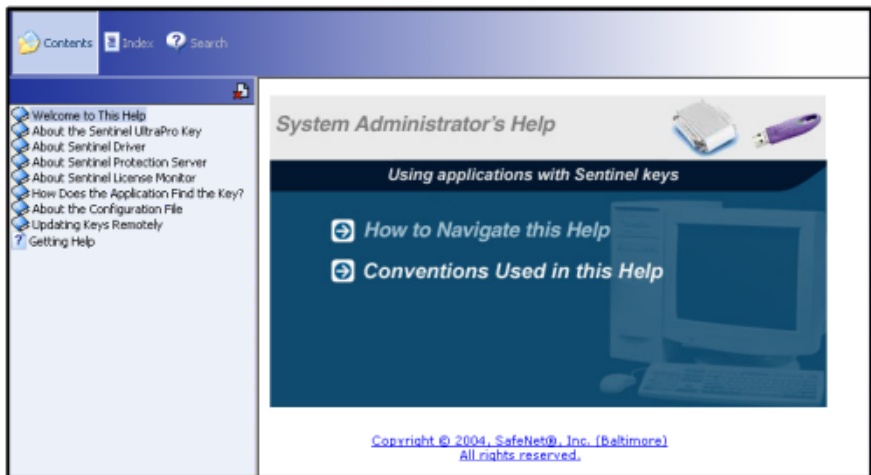
- The Key Manager application
- Sentinel System Driver

- Sentinel Client Activator
- The Field Activation Utility
- Java 2 Runtime Environment version 1.4.2 (only if not already existing on target the system)
- The Key Manager Help file.

Deploying the System Administrator's Help

This is an Online Help that contains basic information of interest to anyone who uses or administers software that has been protected using Sentinel UltraPro.

You can provide the complete Help to your customers, or use a portion of it, depending on the requirements. For example, you can extract the information on using the configuration file and include it in your product documentation, or send to your customers via e-mail.



The System Administrator's Help

For Windows

You can deploy the complete contents of the <install_dir>\SafeNet Sentinel\UltraPro\1.0\Manuals\English\SysAdminHelp directory. To launch the Help, click the *index.htm* file.

Your customers need Internet Explorer 5.0 (or higher) NetScape Navigator 4.6 (or higher) to view the Help.

Appendix A

Glossary

A

Access Code

An attribute that identifies the accessibility and functionality of a cell.

At low-level, the UltraPro key stores data the basis of the access codes. You are not required to deal with the access codes as such because this abstraction level is handled by the Toolkit Cell Addresses. The **Memory View** in API Explorer displays data on the basis of the access codes.

A cell can have any of the following access codes:

Access Code	Description
0	Identifies a read/write data word. Your application can read the values in the cell and modify its contents using the Write Password.
1	Identifies a read-only/locked data word. Your application can read the value in the cell, but cannot change it without the Overwrite Passwords.
2	Identifies a counter word. The cell contains a value that your application can decrement using the Write Password. The cell's value cannot be changed (other than by decrementing it) without the Overwrite Passwords.

Access Code	Description
3	Identifies a locked and hidden/algorithm word. The cell value is hidden and your application cannot read it. Modification requires the Overwrite Passwords.

See also, Toolkit Cell Address.

Access Mode

An access mode determines where and how your application will look for the key to obtain a license.

See “Which Access Mode To Use?” on page 60 for more information.

Action

Refers to a group of one or more field activation commands. See also, Command.

Activation Password

Refers to a two-word value that can be used to activate an inactive algorithm so it can be used for queries. This allows activation of an algorithm at a customer’s site.

The activation passwords are automatically associated with the algorithm-based elements (Full License, Short License, and Shell).

Active Algorithm

Refers to the algorithms that can return a valid response to a query.

The active/inactive bit of an algorithm word controls whether an algorithm is active or not. Whenever the second word value is between 0x8000 and 0xFFFF, it is active.

Active Application

Refers to an application that is ready to run (as long as the hardware key is attached) when shipped to your customer.

Address

Refers to the memory address of a cell in the key. See also, Cell and Toolkit Cell Address.

Algorithm Word

Refers to the two-word long algorithm descriptors that define how the key will scramble the query sent by your application. You design your application to send queries to the key and then evaluate and act upon the responses.

The algorithms are automatically associated with the Full License and Short License elements to bring the query/response protection into play. See also, Active Algorithm and Inactive Algorithm.

Algorithm-Based API Elements

Refers to the Full License and Short License API elements that allow you to use the query/response protection in your application. See also, API Elements, Data-Based API Elements, Query, and Response.

Application Programming Interface (API)

Refers to a set of client interface routines your application uses to communicate with the Sentinel driver, which in turn communicates with the key.

API Elements

Refers to a form of application protection in which the UltraPro API functions are to be added directly in your source code. Thereby, you can customize the application protection strategy, with control over the amount and location of the API calls.

This API elements based protection differs from Shell in which a protective layer is wrapped around the executable without requiring any source code or compilation effort.

Given below are the various API elements:

- Algorithm-Based API Elements
 - Full License
 - Short License
- Data-Based API Elements
 - Counter
 - Boolean
 - String
 - Integer

API Explorer

A screen in the Toolkit where you can experiment with the UltraPro API functions. You require a design and a key built with that design in order to use the API Explorer.

B

Boolean (Element)

Boolean is a data-based API element that can contain a true (1) or a false (0) flag. A single cell can accommodate 16 boolean elements.

C

Cascading

The connecting of hardware keys one after the other to the same parallel port is called cascading (also known as *daisy-chaining*).

Cell

Refers to a memory location on the hardware key that holds 16-bit values. An element occupies a single cell or a block of cells. Also referred as *Word*.

Cheat Counter

Refers to a counter limit that is decremented by one each time a minor time tampering is attempted. Using the cheat counter value, you can tolerate certain minor time tampering attempts. Once the cheat counter is exhausted, the `SFNTsntlQueryLicenseLease` and `SFNTsntlQueryLicense` function will return an error. See also, *Minor Time Tampering*.

Client Activator

Refers to an automated key update utility that is used to activate applications/features in the field. It is a recommended means of field activation for UltraPro-protected applications, due to its user-friendly interface. You may instead use the Field Activation Utility or create your activation utility (using the SAFE functions).

Code Sketch

Refers to a protection plan that outlines the UltraPro API functions that you should incorporate in your source code. It is a good reference when you are not sure about which API functions are relevant for the elements in your design.

Toolkit generates the code sketch for the most-frequently used development languages (ANSI C, Visual Basic and Pascal) in the working folder when the **Build** button is clicked. See also, *Working Folder*.

Command

Refers to the function calls that describe what will be done to a key in the field. For example, the Decrement Counter command locates the Counter word on the key and decrements it by the value you specify. See also: *Action*.

Counter (Element)

Refers to an API element used to count down from a pre-programmed value. The value is decremented each time your application calls the SFNTsntlDecrementCounter function. It uses a counter word in the key. See also, Counter (Word).

Counter (Word)

Refers to a cell with an access code 2. It contains a word (value) that can be decremented using the Write Password. The cell's value cannot be changed (other than by decrementing it) without the Overwrite Passwords.

D

Data-Based API Elements

Represents the Counter, String, Integer, and Boolean API elements that let you store and manipulate certain data in the UltraPro key. See also, API Elements and Algorithm-Based API Elements.

Data Word

Refers to a cell with 0 access code. These cells can be read by your application. To write them, you require the Write Password.

Design

Refers to your application protection strategy made up of one or more elements. The UltraPro Toolkit follows the hierarchy of Project-Design-Elements, where a project can contain multiple designs, while a design can contain multiple elements. Please note that an UltraPro key can be programmed with only one design. See also, Design ID and Element.

Design ID

Refers to a 16-bit value generated by the Toolkit for the design you created. The design ID is written into the key during the key programming and is

used (along with the developer ID) by the SFNTsntlGetLicense function for finding your keys on the customer's site. The design ID can be obtained from the header file or the **Design Properties** dialog box.

Developer

An individual or a software development company that uses the Sentinel UltraPro protection system to protect and license their applications.

Developer ID

An identifier of the UltraPro key that represents the developer or an organization. This is assigned by the entity from whom you purchased the UltraPro keys. The developer ID is stored in the cell 01 and it cannot be moved.

It is passed as a parameter in the SFNTsntlGetLicense function for finding the key and obtaining a license.

Distributor

An entity (such as resellers or fulfillment centers) authorized to distribute your protected application. They can use the Key Manager application to program keys and generate license codes for customers.

Distributors must receive a distributor key and the design file (*.dst*) to operate.

Distributor Key

A key that allows your distributors to program product keys and generate license code in order to sell the your protected application to customers. The distributor key carries a counter to track the number of keys programmed and license codes generated by your distributors.

DUSAFE

Stands for *Developer's UltraPro Secure Authenticated Field Exchange*. The DUSAFE and UUSAFE objects form a unique pair and are used for field acti-

vation. While the UUSAFE object is deployed on the customer-end to generate the locking code; the corresponding DUSAFE object must be deployed with the developer/distributor, whosoever generates the license code.

DST

Refers to the format of the file that is provided to your distributor who will be programming keys and activating applications for customers. A *.dst* file is made for a design.

E

Element

An element is the basic unit of a protection strategy that you create in Toolkit. The various elements offered by the Toolkit are: Full License, Short License, Integer, Boolean, String, Counter, and Shell.

A design can contain as many elements as you want. However, a key can be programmed with one design. See also, Project and Design.

Element View

An option in the API explorer screen that shows a graphical layout of the design (and its elements) you are evaluating. This view is enabled only when the SFNTsntlGetLicense function successfully obtains a license.

Execution Count

The maximum number of times the application will run. You can specify an execution count value for a Full License and Shell element ranging between 1 and 65,535. The counter is decremented every time the application is launched. The algorithm is disabled when the counter is exhausted

When “unlimited” execution counts are specified, the application runs without decrementing the counter.

Expiration Date

Refers to a fixed expiration date (month, day and year) after which the protected application will not run.

F

Field Activation

The process of activating applications/features in the field. This same process also allows you to support field upgrades and control feature access.

Full License

An algorithm-based API element that offers maximum licensing options—demos, lease, trials, number of users, and so on—along with the query/response functionality. See also, Short License (Element).

H

Hard Limit

The hard limit is the factory-programmed limit that defines the maximum number of licenses that can be obtained from a key. The UltraPro keys come with the following hard limits: 1, 2, 3, 5, 10, 25, 50 or unlimited. See also, User Limit.

Header File

The header file is created for a design in the working folder when a design is built using the **Build** button in the **Protection Manager** screen. It contains important information about your design such as, the query/response table (if you included a Full License or a Short License in your design), design ID, Toolkit Cell Addresses, Developer ID, and Write Password and Overwrite Passwords. You must keep it secure!

Heartbeat Interval

Refers to the time interval used by the Sentinel Protection Server for maintaining a license. If no communication takes place between the application and the UltraPro key for 120 seconds, the Sentinel Protection Server will release the license.

You can customize the heartbeat interval using the `SFNTsntlSetHeartBeat` API function, or allow your customers to edit it in the configuration file (*sntlconfig.xml*). However, the heartbeat set using the `SFNTsntlSetHeartBeat` function will always override the value set in the configuration file.

I

Ignored Key

Refer to a Sentinel key ignored by the UltraPro Toolkit. The ignored keys are not recognized by the Toolkit, unless re-configured using the option provided by the **Settings** menu item.

Inactive Algorithm

Refers to an inactive (disabled) algorithm in the key; meaning that it cannot be used for a query until it is activated.

The active/inactive bit of an algorithm word controls whether an algorithm is active or not. Whenever the second word value is between 0x0000 and 0x7FFE, it is inactive.

Integer (Element)

A set of data-based API elements using them you can write 8-bit, 16-bit, and 32-bit long integer values in the UltraPro key's memory.

K

Key Layout

The **Key Layout** option on the **Protection Manager** screen provides a virtual layout of the design and its elements in the key. You can also rearrange the various elements by cutting and pasting them. Refer to Toolkit Help for details.

L

LCK File

The file in which the Field Activation Utility copies the locking code. Your customer may send this file to you/distributor for obtaining the license code. See also, LIC File, Locking Code, and License Code.

LIC File

The file in which the license code generated using the **Key Activator** option is copied. You may send this file to your customer so the key can be updated in the field. See also, LCK File and License Code.

License

A license is an agreement under which the user is granted the right to use an application in the manner specified in the software license agreement.

License Code

A code that describes the actions to be performed on a key in the field. It determines how the application will be activated or updated. See also, Locking Code.

License Model

A license model refers to the rules that govern the permissible usage of a protected application.

License Sharing

When unlimited instances of an application can be run by using only one license. Sentinel UltraPro allows default license sharing for a seat. This default sharing of licenses for a seat cannot be turned off. If you have specified a user limit for your element, you can customize its sharing on the basis of user name or MAC address. See also, User Limit, Hard Limit, and Seat.

Last Known Date and Time (LKDT)

Stands for *Last Known Date and Time*. It is the date and time lastly written into the key. A time tampering condition exists whenever the LKDT programmed into the key is ahead of the system's current date and time. This indicates that the system clock has been tampered with.

The key initially bears the date and time when it was programmed using the **Make Keys** option. Each time a query call is made to the key using the SFNTsntlQueryLicense or SFNTsntlQueryLicenseLease functions, LKDT is updated.

Locking Code

Refers to a code that includes information about how a key is currently programmed, including the key's serial number and developer ID. You must have a customer-generated locking code to create a license code. Locking codes are unique for each key. See also, License Code.

M

Major Time Tampering

When the system clock is set back by more than 30 days, it is referred as major time tampering. In this case, the associated algorithm automatically deactivates denying access to the protected application.

Manufacturing Code

This code identifies the key's manufacturing lot. It is helpful if a key is returned back for any reason.

Minor Time Tampering

When the system clock is set back by more than 90 minutes and less than 30 days, it is referred as minor time tampering.

Memory Meter

A progress-bar like indicator in the **Protection Manager** screen that gives an approximate idea about the memory size of the UltraPro key required to fit the design you have prepared.

Memory View

An option in the API Explorer screen that provides the low-level, graphical view of the elements—such as the memory cell addresses and values—based on the access codes. The values shown are in hexadecimal format (“0-9” or “A-F”).

This view is enabled only when the SFNTsntlGetLicense function successfully obtains a license. See also, Element View.

MAC Address

Stands for *Media Access Control Address*, a hardware address that uniquely identifies each node of a network.

Sentinel UltraPro lets you share the licenses issued on the basis of the user name, MAC address, or seat if you have specified a user limit in your strategy. By default, licenses issued to a seat will be shared. See also, License Sharing, User Limit, Hard Limit, and Seat.

N

Network Application

If the application is a network application, only one key—located on the network—is required. The single key can issue multiple licenses, allowing for simultaneous use of your application by several clients. Sentinel UltraPro provides various access modes that define how and where the application will search for the key. See also, Stand-alone Application.

NSP_HOST Environment Variable

Refers to an environment variable using which your customers can set the access mode to find a key on the network.

O

Overwrite Passwords

A set of two passwords for the key that allow you to unlock the locked data words (access code 1) and change the low-level state of any cell (barring the reserved cells).

These passwords are assigned by the entity from whom you purchased the UltraPro keys. Keep them secure; they have the power to reprogram all unrestricted cells in your key. See also, Write Password.

P

Part Number

A part number is a customer identifier and is used for key ordering purposes. The first character is always “S” identifying the key as a Sentinel UltraPro key. The next two characters identify the distributor and the last six characters are the sequence numbers. Please note that the part number is NOT the same as the developer ID programmed into the key. See also: developer ID and Manufacturing Code.

Project

A project is a container of designs. It comprises of one or more designs made up of the protection elements.

A project can be duplicated and imported/exported across different Toolkit installations to facilitate easy management of your strategies. See also, Design and Element.

Q

Query

The value an application sends in a query to the hardware key. The key scrambles the string according to its internal logic and the bit pattern of the algorithm descriptor (word) you specify (in Full License or Short License element). It then returns a response to the application for evaluation. See also, Active Algorithm, Inactive Algorithm, and Response.

Quick Shell

Refers to a tool/screen in the UltraPro Toolkit where you can quickly protect an executable with basic controls. It is best used for getting familiar with Sentinel UltraPro and Shell. See also, Shell.

R

Response

The scrambled result derived when the hardware key processes query data according to the bit pattern contained in an algorithm. The hardware key returns the response string to the application. The application then uses the response to determine whether the user is authorized to run the application.

If the response obtained is different from the query sent, the algorithm is active (enabled). If the response returned is same as the query sent, the algorithm is inactive (disabled). See also, Active Algorithm, Inactive Algorithm, and Query.

Reserved Cells

Refers to the cells in the UltraPro key that contain fixed, factory programmed key information (such as the serial number, developer ID, and passwords) or are written by the UltraPro Toolkit for implementing your protection strategy. These cells are fixed and cannot be moved to any other location in the key.

S

Seat

Refers to the combination of a user name and MAC address. Licenses are shared for the applications accessed from the same seat.

Secret Code

Refers to a string consisting of 9 to 16 characters, with at least one numeral. It is used for creating the field activation objects—the DUSAFE and UUSAFE pair.

Sentinel License Monitor

Sentinel License Monitor shows the details of the keys and clients accessing them via a Web browser. It is a convenient way to view and track license activity and analyze application usage.

The Sentinel Protection Server makes use of the HTTP protocol for fulfilling the license monitor requests from clients on the LAN/WAN.

Serial Number

Refers to the hardware key serial number—a 32-bit value sequentially assigned per key. The serial number is pre-programmed, readable, and cannot be modified.

Serial numbers are not guaranteed to be unique. If you require unique serial numbers, please contact the Sentinel sales representative before ordering keys.

Shell

Sentinel UltraPro's automatic protection method in which a protective Shell layer is wrapped around your executable. This layer is encrypted and handles all the communication with the key. Hence, the application will not run unless the correct key is found. Further, while the application is running, the layer periodically checks to verify the hardware key is still attached—if at any time the key is missing the application shuts down immediately. Shell also allows you to program expiration dates, counters time/date limits and so on. Generally, Shell is used when you do not have access to the application's source code and want to protect applications instantly.

Short License (Element)

An algorithm-based API element suitable for protecting fully-licensed applications with/without the user limit. See also, Full License.

Software Lock

A decision point in an application. The purpose of a software lock is to verify the presence of the correct hardware key.

For example, an application might send query data to the hardware key, and require a specific response in order to continue execution. Other software locks may simply read the value in a cell and compare it to the value known to be programmed in that cell.

Stand-alone Application

If an application is stand-alone, each user needs his own hardware key to run the application. This is because both the application and the key exist on the same system (may be networked or non-networked). For stand-alone applications, generally keys with one hard limit are used.

Sentinel UltraPro provides specific access modes for stand-alone applications viz, `SP_STANDALONE_MODE`, `SP_DRIVER_MODE`, and `SP_LOCAL_MODE`. Refer to the topic “Which Access Mode To Use?” on page 60 for more details. See also, Network Application.

String (Element)

Refers to an API element using which you can write a string—consisting of maximum 50 ASCII characters—into the UltraPro key's memory. This string can be read and written using the `SFNTsntlReadString` and `SFNTsntlWriteString` functions, respectively.

Sentinel Protection Server

Sentinel Protection Server is the license manager of your networked applications. It manages the licensing needs of the protected applications that concurrently access the key(s) attached in the network. A single Sentinel Protection Server can manage licenses for up to 10 keys (any combination of parallel and USB keys). It **MUST** be shipped with the protected applications meant to run in the network.

The Sentinel Protection Server also runs an HTTP thread to allow license monitoring over the LAN/WAN. See also, Sentinel License Monitor.

T

Toolkit Cell Address

Refers to a 32-bit integer value generated by the UltraPro Toolkit to map an element or its control (such as user limit, execution control) on the key.

The Toolkit Cell Address provides a high-level interface to a developer who is provided with the task of adding UltraPro API into the application's source code. The Toolkit Cell Address internally handles several complexities as it can distinguish between the various types of elements, identifies the corresponding cell addresses, data length, and points to location of the data (including bits and bytes).

The header file generated by the Toolkit contains the Toolkit Cell Addresses.

Trial Days

Refers to the number of days your protected application will run for. This option is enabled only if you have specified a fixed expiration date.

The actual expiration date for the application is calculated on the basis of the (fixed) expiration date and trial days you specify. As a result, the application will not run as soon as the relative expiration date is reached.

U

UUSAFE

Stands for *User's UltraPro Secure Authenticated Field Exchange*. The DUSAFE and UUSAFE objects form a unique pair. The UUSAFE object is deployed on the customer-end to generate locking code.

If you wish to activate applications/features in the field, you **MUST** ship the UUSAFE object along with the protected application and key.

User Limit

A limit—less than the hard limit—that defines the number of users for a UltraPro key. The user limit can also be understood as a “soft limit” because it allows you to customize the factory-programmed hard limit.

The licenses issued against a user limit can be shared on the basis of a user name, MAC address, or seat.

W

Word

See “Cell” on page 181.

Working Folder

A directory on your system where the project and design-related files are written.

Write Password

A password for the key assigned by the entity/organization from whom you purchased the UltraPro keys. It is used for performing write operations on the cells with access code 0.

Index

Symbols

.dst file 15
.rac 172
.sud 164
.sup 164

Numerics

8.3 naming convention 84

A

access modes 60–63
action, field activation
 adding 112
activate algorithm,
 command 113
activation password
 adding 91
add distributor counter,
 command 113
address, cell 33
ainst.exe 172
algorithm descriptors 10
 adding 91
ankle straps 42
API elements
 about 13, 46
 adding 89–102

API Explorer

 Key Status 29
 screen 26
API, UltraPro 23
application protection
 designing
 API elements 89–102
 field activation 93–115
 Shell 73–86
 implementing
 final steps 129–135
 tricks and tips 117–127
 planning 45
attaching keys
 steps 37
 using cables 40
attacks, basic types
 application 118
 hardware 118
 SafeNet code 118
authorizing
 distributors 71
automatic protection 74

B

bit mask AND/OR,
 command 113

Boolean, element

 adding 97
 relevant API 54
build options
 setting 131
building design 131
built-in driver 167

C

cables 40
cascading 38
cell
 definition 33
 reserved 34
cheat counter 57
checksum code 125
Client Activator
 configuring 147
 deploying 172
 using 109
client for Novell
 NetWare 168
code sketch 14, 131
cold plastic 42
commands, field activation
 definition 112

- comparing
 - API elements 55
 - Client Activator and Field Activation Utility 107
 - Full and Short License 48
 - Shell and API elements 46
 - SP_STANDALONE_MOD E and SP_DRIVER_MODE 62
- compatibility
 - hardware keys 39
- components, UltraPro 17–24
- conductive plastic 42
- configuration file
 - priority 170
- conventions, manual xiii
- Counter
 - adding 98
- Counter element 114
- Counter, element
 - relevant API 54
- customizing
 - element settings 144
 - license monitor 20

D

- daisy-chaining 38
- data protection driver
 - deploying 173
 - purpose 83
- decentralizing security
 - checks 123

decrement

- command 114
 - counter to zero* 114
 - execution counter* 114
 - execution counter to zero* 114
 - time tamper counter* 114
 - time tamper counter to zero* 114

deploying

- Client Activator 172
- configuration file 170
- data protection driver 173
- Field Activation Utility 171
- Key Manager 174
- on Novell platforms 167
- Sentinel driver 165
- Sentinel Protection Server 166
- system administrator's help 175

design

- definition 29
- programming key 32
- design ID 32
- developer 183
- developer ID 119, 183
- directed call to system 61
- dissipating mat 42
- distributor 183
- distributor key
 - about 34, 183
 - programming 149–155
- DUSAFE 110, 183

E

- electrostatic charges 41
- element
 - adding
 - Boolean* 97
 - Counter* 98
 - Full License* 91
 - Integer* 100
 - Shell* 79
 - Short License* 94
 - String* 96
 - properties 134
 - types 30, 46
- encrypted data files 173
- environment variable
 - NSP_HOST 64, 190
- Ethernet 802.2 168
- export considerations xvi

F

- field activation
 - adding actions 111
 - generating objects 110
 - methods 107
 - process 104
- field activation objects
 - backing up 111
- Field Activation Utility
 - about 107
 - deploying 171
 - using 109
- Full License
 - adding 91
- functions, UltraPro 23

G

- GMT 56
- grounding wrist 42
- guidelines
 - packaging keys 41

H

- hard limit 34
- header file 130
- header file generation
 - building design 131
- Help
 - Field Activation 171
 - Key Manager 175
 - Sentinel Protection
 - Installer 167
 - System
 - Administrator's xii, 175
 - UltraPro Toolkit xii
- hhactiveex.dll* 164
- hhupd.exe* 164
- hidden passwords 12
- hierarchy, *project-design-elements* 14, 31

I

- ignored keys 28
- Include Shell elements
 - check box 120, 132
- increment counter,
 - command 114
- installation 25
- installldr* xiii
- Instldr.exe* 173
- Integers, element
 - adding 100
 - definition 53, 186
 - relevant API 54
- IPX
 - on Novell 168

K

- key
 - memory layout 35

Key Configuration, dialog

- box 28

Key Layout

- element properties 134
- view 133

Key layout

- rearranging
 - elements 134

Key Manager

- application 15
- deploying 174
- Help 175
- screen 26

Key Status pane 28**keys**

- attaching 37
- compatibility 39
- NetSentinel 39
- ordering/returning 41
- packaging 41
- types 39
- using cables 40

L

- languages, supported
 - building design 132

license monitor

- See *Sentinel License Monitor* 19

license sharing 70**licensing options** 47**licensing schemes** 8**limit**

- hard 34
- user 34, 67, 70

list

- redistributables 164

LKDT 56**M**

- major time tampering 56
- manually rearranging
 - elements 33
- memory, key
 - data storing 10
 - layout 35
- methods
 - field activation 24
 - protecting software 13
- minor time tampering 56
- missing hardware
 - keys 126
- models, licensing 8
- modify licensing
 - settings 15, 144
- modifying licensing
 - settings 144
- Msvrt.dll* 164
- multiple keys,
 - attaching 38

N

- NetSentinel keys 39
- network applications 7, 190
- network keys 34
- network protocols 19, 168
- Novell NetWare 167
- NSP_HOST 64, 170, 190
- nspdns.nlm* 168
- number of keys
 - supported 38
- number of queries
 - building design 132

O

- one-time update
 - feature 12, 112
- ordering keys 41

OS drive xiii
Override check box 144

P

packaging keys 41
parallel key
 attaching 37
 Key Status panel 28
 multiple keys 38
part number
 ordering keys 41
PCI card support 39
Personal folder xiii
planning application
 protection 45–72
product key
 about 34
 programming 142–147
programming keys
 distributor 149
 product 142
project
 definition 31
project-design-elements 14, 31
Protection Manager
 screen 26
protection methods 13, 24
Protection Wizard 27
prototyping design 130

Q

query size
 building design 132
query/response protection
 about 10–12
 table 122, 130

Quick Shell
 about 26
 adding 76
 saving design 78

R

randomly querying 122
read
 API 54
read/write memory 10
rearranging elements
 Key Layout 134
record and playback
 attack 118
redistributables 163
regulations, export xvi
repairing keys 41
reserved cells 34
returning keys 41
ribbon cables 40
RMA 41

S

SAFE objects 110
sales distributors 71
sample designs 14
scrambling data 10
screens, Toolkit
 API Explorer 26
 Key Manager 26
 Protection Manager 26
 Quick Shell 26
seat 70
secret code 110
Sentdata.vxd 173

Sentinel driver
 about 5, 18
 access modes 60
 deploying 165
 number of keys
 supported 38
 PCI card 39
Sentinel Express 109
Sentinel License Monitor
 about 19
 deploying 167
Sentinel Protection
 Installer
 about 16, 24
 Help 167
Sentinel Protection Server
 about 7, 18
 for Novell 167
 number of keys
 supported 39
setting
 access mode 64
 build options 131
 cheat counter 57
SFNTsntlDecrementCounter 54
SFNTsntlLockData 54
SFNTsntlQueryLicense 51, 56
SFNTsntlQueryLicenseDecrement 50
SFNTsntlQueryLicenseLease 50, 56
SFNTsntlQueryLicenseSimple 50
SFNTsntlReadString 54
SFNTsntlReadValue 54
SFNTsntlSetContactServer 170

SFNTSntlSetHeartBeat 17
 0
 SFNTSntlSetProtocol 170
 SFNTSntlUnlockData 54
 SFNTSntlWriteString 54
 shipping protected
 application 163
 Short License
 about 48, 52
 adding 94
 relevant API 50, 94
sntlconfig.xml 64, 170
 soft limit 34
 software locks 60
 SP_ALL_MODE 61
 SP_BROADCAST_MODE 6
 1
 SP_DRIVER_MODE 60
 SP_LOCAL_MODE 60
 SP_SERVER_MODE 61
 SP_STANDALONE_MODE
 60
SPNNWSRV.NLM 168
 stand-alone access
 modes 60–63
 String, element
 adding 96
 definition 194
 relevant API 54, 96
 symmetric encryption
 algorithm 12
 System Administrator's
 Help 175

T

TCPIP.NLM 168
 technical support xiv
 terminology, key 33
 time tampering
 conditions 56

time-out 20
 Toolkit
 about 21–22
 Key Status pane 28
 Protection Wizard 27
 screens 25
 tools 25
 video tutorials 27
 tools
 Toolkit 25
 tricks and tips
 application
 protection 117–127
try-or-buy option 109

U

UltraPro
 API 23
 key, about 18, 36
 Toolkit, about 21, 25
 unique solutions 119
 unknown key 29
 update LKDT,
 command 115
 USB key
 attaching 37
 Key Status panel 28
 number of keys 38
 user limit
 about 34, 67
 license sharing 70
 UUSAFE 59, 110, 164

V

video tutorials, Toolkit xii,
 27
 viewing
 keys details 19
 license details 20

W

Web site, technical
 support xiv
 WHQL 18
 Windows NT, USB
 support 37
 write
 API 54
 command
 boolean 115
 execution counter 115
 integer 115
 lease date 115
 string 115
 user limit 115
WS2_32.nlm 168

Z

Zip drives 39

