Virbox User Manual

Virbox License System Architecture

Version 2.1

Copyright & Trademarks

The **Virbox**, **Virbox LM**, **Virbox Protector**, **Virbox Elite 5**, **Virbox EL5 Acme** are Registered Trademarks of SenseShield in China and other countries.

All products referenced throughout this document are trademarks of their respective owners.

Disclaimer

All attempts have been made to make the information in this document complete and accurate. But we cannot guarantee everything is perfect, we will correct it in next version released in case some error has been found. Senseshield 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.

Documentation Improvement

Any suggestion to this manual from you are welcome, We are glad to hear any feedback from you which will help us to continuously improve the documents quality and support and serve the developer to protect software products more efficiently.

Contact

Company: Beijing Senseshield Technology Co., Ltd

Address: Suite 510, Block C, Internet Innovation Center, Building 5, No.10, Xibeiwang East Road, Haidian District, Beijing China

Tel: +86-10-56730936

Fax: +86-10-56730936-8007

Sales: info@senselock.com;

Website: https://lm-global.virbox.com/

Virbox Developer Center Website: https://developer.lm-global.virbox.com/

About this document

This document is designed to help Software developer to learn about the Principle, System Architecture and License Terms of Virbox License System.

Target User: R & D of software developer who responsible to design and develop the concrete license scheme to software product.

Virbox Software Protection and License Solution (Virbox LM or Virbox Cloud) supports to protect and licensing to the software program running in **Windows, Linux and Mac** environment.

Virbox support to protect/encrypt software application includes:

Windows: .exe, dll;

Linux: executable file and .so

Program Language includes: C/C++, C#, Java, .NET, VB6.0, VC6, VC2003, VS2005-VS2017, Delphi7 DeliphXE-XE10, BCB6.0, Unity 3D, UE4

Plugin can be protected with **Virbox Protector:** **AutoCAD ARX, Revit** etc.

Objective: Software developer will understand to Virbox License System architecture, essential concept of "Product, Template, License", etc and understand the License Terms of Virbox System, Developer will have competence to design dedicate license scheme to each concrete software product, according to software sales strategy to different market segment.

For Software developer who want to use Virbox LM to protect and license the software and learn the process of software protection and licensing, please refer to:

User Manual:

 ***Software Developer Guide*;**

For More detail information for how to protect the software by use of Virbox Protector, please refer to:

User Manual:

 ***Virbox Protector*;**

For More detail information for how to protect the Data file or other digital resource by use of Virbox Protector DS (plugin), please refer to:

DSProductor Description and Use Case:

 ***DS Product Description & Use case;***

Virbox also provide a series **Quick Start Guide** to help developer to quickly use Virbox solution to protect software and issue license to hardware lock (dongle), Cloud License and soft license, please refer these "quick start guide" documentation:

Virbox Provides Virbox User License Service and User License Tools which installed software user premise which for proactive software protection service (runtime service) to safeguard software copyright, IP, and license verification to software user, for more detail information how to install and use this Virbox User License Tools in user computer, please refer to:

User Manual:

 ***Virbox User License Tools*;**

Table of Contents

1 Virbox License System Overview

This document is designed to help Software developer to understand principle and architecture of Virbox License System and describes the concept and relevant definition of license. And how to use Virbox LM to protect your software.

With more than 20 year experience in software copyright protection and Intellectual Property Protection, SenseShield launched the latest software protection & License Entitlement Solution: Virbox LM (Virbox Cloud) Solution.

Virbox LM Solution is designed and developed with new license management concept and support developer to enhance the capability in software protection and flexible issue the License to protected software with latest technology in proactive monitoring, Anti hackers Service. With Deployment and Achievement in Software encryption, Anti-Reverse Engineering, anti-debugging techniques and other technology, Virbox LM Solution represents the most advanced technical trends in Software Protection and License Management.

Virbox License System Architecture consist of following contents: License, License Entitlement management and License Service,

The license, serves as the top-level concept for Virbox Solution and all operations to software are based the license. This document describes in detail of the concept and usage of the Virbox License Solution and Architecture. It will provide a comprehensive explaination to Virbox Principle in licensing Architecture, licensing management, and license entitlement.

2 Key Words

Prior to this, in order to facilitate the reader a clear understanding of some of the content described in the document, it is necessary here to explain some key words and terms in Virbox LM.

The Virbox system is based on the development of a secure repository developed by SenseShield. It provides secure repository for local, network, cloud, and soft locks. The "lock" mentioned in this article, generally refers to all secure repository to store the license unless otherwise specified.

The term of "hardware lock" mentioned in this document generally refers to the Virbox EL 5 Series dongle , Also refers to the local lock, network lock and Developer Master Lock;

The term of "Cloud lock" is the repository which store the cloud license used in online environment;

The terms of "Soft Lock" is the repository which stored the soft license used in periodical online or offline environment.

Software License, is a digital certificate that is signed by the software developer and issued to the software user which specify and limited the software user's rights to use the software (or its source code) and defined the developer's obligations to software.

The Principle of Virbox License Solution is controlled and managed the permission of user to use the software with "license" and also to use multi advance protection technology to protect the copyright and IP of Software,

with Virbox LM solution, developer may have more flexible and effective ways to promote software sale, trial the software to potential user without traditional dongle protection, Virbox LM also help developer to effectively identify the dongle lost claimed by software user and avoid developer and user potential profit lost.

Virbox LM is the software protection and licensing solution to software developer with top security protection and keeps licensing flexibility for both. It helps developer to design and maintains secured software Encryption and protection plan and no license leakage risk coming from internal.

Virbox LM is suitable to be deployed in different kind of software developer or software vendors, from freelancer, SME vendors or large size of software enterprise.

Virbox LM supports with different license modes with different license repository: Node Locked, Network/Floating/Concurrent, Perpetual, Subscription, Usage Consumption, soft license and cloud license etc.

Virbox License consists of following contents: Product Information and setup a Limitation to product, Product Information may contains product ID, Product modules etc. Developer may setup limitation with following terms: setup license startup time, expire time, usage counts, the No. of the concurrent Nodes etc.

With Virbox LM solution and relate unique Digital Certificate endorsement, each developer may create/issue licenses to products/product modules more than 4 billion.

For Each Virbox EL5 Hardware Lock (standard version), maximum No. of license stored up to 3000 licenses, For Virbox EL5 Ultimate version, maximum License stored up to 6000 license; and No maximum limitation for qty.of license stored for Virbox Cloud Lock and Soft Lock.

2.1 License Terms

License Terms	Description	Access Right: Read Permission	Access Right: Write Permission	Note
flag	Flag of Licensed Feature	No limitation to access right	Master only	Binary Indicator to ensure if the licensed feature enabled; The way of use and judgement to the each flag follows to the bitwise operation.For Specific meaning to each flag of licensed feature, refer to following description: Description to Flag of Licensed Feature
version	License version	No limitation to access right	Master only	Current license version
LicenseID	License ID	No limitation to access right	Master only	Developer can setup License ID which varies from 1 to 4294967295，and License 0 is set in manufacture period and reserved for management purpose for developer and can't be deleted and modified.
start_time	software valid	No limitation	Master only	User can't use software before this

	start time	to access righ		"start time" setting, which time setting can be varied from:"2000.1.1 00:00:00" to "2099.12.31 23:59:59"
end_time	software Expire time	No limitation to access right	Master only	User can't use software if later than this setting time , which time setting can be varied from：2000.1.1 00:00:00 to 2099.12.31 23:59:59
first_use_time	first time to start to use software	No limitation to access right	-	The time of first time to login user account which will be used to issue time span based license
span_time	Time Span	No Limitation to access right	Master only	The Time span: start from License start time of first used
counter	Set the Counters of Software usage	No limitation to access right	Master only: to issue and set the counter for software usage	decrement when every time to login and use software, and license will be invalid when counts to 0
concurrency	Maximum No. of concurrent user, Nodes of Network, Process	No limitation to access right	Master only	For Setup the No. of Concurrent Nodes or users for Network Lock and set a limitation of licenses to use software in Network environment.
ROM	Read only memory	valid for licensed user	Master only	Data area which data saved can be read by software user (login license) but can't be write into this area when Login license, refer to License Data Area
RAW	Read and Write Memo	Valid for licensed user	Master only and valid after login license	Data Area which supports to read and write data to this area after login license, but the length of the write data cannot exceed the maximum size defined when the license is issued.
PUB	Public Data area	No limitation to access right	Master only	Data can be accessed but can't be write in，user visible
Algorithm for user executed	The algorithm defined by software developer	No access right	Master only	Algorithm can be bound with one or more license ID, algorithm will be executed after verify the bound license successfully, only valid for the specific bound license.
module	Module	No limitation to access	Master only	Read permission only

		right		
time_stamp	time stamp to license update	No limitation to access right	Master only	the time which license issued based UTC time in seconds (s)
serial	time stamp to license update	No limitation to access right	Master only	the series No. of those licenses issued within 1 second based UTC time in seconds

Note: "Master Only" means only the Administrator of Developer who use Master Lock has the access right to issue/change/update the relate license

Description to the Flag of Licensed Features

The flag to licensed features is an enable bit that is used to control whether the license terms are available. Developers don't really need to care about their specific numerical values, they just need to know what the specific meaning is.

Item	Meaning for:
L1	Limitation to time, terminate
L1	Limitation to time, start
L1	Limitation to time, license to time span (License will be valid within n days)
L1	Limitation to the counter for usage of software
L1	Concurrency qty., set the limitation with concurrency system session
L1	Concurrency qty., set the limitation with concurrency system session
L2	If it is reserved and used by license 0, set this flag to disable all licenses.
L2	Setting this flag which the license assigned to be the license of account (not available yet)
L2	Setting this flag and check if the license valid or not

2.1.1 The Structure of License Terms

License Terms can be obtained via call related API, License Structure information can be obtained by use of Control API, more information for how to use the control API, please refer to calling API interface from the reference of ss_lm_control.h;

License Data Area

License Data area consist of ROM (Read Only Memo), RAW (Read and Write area) and PUB (Public Area), the size of License data area was defined and assigned when create the license in first time, the license data area can be modified or expansion via remote update/upgrade later, the maximum size of access area to these data area will not exceed the sized which assigned in advance, login the license is needed before access license data area.

Access Right and Description to License Data Area ：

License Data Area	Description	Size of Data Area	Note
ROM	Read Only Memory	0-65535 Bytes	data stored in this area can be modified by use of master lock only
RAW	Read and Write	0-65535 Bytes	the data can be accessed (data can be read and write)

	Area		after login license
PUB	Public Area	0-65535 Bytes	data in this area can be read without any limitation

Read Only Memo Area

Developer may use this ROM area to save the data required by software, such as some configuration information, ROM is more frequently used than RAW, because the data saved in this area can't be modified freely without licensed.

The only way to modify the data stored in ROM area is developer use the master lock to issue an update package file to remote update the data in the ROM, and use software tools to write into the hardware lock and update the data located in the ROM.

Read and Write Data Area

RAW allowed developer to save the data in execution process into the lock and may be accessed to read and used when software launch next time.

Public Area

The data Stored in the Public area allows developer to access with read only permission, developer may use this data area to save the product information of the software, Data stored in Public Area will be displayed and showed via Virbox User License Tools, the modification to the data saved in Public area will be only rely on to issue remote secure update package.

Access the Data Area

Virbox API provides a whole sets of API interface to operate/access the data area, which includes:

slm_user_data_getsize，slm_user_data_read，slm_user_data_write，slm_pub_data_getsize，slm_pub_data_read

For Detail information to API interface, please refer to ss_lm_runtime.h.

2.1.2 License 0

Virbox License classify the licenses with 2 kind of license: License 0 and the ordinary licenses. License 0 refers to the license which License ID is 0, the rest License ID is ordinary license used in normal. License 0 set to be the on default license, which set during manufacture period, license 0 is perpetual license and never expire, the license terms of license 0 is same as with other license terms, but the License 0 has its own special features.

License 0 provides developers with general license management functions. All other licenses can be controlled by license 0. For example, lock all other licenses by locking the license 0. Once issue "locked all licenses", all licenses will be locked. The flag of License 0 will be set to be the "locked" status and all the other licenses will not be available (since license 0 is perpetual license and never expired, license 0 can still be used).

License 0 can't be deleted by software developer, but developer can modify the flag of license features and license terms of License 0

2.2 License Mode

License Mode	Description
Perpetual License	perpetual license and never expired

Concurrent License	This license stored in the Network Lock (El5 Hardware lock with network mode) located in the Server of Network, developer may setup the limitation to total qty of license based to computer nodes, applications connected and use this concurrency license, each connection will takes up one concurrency number
Trial License	Software user may use the part of feature or modules of software in the time period which grants by developer
Subscription License	The license expired time has been specified by software developer
Feature on Demand	Developer may set up the license to activate to specific features or module of software

3 License Management

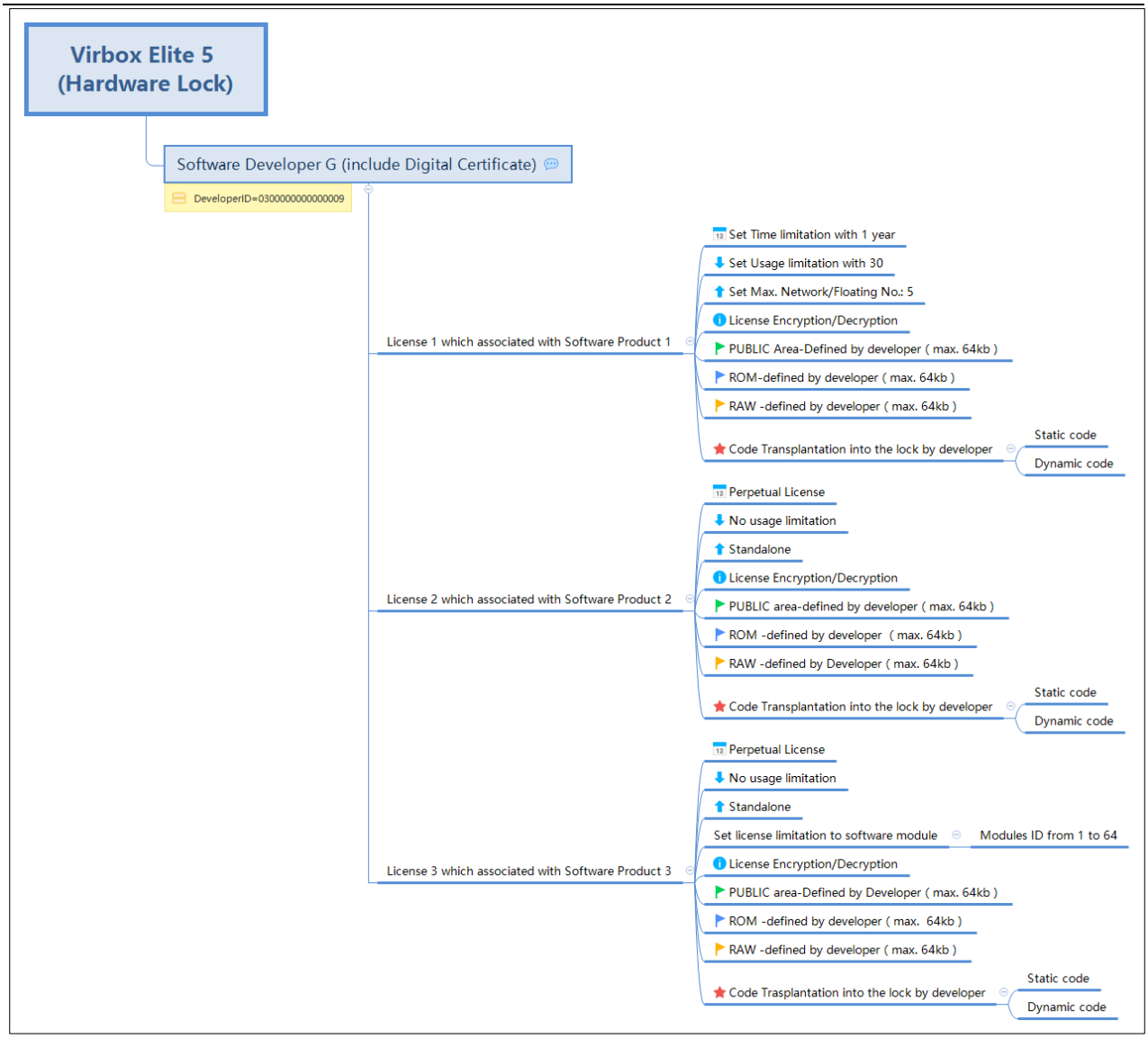
Virbox License Management refers to the management the license entitlement in whole license lifecycle which cover from Create the Product, Create the license associated, License distribution, License verification and license revoke, and also manage the software user.

- [Security] The Algorithm of License and the operation to License remote update are executed within the Virbox EL5 hardware lock, Private key and algorithm are well protected inside of EL5 lock and can't be cracked, the core parts of software, such as data, algorithm, after bound with license, must be access by call the relevant bound license and authorized then activate related functions or features, and all algorithms will be invalid if bound license expired. and hacker also can't be crack the whole software via crack related license or API
- [Easy to Use] Virbox Split the software protection/encryption and License into 2 process, software developer just need to well plan and define the related module , functionality or feature which bind with license in the product planning period and not necessary to plan and process license relate control logic. All License related process will be operated by Virbox LM later. So, it well support developer to issue license with more flexible mode and dramatically reduce workload.
- [Future Proof] Provide extremely large No. of License ID to each developer which up to 4 billion. (1--0xFFFFFFFF) License ID can be setup for software product. For Virbox Cloud Lock even support more No. of Cloud License ID be issued which enough for development plan in coming decades.
- [Upgrade] Easy for license upgrade in end user premise and no need for developer to develop the license upgrade tools. All data, program will smoothly upgrade and compatible.

The Idea of Virbox License Management:

- One times Software protection and Flexible to Issue multiple license separately.
- Secured License: Data (Module, feature) and Algorithm bind with License and executed within Virbox EL5 hardware lock, and License upgrade operation within hardware lock also.
- Future Proof Plan and License management: > 4 Billion License ID space and Role Assignment supported.
- Top Security: Build up Secured software and hardware environment with top performance, Algorithm bind with License directly.
- Easy to Use: Complete workflow and tool chain support; commercial API support; common interfaces for multi-type lock.
- Flexibility: Separation of Software Protection and License issue Process, flexible license mode and license type support and safeguard license upgrade.
- Scalability: Provides completely foundational interface platform supports secondary development.

Linkage between License and Software application show as below:



3.1 Lifecycle of License

3.1.1 Create the License

Only License 0 available when the hardware lock has been shipped from Virbox manufacture, the rest license will be created with issue the license update package and distribute this update package via remote update by software developer by use of Master lock.

License Update package classified with following packages: Create License, Update License, Delete License, Unlock the locked license and Delete all License.

3.1.2 License Update

Rule of License Update

Following rules are only valid and effective for license issued to hardware lock, for Rules to Cloud License and soft license, please refers to the online documentation of Virbox Developer Center (Virbox Cloud LM).

No.	Rules and	Description	Remark

	Definition		
1	License update can't be replayed	The update package cannot be used twice, and the update package that has been updated cannot be used again.	Each upgrade package has a time stamp and serial number for unique guarantee.
2	Update in sequence for multi update package	License update must follow the sequence if multi update package need to update	Each update package has an issue timestamp and a serial number guarantee, and the subsequent package issue time and serial number are incremental.
3	Create a new License (not mandatory)	Can only create license that do not exist in the lock	Prevent and avoid business errors
4	Create a new License (mandatory)	Create a license that does not exist in the lock, overwrite the original license if the license already exists in the lock	Effective optimization way to update licenses
5	Update the License	Can only update licenses that already exist in the lock	Prevent and avoid cheating license
6	Delete the license	Delete the existed license in the lock	For Business accuracy and strictly
7	Locked global Licenses	All licenses which issued by this developer in the hardware lock will be locked and not available	the protection way to software developer, if the software users who violate the license terms, only the developer unlocks the locked license
8	Unlock the locked global licenses	Unlock global license restrictions	remove the lock to global licenses

3.1.3 License Revocation

License Revocation refers the way to revoke the valid license during license validity period, which developer issue the "delete" license update package and execute with remote update.

3.2 Advanced Guide for License

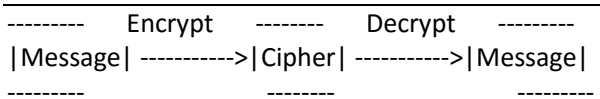
3.2.1 Encryption Algorithm

Virbox License Manager provides several encryption algorithm for licensing, besides of AES algorithm, Virbox also use DES, RSA, SHA~1 HASH, Developer-defined algorithms are available to developers. Among them, AES is the most commonly used software protection algorithm. RSA algorithm is often used for official version verification of software. Developers can also improve the software protection level by customizing a set of algorithms.

3.2.1.1 AES Algorithm

The AES algorithm is the latest NIST symmetry algorithm standard in the United States, providing the best security. Virbox LM integrate the AES algorithm inside, developers can complete the encryption and decryption through the API interfaces to encrypt and Decrypt.

The AES algorithm process is reversible, that is, encrypt a message and then Decrypt this encrypted Message, the result is the same as original Message, as shown as below:



Note: Encrypt and Decrypt calculation results are related to the license ID. Developers must first log in to the current license when using the algorithm. In other words, Encrypt and Decrypt are bound to a specific license ID. For different license IDs, the calculation results are different, which ensuring the security of different license encrypted data. In addition, the hardware lock (user dongle) provided by Virbox to each developer are different from each other, so the results obtained by the same license ID encryption of different developers are also different.

The Purpose for which algorithm binding with License ID:

1. The Algorithm implemented will up to 2^32 different versions when binding with License ID, it will achieve best protection and encryption result.

2. It will much convenient to license a software.

The AES algorithm is frequently used to encrypt data or variables in software to improve the security of software data.

3. Developers can use the encryption interface to encrypt key data (such as global variables, sensitive configuration data, account information, etc.) in the code during the program coding phase;
4. Replace the data recorded with plaintext in the original file with the ciphertext in the code file;
5. When the software needs to use the encrypted data, call the decryption interface to decrypt and use it.

Recommendation Developer may use symmetric algorithms to encrypt all or the key parts of the file which user generated. For example, when the file is opened and saved, the hacker cannot attack the key because the key is stored in the secure container (Located in EL5 dongle). The result is that there is no possibility that a pirated software can open the data files generated by the software, thus protect the value of the official released software. SLM encryption is directly encrypted and decrypted in the lock, and neither the algorithm nor the key can be obtained from the outside.

The key uses the hidden key located in the lock. The value of the key is calculated by the developer ID, the salt for encryption, and the bound license ID. There is no mechanism to export any copy of the key out of the hardware lock.

In General, it is recommended that the key for direct encryption/decryption algorithm in the lock uses the hidden key in the lock; The key for fast encryption and decryption used outside of the hardware lock is set by developer. If the key is not set by developer, the default value will be used as the key (The default value will be the result of direct decryption of when all default value is 0).

3.2.1.2 RSA Algorithm (#RSA)

RSA is an asymmetric algorithm, and RSA is suitable for the following scenarios:

With the feature of digital signature and signature verification based RSA algorithm, it guarantee that no "clone lock" will occur in any cases (even if the software is cracked) - a clone lock means that a clone lock can be 100% accepted by official released software. In some countries or regions, pirated software sales will bring huge profits, they (hacker) sell pirated software and ensure that the pirated software can be as close as possible to the original official release version, and protect their "labor results" (cracked software), they will use the clone lock to replace the official released hardware lock and sell the pirated software together.

The internal RSA algorithm of the SS LM module uses 2048 bits, which is the most reasonable key length for both performance and security.

The basic idea of the anti-cloning of the SS LM module is that the software verifies the validation of the hardware lock. It can be understood as the identification process. The workflow is as follows:

1. The Challenger of the software initiates a challenge to the hardware lock (dongle). The easiest way is to generate a random number and send it to the dongle, requesting a corresponding response;
2. The Responder of the dongle is digitally signed after the received random number with simply processed, it is performed by the secret private key Kb stored inside the dongle;
3. The Verifier in the software verifies the signature result of the dongle, and performs by the Ka paired with Kb stored in the software;

The Feature for RSA algorithm is the private key can't be calculated with the key length 2048 under the condition that the public key is fully disclosed, so no one can make a clone lock that can be accepted by the software;

Whether or not to implement this anti-cloning strategy in software depends on the piracy situation of the software, especially whether or not the cloning lock existed in the market.

The method of remote official release verification is similar to anti-cloning. The only difference is that the software that verifies the validation of the hardware lock is not the client software, but the networked server. The software company can provide the user with official released verification by developing a small network service program for official released version verification service.

3.2.2 Encryption to License Terms/Information

License encryption is often used to encrypt user-defined data and shell decryption code execution.

The key is isolated for the software application and is associated with the developer ID.

The key of a vendor's specified license ID is always the same, and the keys of the same license ID are different for different vendors.

The key is calculated and generated from inside of the hardware lock, which is related to the software developer and the license ID and the private value. The key located inside of the hardware lock cannot be obtained from the outside which ensure the security of the license encryption.

3.3 Managed Memory

The principle of Managed Memory is that the software uses a valid license as a credential to encrypt data and verify data in the SS security environment. The data in the managed memory has no plain text and cannot be modified illegally. Hackers are extremely difficult to view and tamper the data in the managed memory.

Developers can save sensitive data in the software to managed memory, such as account passwords, SQL database accounts and passwords, and temporary data related to operational privileges. The Managed Memory processing improves the coupling between software and the security environment (SS service), effectively preventing hackers from bypassing SS security services to debug or run client software.

The Advantage of Managed Memory:

- Sensitive data memory is not leaked and cannot be tampered with.
- Data can be exchanged securely across threads.
- Strong coupling of software, license and SS security services to improve software anti-cracked capabilities.

Managed Memory has the same function as the user data area, but there are still differences between of them. The memory is stored in the local SS, and the user data area is written in the License data area of the hardware lock. The difference between them is as follows:

Comparison	Managed Memory	User Data Area
The Way	SS Secured Memo environment	inside of Hardware lock
the Speed to Read and Write	Fast	Slow
The space	0—1G	0-65535 bytes
Life cycle	no	Depends on hardware lifetime
Isolation of APP access	Shared, Isolated	Shared only
Data Invalid	Data will be invalid when quit the APP	Permanent saved
Data Security	Cannot read or write without license, extremely difficult to tamper it	Cannot read/write and tamper it without license,
Data integrity	Almost up to 100% which guaranteed by Algorithm	Depends on hardware

Recommendation:

- For the temporary, large amounts of data, frequent read and write, and runtime sensitive data, save to be managed memory.
- For the data which the results of the business operation data that needs to be acquired at the next startup, save to the user data area.

4 License Enforcement

4.1 Fast License Enforcement

Virbox Protector is an automatic protection (encryption) Tool with high security level and developed by Virbox with integration of the experience in software protection and encryption more than 20 years.

Virbox Protector works with Virbox License Services, Cloud Lock, Elite 5 Hardware Lock or Soft Lock, which integrate latest of technology includes Code snippet(Code fragmentation), obfuscation, virtual machine environment, and data encryption.

Virbox Protector is the industry's leading software protection tool for high protection without programming competence needed.

Virbox Protector will support the software developer to quickly protect the software program. With Virbox Protector, it is not necessary to add any modules or features for protection in software development process.

When Software development completed, developer may use the Virbox Protector to protect your software with one time protection: Select the type of hardware lock (local lock, Remote: network lock), cloud lock or soft lock during protection. Or developer may combine any type of locks. Then software can be protected and licensed by use of Virbox LM solution directly.

For more detail information to Virbox Protector, please refer to: "User Manual: Virbox Protector"

4.2 Flexible License Enforcement

The Virbox SDK provides a complete package of APIs for license management and licensing that will help software developer get the most out of the Virbox licensing system. Developers can flexibly implement their own encryption scheme by calling the APIs, and combine the functions of encryption, decryption and data area provided by Virbox license to realize deeply protection scheme and further increase the difficulty of cracking.

Following API interface provided by Virbox SDK

4.2.1 slm_runtime

The API interface to access Virbox license, referred to as RuntimeAPI, is the most critical interface for the Virbox licensing system. Developer software provides licensed access via this interface, which is the interface directly used by the developer software.

File Presents in the SDK

FileName	Function	Description
ss_lm_runtime.h	Headfile of RuntimeAPI	Description of each interface functions.
slm_runtime_api.lib	The Static Libs of RuntimeAPI	Includes Anti Debugging functions, to import this lib for developer's software official launched;
slm_runtime_api_dev.lib	the Developing static lib of RuntimeAPI for developing	Excluded anti debugging functions, developer may import this lib during developing process;
slm_runtime.dll	DLL Lib of RuntimeAPI	Includes anti debugging functions which import this lib for developer's software official launch;
slm_runtime_dev.dll	the Developing DLL Libs of RuntimeAPI for developing	Excludes Anti-Debugging functions which imported this lib by developer during developing process;
slm_runtime_easy.dll	DLL libs of RuntimeAPI	The easier interface to be called by other program language;
slm_runtime_easy_dev.dll	The Developing DLL of RuntimeAPI for developing	Same functions above
libruntime.lib	The Static libs of RuntimeAPI	Use the loading memory, the software doesn't need to use VMP to protect by use of this Static lib
libruntime_dev.lib	The developing static lib of RuntimeAPI for developing	Use the loading memory, which will be called for testing in developing process. same above

** **Note**:

- Developing Lib (with suffix: _dev), the libs excludes anti-debug functions, is the libs only used for developer in developing process which is convenient to debug and test the software during developing process and not be used for official software launch. otherwise anti-debug function of the launched software will be disable and be easier for hacker to crack the protected software;
- libruntime.lib is the static libs which use loading memory, the functions is same as the functions of basic static libs used, the way of loading memory is continue to use VMP to protect software application when excluded static libs called by software program

Here is the show case to call runtime API by use of C language:

```
#include "stdio.h"
#include "ss_lm_runtime.h"

int main(int argc, char **argv)
{
    SLM_HANDLE_INDEX hslm = 0;
    BYTE original_data[TEST_ENCRYPT_DATA_SIZE ] = {"test_data1234567890"};
    BYTE encrypted_data[TEST_ENCRYPT_DATA_SIZE] = {0};
    BYTE decrypted_data[TEST_ENCRYPT_DATA_SIZE] = {0};

    SS_UINT32 sts = SS_OK;

    SS_BYTE psd[16] = { 0xDB, 0x3B, 0x83, 0x8B, 0x2E, 0x4F, 0x08, 0xF5, 0xC9, 0xEF, 0xCD, 0x1A, 0x5D, 0xD1, 0x63,
0x41 };      // API Password of developer， every developer has own a unique API password, the API password must
 be obtain from Virbox Developer Center only.
    ST_LOGIN_PARAM login_struct = {0};
    ST_INIT_PARAM st_init_param = {0};

    //0. Global initilized function, call this function to initial the slm_runtime
    st_init_param.version = SLM_CALLBACK_VERSION02;
    st_init_param.pfn = & (app_ss_msg_core);
    memcpy(st_init_param.password, psd, sizeof(psd));

    sts = slm_init(&(st_init_param));

    //1. slm_login login license
    login_struct.license_id = 0;   //login the License 0
    login_struct.size = sizeof(ST_LOGIN_PARAM);
    login_struct.timeout = 86400;

    sts = slm_login(login_struct,STRUCT, &(hslm),NULL);//
    if (SS_OK == sts)
    {
        printf("slm_login ok,login license successfully！\n");
    }
    else
    {
        printf("slm_login faield error = 0x%08X, lgoin license failure：%s\n",sts, slm_error_msg(sts, 1));
        goto end;
    }
    // 2. Use the encryption to license to encrypt the testing data
    printf("before encrypt: %s\n", original_data);
    sts = slm_encrypt(hslm, original_data, encrypted_data, TEST_ENCRYPT_DATA_SIZE);
    if (SS_OK == sts)
    {
        printf("slm_encrypt ok！Encryption to license successful \n");
        //hex printf encrypted_data，It can be printed in hexadecimal to check that the encrypted data is unreadabl
e.
    }
    else
    {
        printf("slm_encrypt faield error = 0x%08X, Encryption failure: %s\n",sts, slm_error_msg(sts, 1));
        goto end;
    }
    // 3. Use decryption to license to decrypt the encrypted data
    sts = slm_decrypt(hslm, encrypted_data, decrypted_data, TEST_ENCRYPT_DATA_SIZE);
    if (SS_OK == sts)
    {
        printf("slm_decrypt ok！Decryption the license successful\n");
        // Comparision if the orignial data (original_data) is consistent with the decrypted data(decrypted_data)
    }
    else
    {
        printf("slm_decrypt faield error = 0x%08X, Decryption failure: %s\n",sts, slm_error_msg(sts, 1));
```

```
        goto end;
    }

end:
    slm_logout(hslm);
    return sts;
}
```

4.2.2 slm_control

This API is the management interface of Virbox License Manager, here referred to as ControlAPI. Control API is mainly used and provides the inquiry to the hardware lock (dongle)'s information, inquiry to license terms and status, inquiry to license session etc.

FileName	Functions	Description
ss_lm_control.h	Headfile of ControlAPI	Description to each Interface functions
slm_control_api.lib	The static libs of ControlAPI	the called related libs for the implementation of Control API
slm_control.dll	The DLL libs of ControlAPI	the called related libs for the implementation of Control API

Here is the show case to call Control API by use of C language:

```
// This show case presents how to obtain the license terms/information located inside of hardward lock (dongle)

#include <stdio.h>
#include "json/json.h"       // jsoncpp has been used in this show case to parse the json structure.
#include "ss_error.h"
#include "ss_lm_control.h"

int main(int argc, char **argv)
{
    SS_UINT32 buf_size = 0;
    SS_UINT32 ret = 0;
    void *ipc = NULL;
    char *dev_desc = NULL;

    Json::Reader reader;
    Json::Value  root;
    Json::Value  desc;

    ret = slm_client_open(&ipc);
    if (ret != SS_OK)
    {
        return -1;
    }
    printf("slm_client_open ok\n");
    ret = slm_get_all_description(ipc, JSON, &dev_desc);
    if (ret != SS_OK)
    {
        printf("slm_get_all_description failed,errorcode=0x%08x",ret);
    }
    else
    {
        printf("slm_get_all_description ok\n");
```

```
        printf("begin prase all lock\n");
        if (!reader.parse(dev_desc, root))
        {
            printf("parse error \n");
        }
        else
        {
            printf("parse ok \n");
        }
    }
    std::string dev_type;
    std::string host_name;
    std::string ip;
    std::string sn;
    std::string devp_id;
    std::string print_str;

    int dev_num = root.size();              // the qty. of device
    for (int i = 0; i < dev_num; ++i)
    {
        desc = root[i];
        if(desc != NULL)
        {
            char *lic_context = NULL;
            print_str = "\nbegin to read license \n";

            if (!desc["host_name"].isNull())
            {
                host_name = desc["host_name"].asString();
                print_str += "[host_name:" + host_name + "] ";
            }
            if (!desc["ip"].isNull())
            {
                ip = desc["ip"].asString();
                print_str += "[ip:" + ip + "] ";
            }
            if (!desc["type"].isNull())
            {
                dev_type = desc["type"].asString();
                print_str += "[type:" + dev_type + "] ";
            }
            if (!desc["sn"].isNull())
            {
                sn = desc["sn"].asString();
                print_str += "[sn:" + sn + "] ";
            }
            if (!desc["developer_id"].isNull())
            {
                devp_id = desc["developer_id"].asString();
                print_str += "[devp_id:" + devp_id + "] ";
            }
            print_str += "\n";
            printf(print_str.c_str());

            ret = slm_read_brief_license_context(ipc, JSON, desc.toStyledString().c_str(), &lic_context);
            if(ret == SS_OK && lic_context != NULL)
            {
                printf(lic_context);
                printf("\n");
                slm_free(lic_context);
                lic_context = NULL;

                printf("slm_read_brief_license_context ok\n");
            }
```

```
                else
                {
                    printf("slm_read_brief_license_context failed, [sn=%s] \n[errorcode=0x%08x]", sn.c_str(), ret);
                }
            }
        }

        slm_client_close(ipc);

        return 0;
}
```

4.2.3 D2C API

Virbox SDK provides "Developer Utility", the local License Manager, for developer to issue the license to hardware lock (Virbox El5) quickly. With "Developer Utility", it is not necessary for developer to build up the "In house License system" with large Capex investment, Virbox Utility is the light license toolkit to issue the license with API interface to issue the license to hardware lock. D2C API provides the interface to developer which used to issue license to hardware lock. Developer may call the "D2C API" to issue license with more flexible operation and flexible to combine different license terms.

For Cloud license issuing, Developer can issue the cloud license via Virbox Developer Center directly.

the D2C package created by d2cAPI is the only license update package which can be recognized and identified by Virbox EL5 hardware lock, d2c package includes digital signature and encryption to the license terms which will guarantee the integrity and security of license data, on the other hands, the Virbox EL5 hardware lock provides the function to anti-replay, all d2c license update package will be used one time in the Virbox El5 hardware lock only.

File Presents in the SDK

FileName	Functions	Description
d2c.h	Head file of d2cAPI	Description to each Interface functions
libd2c.dll	the DLL libs of d2cAPI	The called related DLL libs for the implementation of d2cAPI

Note: Calling d2c API is quite complicate process, more details, please refer to sample file.

4.2.4 Update API

This Interface provides the functions to update d2c package, developer may call d2cAPI to create d2c License update package and upgrade this d2c update package into the hardware lock by call "slm_update" and increase the flexibility to issue and update the license in the hardware lock.

Implementation of slm_update interface, this interface can be implemented in both RuntimeAPI and ControlAPI;

Software update will be implemented by calling this interface by use of any of libs from runtime API and control API.

More details for how to use this interface, please refer to the sample file.

5 Appendix

Glossary

Virbox Elite 5 Acme	Virbox El5 Acme is latest Generation of Virbox hardware Lock which improved both in Hardware performance and Security Performance. Virbox El5 Acme is key component and License repository of Virbox LM solution. Virbox El5 Acme can be used to be a standalone dongle without connect to Virbox LM, and also can be used seamless integrated with Virbox LM with additional and advance feature available. Virbox El5 Acme support and suitable to deploy in tough industry environment. Software Developer can use this product to protect the Software copyright and IP. It will supports developer to protect software with most security technology, best performance and easy to use. For detail Specification of Virbox El5 Acme, please refer to the Virbox EL5 Acme datasheet.
Virbox LM	Virbox LM is Cloud based Software protection, License Entitlement Solution and cloud service which provides to software developer with lowest Total cost ownership, Top Security of Software Copyright Protection and Easy to use in whole software life cycle. By use of Virbox LM, Developers can achieve extremely protection to software copyright/IP and flexible to issue license to software user with minimal effort and investment. Virbox LM One Stop Solution consist of following components: Virbox license Manager (Virbox Developer Center), Software Protection (Envelope) Tools (Virbox Protector), Local License Manger (Virbox Developer Utility) and Virbox License Service and User License Tool (Virbox User License Tool) and other support tools and documents. Virbox License Type supports: hardware lock (Elite 5 Acme Dongle), Soft Lock (Soft license) & Cloud Lock (Cloud license) and suitable to deploy in different Software implementation scenario, on line, offline, firewall isolated environment etc. Virbox provides flexible license mode to support ISV/software developer Software monetization which includes: Perpetual, trial, time based, feature on demand, subscription, concurrent; etc.
Virbox Developer Center	Virbox Developer Center consists of Developer website and Open API, It provides cloud based Service and website for License Issue and License Entitlement solution to software developer, Developer may use [Virbox Developer Center] (https://developer.lm-global.virbox.com/) to manage software product, user management, License Issue and Entitlement, Virbox Developer Center is online License solution for software developer, By use of Virbox Developer Center, it is not necessary for developer to build up in house

	License entitlement system and save the investment both in Capex and Opex.
Virbox Protector	Virbox Protector is the latest envelope tools for software protection and encryption, Virbox Protector integrated multi security technology includes Virtual Machine, Code Snippet (Code Fragmentation) and other static and dynamic technology/Service and provides top secure protection level to developer's software, Developer may protect software automatically by select protection options without any coding competence and save lot of workload and effort in software protection process.
Virbox Developer Utility	Virbox Developer Utility is local License Manager provided by Virbox LM, this Utility helps developer to issue the license to hardware lock (Virbox EL5 Acme) quickly. With Virbox Developer Utility, Developer will save Capex and Opex and not necessary to spend money to build up the In House License System.
Virbox User License Tool	A software installed in software user site that can be used for license verification, update and transfer service, Virbox User License Tools works with Virbox License service and provides static and dynamic protection service to software installed in software user site.
Product	Virbox Use "Product" refer to the Software application developed by software developer, software publisher which need to encrypt, protect the "product" by use of Virbox Protector.
Sales Template	The Sales mode for mature commercial software should support different sale mode and charging policy to meet the requirement of software users: pay per year/month/day, per usage, or pay for perpetual license. Above charge policy is independence with Product functions but depends on sales strategy to each market segment. So, Virbox designed the following "Concept" in Virbox License system to reduce duplicate workload for developer in protection and license process: *Sale Template*, developer may setup the different limitation to same product in different Sales Template to meet the different sales and marketing strategy.
License ID	License ID is the unique indicator which used in the software protection and license issuing/distribution process in the Virbox LM system, which presents the License No. for software. For details introduction to Virbox License Management and License concept, please refer to "Virbox License System" Developer must assign associate License ID to the product created. And the License ID must not be duplicated assigned and used for other product. The License ID associated with specific product will be use unique ID No. and Indicator to presents for this product in all protection and license process. The protected software will verify the associated License when software user execute the software.
Account	Account sign up by "user" in Virbox developer Center or Virbox User Center, For software developer, they can sign up account in Virbox developer center to evaluate and test the Virbox Solution for software protection and licensing to their product; For Software user, they can sign up an account or assign a account in Virbox

	User center by software developer, this user account will be the unique identifier for user to use and verify license issued by developer;
Master Lock	Virbox Solution provides 2 kinds of hardware lock for software developer: **Master Lock** and **User lock**; master lock is the unique Identity for each Software developer to protect software and issue license, it is mainly used for software encryption , protection, issue license and issue upgrade package to EL5 Acme user lock.
User Lock	User lock is the hardware based lock deliver to software user which contains the license file issued by Software developer, Virbox Provides following Virbox EL5 user lock to developer: EL5 Acme, EL5, EL5 RRTC, EL5 Genii.
Soft Lock	Soft Lock contains software license, saved in local safety environment. Which is used to verify the protected software. Software developer may issue the Soft license from Virbox Developer Center to licensed software user account, Soft lock supports both online mode and offline mode.
Cloud Lock	The cloud license (cloud lock) is one of license type supported by Virbox LM, , Cloud License issued by developer from Virbox Developer Center and stored in Virbox Cloud and used to verify the protected software. The cloud license must be bound with the software user account. Users can use the protected software in online status (Access internet) and successfully logged in Virbox User Account.
API	Virbox SDK provides a complete package of APIs for license management and licensing that will help software developer get the most out of the Virbox licensing system. Developers can flexibly implement their own encryption scheme by calling these APIs, and combine the functions of encryption, decryption and data area provided by Virbox license to realize deeply protection scheme and further increase the difficulty of cracking.
Control API	This API is the management interface of Virbox License Manager, here referred to as Control API. Control API will mainly use and provides the inquiry to the hardware lock (dongle)'s information, inquiry to license terms and status, inquiry to license session etc. More details to how to use Control API and use case, please refer to Virbox License Architecture.
Runtime API	The API interface to access Virbox license, referred to as Runtime API, is the most critical interface for the Virbox licensing system. Runtime API is the key API interface for developer to use and call API, the software of Developer will use this API to access license. More details to how to use runtime API and use case, please refer to Virbox License Architecture.
d2c API	Virbox SDK provides "Developer Utility", the local License Manager, for developer to issue the license to hardware lock (Virbox El5) quickly. d2c API provides the interface to developer to issue license to hardware lock. Developer may call the "D2C API" to issue license with more flexible operation and flexible to combine different license terms. the d2c package created by d2cAPI is the only license update package which

	can be recognized and identified by Virbox EL5 hardware lock, d2c package includes digital signature and encryption to the license terms which will guarantee the integrity and security of license data, on the other hands, the Virbox EL5 hardware lock provides the function to anti-replay, all d2c license update package will be used one time in the hardware lock only. Note: Calling d2c API is quite complicate process, more details, please refer to sample file.
Open API	Open API is the API that Virbox LM provides to developers to access the Virbox Developer Center. Open API can be used for integration with existed third party system in developer site, such as CRM or ERP system. Developers may use and call this OPEN API for user sign up, license issue, license distribution and other operations via third party system. For details, please refer to [**Open API Guide** V2.0**/Virbox Developer Center]**
API Assistant	API Assistant is a quick starter tool that helps software developers learn and understand the Virbox API. Execute the API via friendly GUI prompts message to view the API execution result. You can quickly familiarize with API and functions without coding. API Assistant is suitable for developers to quickly learn and understand the usage/calling of the API and verify the feasibility of the encryption scheme.
Code Fragmentation Execution	"Code Snippet Execution" or we can named with more simple and easy understanding: "Code Fragmentation Execution". Code Snippet Execution will utilized latest and mature technology in "Extracting the coding" and select and extract large amount of "coding" from original program code, and executed in the security virtual environment after encryption and obfuscation and integrated with Virbox Encryption Compiling Engine, Virtualization and driving technology, all technology integrated together makes protected software in more security during execution process. The Code Snippet Execution" can be simply understand as breaking down the functions or modules of software into code of fragments and execute these "fragments" into a secure environment, so that the crackers can't start to use hacker tools to analysis and get the original code and execution result. "Code Snippet Execution" is major milestone and breakthrough in the Software Encryption and Protection field. In addition to high Security, Code Snippet also rely on and work with "Virbox License Service“ middleware to shield the interface from the fundamental layer and also compatible with Virbox Cloud Lock and Soft Lock, and help developer to realized business agility with different kind of license type.
Anti-Hacker Service	Anti-Hacker Service, usually, software will be protected in static way by use of traditional protection technology: obfuscation, Virtualization, it will not be update after software delivered to customer , Virbox use "Anti Hacker Service" and associated tools installed in customer side which provides proactive detect hacker behavior when software execution, with Anti-Hook, Anti-Reverse engineering, Anti-Debug, update hacker database functionalities

	and features, it comprise a series anti hacker technology to protect your software with dynamic and proactive way.

Let's Start Your Software Protection and Licensing Journey!

Virbox will help you to protect your software copyright/IP Value and Create Revenue for you.