White Paper

Network Licensing WP-2

Subject: Network Licensing with the CRYPTO-BOX
Version: Smarx OS PPK 8.21 and higher
Last Update: 24 February 2025
Target Operating Systems: Windows, Linux, macOS, iOS, Android
Target Processor Platforms: amd64/i386/aarch64/armhf
Supported Programming Tools: see chapter 5.3.6 in this document
Applicable for Product: CRYPTO-BOX® SC / XS / Versa

Network Licensing with the CRYPTO-BOX®

Network licensing is ideal for cost-effective software licensing in (corporate) networks. The software vendor determines how often the application is allowed to run in a network - with just one CRYPTO-BOX per network.

Furthermore, it allows software licensing not only for PCs and laptops, but also for environments without the possibility to connect a dongle directly:

- Mobile devices (Smartphones, Tablets)
- IoT devices
- Virtual machines (Windows/Citrix Terminal Server)

This document provides an overview about network licensing and the client and server side components.

Quick and efficient hardware based software protection!

Software and information piracy costs billions of dollars in annual losses to software vendors, distributors and content providers worldwide. The internets role in software and data distribution is growing rapidly and increases the importance of the situation dramatically. Hardware based protection can be used for creating robust and reliable secure demo versions of applications in a straightforward manner. While benefiting from strong and effective application protection provided by a hardware based approach you can create flexible and secure demo versions with ease. The CRYPTO-BOX makes it happen!

CRYPTO-BOX®

- Quick and easy protection of Windows applications with AutoCrypt
- Individual implementations with API for all common programming languages and platforms (Windows, Linux, macOS)
- Unique and stable metal case, also available with USB-C plug and in different colors and customized product labeling.
- Network and remote update capability
- EAL 4+ certified Smart Card chip with AES/Rijndael encryption implemented on-chip.
- RSA 2048 bit support on-chip (CRYPTO-BOX SC) or on driver level (CRYPTO-BOX XS).
- The CRYPTO-BOX® is designed in Germany and manufactured in the European Union (TAA compliant).
- Customization: Implementation of specific functions or algorithms in firmware on request.

Order your CRYPTO-BOX Evaluation Kit now – or contact us for any questions:

MARX Software Security GmbH
Vohburger Strasse 68
85104 Wackerstein, Germany
Phone: +49 (0) 8403 / 9295-0
contact-de@marx.com

MARX CryptoTech LP
489 South Hill Street
Buford, GA 30518 U.S.A.
Phone: (+1) 770 904 0369
contact@marx.com

www.marx.com

Download the latest Application Notes: www.marx.com/en/support/documents

Table of Contents

1. Network Licensing – An Overview

1.1. Introduction

Hardware-based protection with the CRYPTO-BOX requires that the protected applications have a corresponding CRYPTO-BOX attached to the computer in order to function properly. If the CRYPTO-BOX is attached, the program will communicate with it, performing more detailed verification of information stored in the device, such as:

- Verification of the CRYPTO-BOX Serialnumber (BoxName) or Developer ID
- Using the hardware-based encryption engine to decrypt information during application runtime
- Using the internal memory of the CRYPTO-BOX to store licensing information of the application

All these, as well as many other unique CRYPTO-BOX features, can be used to build a very reliable protection strategy.
However, in some cases a CRYPTO-BOX attached to the local computer is not desired or possible, for example:

- At places where the CRYPTO-BOX can get lost or stolen.
- Where a centralized license management is required.
- For mobile (Tablets, Smartphones), IoT or industrial devices with no USB connector.
- For virtual machines (eg. Windows or Citrix Terminal Server solution) offering no USB support.

1.2. Advantages of Network Licensing

Network licensing provides the following advantages over the local connection of the CRYPTO-BOX to the USB port of the computer:

- Complete control over licenses, respectively running instances of the protected application, in the entire network.
- Protection and license management for environments where a local USB port is not available/accessible like restricted PCs, Smartphones, Tablets (iOS and Android).
- No administrative access required for both installation and during runtime of the protected application (access to a locally connected CRYPTO-BOX requires administrative rights at least for installation of the CRYPTO-BOX drivers)
- Suitable for licensing in server-based computing environments (Microsoft, VMWare, Citrix, etc.).
- Cost-efficiency: multiple license counters can be defined to protect several applications with one CRYPTO-BOX.
- Integrated tool to monitor and check the status of connected clients.

1.3. License Control System (LCS) for Configuring the Maximum Number of Seats

1.3.1. Overview

The License Control System (LCS) is available for all CRYPTO-BOX models (except CRYPTO-BOX Versa). It allows to configure the CRYPTO-BOX with a seat limit between 0 and 254. The value 255 is reserved for unlimited use.

LCS is available as an option. Please refer to marx.com → Shop → Solutions → LCS License Control System for further details and pricing.

The network license counter is a data object which is stored in a special area of the CRYPTO-BOX memory. It can hold network license information for multiple partitions (different applications). That means you can define independent network license counters for each partition (application) in the CRYPTO-BOX.

For more details on CRYPTO-BOX partitions, we strongly recommend to read chapter 10.2 in the Smarx Compendium. It's important to understand this concept!

When a program tries to login to a proper partition, first Smarx OS will search for a LCS record for this partition (see chapter 5.3.2 for more details). If the record is found, LCS checks the number of permitted licenses and the number of seats currently logged in. If there are free licenses available, access to the partition is granted, otherwise it is denied. For a CRYPTO-BOX Versa, the number of permitted network seats is always unlimited because LCS is not supported for this model.

The Remote Update Management System (RUMS) allows the programming/update of the license counter directly by the end user, providing subsequent transactions through the sale of additional licenses for the software distributor.

More details on RUMS can be found in the RUMS Application Notes.

Depending on your implementation scenario, there are different ways of programming the network license counter.

1.3.2. AutoCrypt

If you use automatic protection with AutoCrypt to protect your application, see chapter 5.2 for more details on setting the license counter.

1.3.3. Implementation with API

For Implementation with API, the license counter can be set using either:
- The Smarx OS Application Framework, see “Smarx OS Compendium”, chapter 4.5
- The command line tool 'SmrxProg'
- API commands, see chapter 5.3

2. Smarx®OS Network Server - Introduction

Smarx OS Networking allows protected applications to access a CRYPTO-BOX attached to the USB port of any computer in a network.

A special program called **Smarx OS Network Server** (or CBIOS Network Server), running on a computer in the network, manages remote connections to the CRYPTO-BOX attached to this computer. Any Smarx OS based application trying to access a CRYPTO-BOX will be recognized as a **Smarx OS Network Client** (see chapter *5)*.

The Smarx OS Network Server is available for Windows (64/32Bit), Linux (amd64/i386/aarch64/armhf) and macOS platforms. See item "Drivers & Tools" → "Network Server" in the Protection Kit Control Center for more details or the /network server/ subfolder in the SmarxOS4Linux/SmarxOS4Mac package.

3. Server Installation

Check our website marx.com → Support → Downloads → Network Utilities to get the latest version of the CRYPTO-BOX Network Server for your platform.

3.1. Windows

For the Windows platform, the server is provided as Windows Installer setup (.msi) and Windows Installer Merge Module (.msm) for 64 or 32 bit. See item "Drivers & Tools" → "Network Server" in the PPK Control Center for more information.

The Windows Installer setup (.msi) also installs the CRYPTO-BOX device driver during installation, so it is not required to install the drivers separately with CBUSetup.exe.

During installation, you can select if you want to start the server as a service. This can be changed later if required, see chapter *4* for details.

3.2. Linux

For Linux platforms, the server is provided as .deb and .rpm packages for amd64/i386/aarch64/armhf platforms.

To install the corresponding package for your platform:
$ sudo dpkg -i netcbios-1.6.10-amd64.deb (for Ubuntu OS)
or
$ sudo yum install netcbios-1.6.10.x86_64.rpm (for Fedora OS)

or double click on .deb (.rpm) file in file manager
and click "Install Package"

4. Server Administration

There are several possibilities to administrate the Smarx(R)OS Network Server:

a) Locally on the same PC where the server is installed. See chapter 4.2 for further details on Windows platforms, or chapter 4.3 for details on Linux platforms.
b) The server can be managed remotely with the Administrative Utility (Windows application), via TCP/IP protocol. The Administrative Utility (AdminApp.exe) is part of the CBIOS Server installation package for Windows (see chapter 3)
c) Alternatively the server can be administrated directly via API commands. A sample can be found in the Smarx OS Protection Kit (PPK) for Windows:
<SmarxOS PPK root>\SmarxOS\Network\Win\Samples\netadmin for details.

4.1. Change Server Configuration Settings (CBIOSSrv.cfg)

There are two ways to change the settings of the Smarx OS Network Server:

- Editing the CBIOSSrv.cfg (resp. CBIOSSrv64.cfg) file located in the server directory (Windows) or in /etc folder (Linux).
- The following settings can be changed using the Administrative Console (see chapter 4.2.2), which allows you to manage the server remotely using the TCP/IP protocol (AdminApp requires a Windows computer):
  - Server administration password (AdminPassword=, see chapter 4.1.3)
  - Connection timeout (ConnectionTimeoutSec, see chapter 4.1.4)
  - Inactivity timeout (KeepAliveTimeoutSec, see chapter 4.1.4)
  - Inactivity scan rate (KeepAliveScanRateMSec, see chapter 4.1.4)

If options are not set in .cfg file, then default values will be used. Some options described below may not be available under Linux and macOS!

4.1.1. Server IP & Port

IP=* // default value
Port=8765 // default value

By default, the Server will listen on all network adapters available on the Server computer. If having more than one active network configured for the Server computer (Ethernet, Wi-Fi, active VPN, virtual adapters of VMWare, etc.) it may be required to assign a dedicated IP address (& optionally port) to listen/response to UDP broadcasting of CBIOS network clients.
It can be done with IP (and port) parameter of the CBIOSSrv.cfg configuration file.

Examples:
IP=10.10.10.32 (IPv4) or fe80::1ff:fe23:4567:890a (IPv6)
Port=1234

Important Hints:

- **a)** If IP=* is set (default), CBIOS Server will listen on all network interfaces available on this computer (IPv4 and IPv6). If you specify an IP address here (IPv4 or IPv6), CBIOS Server will be available only via this particular address.
- **b)** If IP=localhost is set, then the server will be available on the local computer only. This is useful when server and client application are running on the same computer and server should not be exposed to other computers in the network.
  REMARK: *IP=localhost is currently not supported for the Linux versions of the server!*

4.1.2. Server UDP Port (broadcasting)

UDPPort=8766 // default value

Port used by the server to listen/response to UDP broadcasting of CBIOS network clients when doing CBIOS_ScanNetwork on the client side (see chapter 5 for details).

4.1.3. Server Passwords and support for encryption functions without UPW/APW login

Administrative Password:
AdminPassword=admin // default value

Optional Client Password, required for encryption calls not requiring UPW/APW login:
OptionalClientPassword= // ignored (default value)

Allow/disable support for encryption calls not requiring UPW/APW login:
CryptNoLogin=1 // enable support for hardware based encryption calls not requiring UPW/APW login
CryptNoLogin=0 // disable support for hardware based encryption calls not requiring UPW/APW login (default value)

The Administrative Password is used for server administration (AdminApp or customer specific program).

Starting with version 2.28, the Server supports hardware based encryption calls not requiring UPW/APW Login in local mode (CryptFixed, InternalRSA, all CBU SC specific calls) before UPW/APW Logon.

The OptionalClientPassword property is supposed to prevent potential DDoS attacks if the Server is exposed

online (accessible via global network). In this case specifying value for OptionalClientPassword in the CBIIOSSRv.cfg configuration file will require clients to start with the CBIOS_ClientLogin (OptionalClientPassword) call for their requests on hardware based encryption to be processed by the Server prior to UPW/APW login. Please note that this feature will require updated libraries (Smarx OS PPK 8.21 or later is required!)

Overview on possible combinations of OptionalClientPassword and CryptNoLogin and their results:

Option1: OptionalClientPassword not set* CryptNoLogin=1	**Option2:** OptionalClientPassword set** CryptNoLogin=1	**Option3:** OptionalClientPassword set CryptNoLogin=0***	**Option4:** OptionalClientPassword not set* CryptNoLogin=0***
Encryption calls without UPW/APW will be always processed	CBIOS_ClientLogin call is required to process encryption calls without UPW/APW	UPW/APW is always required to process encryption calls	UPW/APW is always required to process encryption calls

* OptionalClientPassword not set is default value (CBIOSSrv64.cfg)
  Clients' requests for hardware based encryption coming prior to UPW/APW login will be processed by the Server

** Assumes that OptionalClientPassword property was set in CBIOSSrv64.cfg
  CBIOS will return "Box not logged" error in case of wrong OptionalClientPassword value was submitted at CBIOS_ClientLogin call

*** CryptNoLogin=0 is default value (CBIOSSrv64.cfg)
  Error message „CryptNoLogin not set“ is displayed for encryption calls without UPW/APW submission

4.1.4. Connection Settings

ConnectionTimeoutSec=30 // default value in seconds

KeepAliveScanRateMSec=3000 // default value in milliseconds

KeepAliveTimeoutSec=180 // default value in seconds

ClientKeepAliveDelayMSec=3000 // default value in milliseconds

ConnectionTimeout: Timeout for an inactive socket-level connection in situations when a connection (socket) to the server is opened but client does not send any data. The connection will be closed on reaching the timeout. After that, the server will be ready to process subsequent requests.
At the client side, a Connection Timeout message (transport layer error of the last CBIOS API call) will be received. The client has to repeat the last call to resume work. Default setting is 30 seconds.

KeepAliveTimeout: This is the session-level inactivity timeout. If the client does not send a “keep alive” packet for the previously opened session before this timeout is reached, the session will be closed and all appropriate resources (network licenses, etc.) will be released.
On the client side, it means that there is still a connection to the server possible, but the current encrypted session with the server is disconnected. So the client should call CBIOS_OpenBy... again.
In case of connection problems (client receives CBIOS_ERR_CONN_REFUSED), the client application can do the following:

- Repeat the last API call (maybe twice).

- If this does not help: Try to reopen the CRYPTO-BOX with CBIOS_OpenBy... call.
- If that also fails: Close the session completely with CBIOS_Close() and start from the beginning with CBIOS_Connect().

Also refer to chapter *5.3.5* for more information on how to handle typical situations such as connection breaks.
Default setting is 180 seconds.

KeepAliveScanRate: This is the rate at which the session table is scanned for inactive sessions (those sessions for which the “keep alive” packet is not received in time). Default setting is 3 seconds (3000 milliseconds in CBIOSSrv.cfg).

4.1.5. Debug Settings

"DebugLevel" parameter in CBIOSSrv.cfg file controls the level of the debug information that will be added to log file (see 4.1.6).

It is specified as DebugLevel=<n> where
Level 0 - quiet, nothing is written to log file
Level 1 - critical errors
Level 2 - warnings
Level 3 - general information (default)
Level 4 - debug information

4.1.6. Logfile Name and location

"LogFileName=" parameter allows to set logfile name and location.

Default values are:
Name:
CBIOSSrv.log (resp. CBIOSSrv64.log)

Location:
\Program Data\MARX\Network Server

If the "LogFileName=" parameter was not set in .cfg before first start of the server, CBIOS Server GUI will show only <skipped>, and no further log messages. In that case please restart the server to show the log messages.

4.1.7. Max Log Size Settings

MaxLogSizeKB=200 (default value). It controls the maximum log file size, but leaves the full session server. The old log is renamed to <...>.bak.

4.1.8. SetUPW/SetAPW Options

Allows to execute CBIOS_SetUPW/CBIOS_SetAPW via network (see CBIOS API reference for more details).

SetUPW=1 - allow CBIOS_SetUPW requests
SetUPW=0 - do not allow CBIOS_SetUPW requests (default value)

4.1.9. Setting Packet Limits - Protection against Denial-of Service (DoS) attacks

Allows to limit the number of requests per client (IP address) to the server during a defined time interval to prevent server overload, for instance by an attack or malfunctioning application.

FilterPacketsLimit=200 // Number of requests that are allowed to receive during FilterPacketsInterval, 0 - disable protection (default value)
FilterPacketsInterval=5000 // Restriction interval (ms)

In case the limit has reached, the client will receive CBIOS:ERR_CONN_REFUSED

4.2. Running the Server under Windows

4.2.1. General Issues

After the Smarx OS Network Server is installed (see chapter *3*), it can be administrated either locally (on the same PC where it is running) or remotely, using the Network Administrative Console.
When started, the Network Server icon appears in the system tray. Right-click for further options: To open the Server console select **"Open"**, to stop/start the Server select **"Stop/Start"** and to shutdown the Server select **"Exit"**.

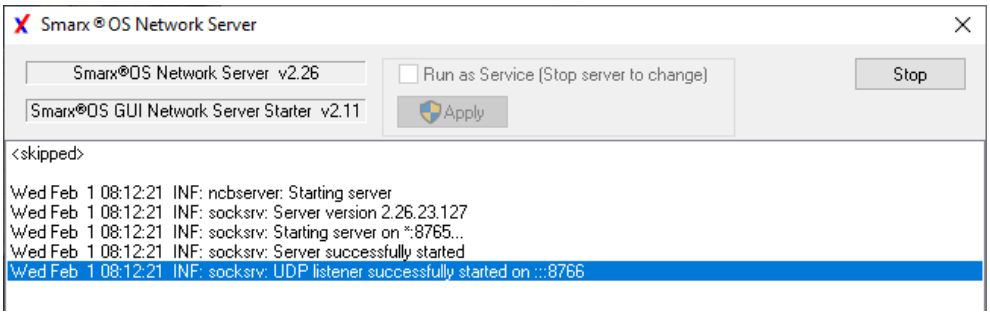

Figure 4.1:
Server Console

4.2.2. Administrative Application

The administrative application **AdminApp.exe/AdminApp64.exe** can be used for Smarx OS Network Server administration. It provides the following functions:

- Displays information about the CRYPTO-BOX and its partitions available on the server-side.
- Displays information about the network applications attached to the Server and the licenses available.
- Permits to disconnect all network clients (by restarting the server).
- Allows to change Smarx OS Server settings (to configure the server).
- Permits to change the password needed for Server Administration.
- Allows to restart or shutdown the server remotely.

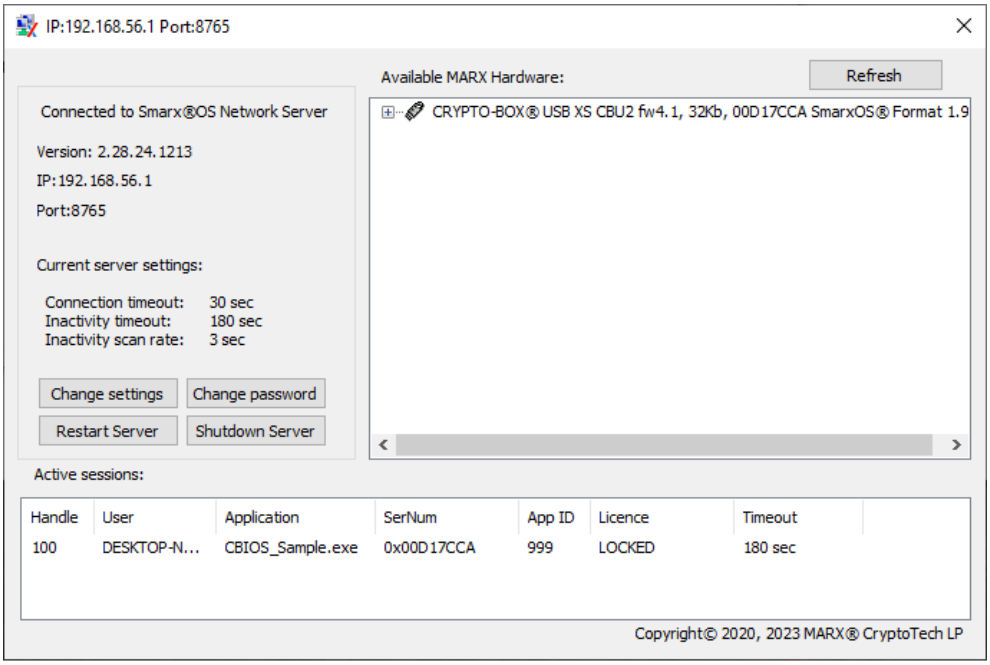

Figure 4.2:
Administrative Application

Connect to the Server
The Administrative Console connects to the server via TCP-IP protocol. Server Name (IP address – 127.0.0.1 by default, which means the local computer), Port (8765 by default) and Password (admin by default) need to be submitted on the connect dialog window.

Server Information
After connecting to the Server, information on Server settings, the attached CRYPTO-BOX and all active applications are displayed.

Change Server Settings
To change the server settings, click the "Change settings" button. Set the Server Connection timeout, the Inactivity timeout, and the Inactivity scan rate (see chapter *4.1* for more information).

Restart or Shutdown Server
To restart/shutdown the server press the "Restart Server" or "Shutdown Server" button.

Remember that when the server is shut down, it can only be restarted locally on the computer where it is installed!

4.2.3. Running the Server as a Service

The Server can be started as a system service under Windows; it will be started automatically when Windows is launched. Simply follow these steps:

- Run the server (CBIOSSRV.EXE or CBIOSSRV64.EXE).
- Stop the server using the "Stop" button in the server console window.
- Enable the "Run as service" option.
- Click the "Apply" button.
- Start the server by clicking the "Start" button on the server console window.

After that, you can exit the server launcher application (on the system tray). You will be asked if you wish to stop the service or keep it running.

Later, after the system reboot, the service will be launched automatically during Windows startup (no login to Windows required).

Administrative rights are required to register/unregister the server as service.

4.2.4. Unregistering the Server as a Service

- Run the server (CBIOSSRV.EXE or CBIOSSRV64.EXE).
- Stop the server using the “Stop” button in the server console window.
- Disable the “Run as service” option.
- Start the server by clicking the “Start” button on the server console window.

After the Server has been registered as a service, CBIOSSRV can be also managed through *Administrative Tools -> Component Services* in Windows Control Panel.

4.2.5. Monitoring of Network Licenses

The Smarx OS Network Server (Windows version only) supports monitoring of network licenses. This functionality allows customers to get real statistics and information on their network licensing process. Standard System Event Log is used for logging: open the Windows Event Viewer (*Control Panel -> Administrative Tools -> Event Viewer*) and look for the "CBIOSServer" section. The main advantage is that it is easy to access System Event Log from any application (even remotely).

Any application can subscribe through standard API and then be notified on new records of "CBIOSServer" type. The Server sends the following notification codes and logs corresponding types of events:

- Server Started
- Server Stopped
- License Locked
- License Unlocked

A sample application (including source code) which demonstrates monitoring of network licenses can be found in the Smarx OS Professional Protection Kit (PPK):
<SmarxOS PPK root folder>\Network\Samples\CBIOSServerEventLog_Sample

4.3. Running the Server under Linux

To start the Server open terminal and enter:
$ netcbios_server

The Server configuration file CBIOSSrv.cfg is located in the /etc directory. See chapter 4.1 for an explanation of the available settings.

Special configuration file can be set using -c parameter:
$ netcbios_server -c /etc/CBIOSSrv.cfg

Log file path can be set using -l parameter:
$ netcbios_server -l /var/tmp/netcbios_server.log

To start the Server in background:
$ netcbios_server &

You can obtain PID of the Server as the last background task:
$ echo $!

To stop server use kill command:
$ ps -l
$ kill %PID%
where %PID% is result from ps -l command for netcbios_server

5. Smarx®OS Network Client

5.1. Introduction

The **"Smarx OS Network Client"** describes an application which connects to the **Smarx OS Network Server** (see chapter *2*) rather than direct access to a CRYPTO-BOX attached to the local USB port of the computer. Network support is provide for the automatic implementation with AutoCrypt (see chapter *5.2*) as well as the Implementation with API (see chapter *5.3*).

5.2. Automatic Software Protection with AutoCrypt

AutoCrypt provides protection of applications without any programming efforts. It supports network licensing for Windows platforms. All information required for network licensing, such as:

- Server IP address and port (alternatively, automatic search can be selected);
- Maximum number of network licenses (seats) for the application;
- Check for locally attached CRYPTO-BOX first;

can be set up during the steps of protecting the application.

The steps of protecting applications including network licensing and CRYPTO-BOX formatting are described in the AutoCrypt Application Notes.

The AutoCrypt logic for detecting the CRYPTO-BOX in the network by the protected application will work the following way:

- If an IP address for the server was specified, it will search for this IP address.
- If automatic search via UDP broadcasting was specified, it will try to locate the server automatically.
- If the CRYPTO-BOX was not found on the specified address or via auto-search, the protected application will open a dialog window asking for server settings.
- If the server was found, the settings will be stored (in the Windows registry) for next application start.

5.3. Implementation with API

5.3.1. Differences Between API Calls for Local and Network Access

The access to a remote CRYPTO-BOX connected to the CBIOS Network Server is provided with the same interface functions as for local access, plus some extra functions, which are required in network mode. The functions listed below are for the standard CBIOS API (functions for other Smarx OS based APIs are similar):

CBIOS_ScanNetwork	Search Smarx OS Network Servers
CBIOS_SetScanPort	Set/Get UDP port used for network searching

CBIOS_GetScanPort	(default port is used if not set)
CBIOS_GetServerInfo	Get short information about Smarx OS Network Server
CBIOS_Connect	Connects to Smarx OS Network Server
CBIOS_Disconnect	Disconnects from Smarx OS Network Server
CBIOS_LockLicense	Locks the network license for the opened application (partition)
CBIOS_ReleaseLicense	Releases the network license for the opened application (partition)
CBIOS_GetAppLicenses	Retrieves license info for the application (partition) from LMT
CBIOS_SetAppLicenses	Sets license info for the application (partition) in the LMT
CBIOS_CheckAppLicense	Verifies the network license for the application

Only standard mode functions are supported on the network, extended functions for CRYPTO-BOX configuration (as available in the XSMRX COM object or in SxAF/SmrxProg configuration tools) are not allowed.

For more details on all API calls please refer to cbios.h (for standard CBIOS API) and to our Developer's Guides with API descriptions for different environments:

- CBIOS API reference for C/C++/Delphi/VB developers
- CBIOS4NET Developer's Guide for C# developers.
- For usage with SmarxAPI/AC API, see corresponding readme in SmarxAPI subfolder

If you use our high level AC API/SmarxAPI, you do not have to deal with API functions in detail, thus making implementation of network licensing much easier! See the Smarx Compendium, chapter 11and corresponding readme file in SmarxAPI sample folder for more details.

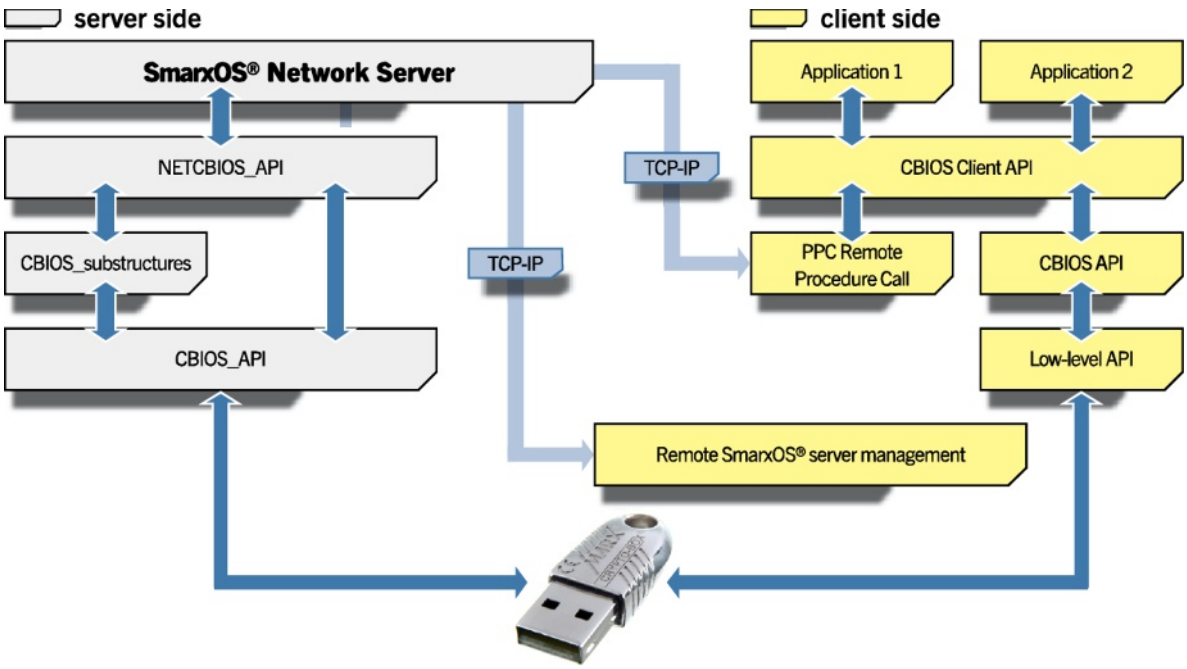

Fig. 4.1:

5.3.2. License Control System – Defining Number of Network Licenses

Each CRYPTO-BOX contains a special License Management Table (LMT) which is placed in a separate system partition of the CRYPTO-BOX memory (LCS system partition with AppID #5). Every record in this table contains information about the Application ID and the corresponding number of network licenses, defining how many clients are allowed to run the application simultaneously which accesses this Application ID (partition).

If there is no record for an Application ID, the number of network licenses for this Application ID is unlimited.

The License Management Table and the number of network licenses for each partition can be programmed using either the Smarx Application Framework (GUI-based) or the SmrxProg command line tool. Please refer to the Smarx Compendium, chapter 4.5 for more details.

When a program tries to login to a proper Application ID (partition), Smarx OS searches for the LCS record for this partition first. When found, the number of permitted licenses and the number of clients currently logged in, are checked. If there are free licenses available, access to the partition is granted, otherwise it is denied.

Every Smarx OS based application can access a local CRYPTO-BOX as well as a remote CRYPTO-BOX (attached to the Smarx OS Network Server somewhere in the network).

The Smarx OS API allows all clients to retrieve general information on the CRYPTO-BOX attached, including serial number, digital signature status, model info, etc., even if there are no free licenses available. The License Control System (LCS) is active only when the client application tries to **login** to an Application ID/partition.

The procedure of login/logout functionality in network mode is different than in local mode!
In local mode, all operations which require a login (read/write access to RAM1 or RAM2, access to encryption functions) have to placed in login/logout brackets, because after login the CRYPTO-BOX is can be used only exclusive by the current application/thread and is blocked for other threads. In network mode, the Network Server manages access from different applications. Therefore, before locking a license for one Application ID (partition), the application must login (and stay logged in until the license will be released. Please check the corresponding network licensing sample code for your compiler in the Protection Kit. It will help you to understand this concept.

5.3.3. Typical Network Session Scenario

First of all, the protected application needs to connect to the Smarx OS Network Server. This can be done through the **CBIOS_Connect** function, if the Server Network Name or IP address is already known to the client. In other cases, the client can try to search for available Smarx OS Network Servers via **CBIOS_ScanNetwork**.

After connecting successfully, the client application can retrieve information on all CRYPTO-BOX units and partitions available on the server-side, open a partition and login to a CRYPTO-BOX using standard CBIOS functions.

To lock a network license for the opened partition, use the **CBIOS_LockLicense** command. The license counter - taken from License Management Table (LMT, see chapter *5.3.2*), if present, will be decremented. If the network license for the application/partition is UNLIMITED, no actions will be performed. To release the network license of the open partition, use **CBIOS_ReleaseLicense**. The license counter will be incremented.

The **CBIOS_LockLicense** and **CBIOS_ReleaseLicense** functions can also be present in code written for local

applications (if the same code is shared for local and network applications). In local mode, these functions can be used to prevent several copies of a protected application from being launched through a Terminal Server.

Before a Smarx OS network application may be closed, it must be disconnected from Smarx OS Network Server using the **CBIOS_Disconnect** function. If the application shows no signs of vitality (ignores keep-alive messages) within the timeout period defined in the server settings (see chapter *4*), it will be automatically disconnected.

5.3.4. Support for License Binding

A special feature of the Smarx API is the ability to bind the protected software to a specific computer. For example, this can be beneficial if the use of the protected software outside the company is not permitted.

There are two possibilities of license binding in network mode:

- The protected software will be bound to the network server where the CRYPTO-BOX is attached
- The protected application will be bound to the client computer which issued the binding request.

For more details on license binding, please refer to the Smarx Compendium, chapter 14.2.

5.3.5. Network Licensing - Sample Code for Developers

The Smarx OS Protection Kit contains sample code which demonstrates CBIOS network licensing logic for all popular compilers and platforms. See section “Software Protection with Smarx API for Developers” in the Protection Kit Control Center for more details.

The following prototype samples for integrating CBIOS Network Licensing logic into your application are included in the Smarx OS PPK:

<SmarxOS PPK root folder>\SmarxOS-Samples \CBIOS Network Licensing\

Besides basic network licensing functionality the samples cover such typical situations as:

- Network connectivity losses and/or CBIOS Server stopped responding.
- Unplugging/plugging back the CRYPTO-BOX on the computer running CBIOS Server;
- and more.

The samples can serve as a prototype for developers integrating CBIOS Network Licensing logic to their applications. They written in MFC C++ (Visual Studio) resp. C# (.NET 4.5 or later), but it serves as a reference code which can be adapted to other programming languages as well.

5.3.6. Network API Calls

For a detailed description of all Smarx OS API calls, see the Developer's Guides with API descriptions for different environments: CBIOS API or SmarxCpp reference for C/C++/Delphi/VB developers and CBIOS4NET Developer's Guide for C# developers.

5.4. Smarx Cloud Security – User Authentication and License Management for Web Applications

Smarx Cloud Security allows a web server to communicate via HTTP with the CRYPTO-BOX attached to the client’s computer or network. No customer specific software is needed on the client’s computer. It works with all standard web browsers.

The communication between the client and online-services is encrypted. This solution can be used for various scenarios:

- Secure user authentication for web sites/portals.
- Licensing of web based applications and services to end-users. Only users with a valid license in the CRYPTO-BOX, either attached to a local computer or in the network, will have access to the web content.
- Automation of remote updates of licensing information stored inside the CRYPTO-BOX, no manual processing is required.

More details are available at marx.com → Products → Cloud Security. Sample code which demonstrates the following features is available on request:

- Access to the CRYPTO-BOX connected either to the local USB port of the computer or on the network.
- Support for Notification (automatic detection if a CRYPTO-BOX is attached or removed).
- Binding the license to a specified computer or location (see chapter *5.3.4*).

Sample code is avilable for PHP, Java/JSP and ASP.NET. Please contact us for further details and sample code.

Please refer to the Smarx Cloud Security White Paper for further details.