2nd Generation License Protection Mechanism
POLQA OEM Library

Version 2.0

15 March 2019

1 Contents

2 Preface

This document describes the 2nd generation copy protection and license management scheme which is introduced in OPTICOM products in 2018.

The background for this update is that many customers request more flexibility with the available types of licenses as well as easier installation of licenses on target machines (e.g. remote installation) and simplified accounting. At the same time, the traditional method with unlimited keys for large volume customers posed a significant risk on these customers in case the keys get stolen or abused.

The goal is therefore to enhance and automate the installation process of new licenses, to prepare features for future, more flexible licensing models and to secure the entire process.

The major new element is the introduction of a Web API which allows the automated retrieval of licenses from OPTICOM's web-based global license management server ("LMS"), as well as the verification of already installed licenses. This verification of licenses is also essential if in the future licenses shall be transferrable from one end product to another.

The main breaking changes compared to the old method are that unlimited keys will not be available anymore and the way in which keys will be distributed. All future keys will be bound to a device and need to be activated online. The traditional method of uploading a LicenseInfoFile to the license management server (LMS) and downloading a license from the server to the device is also still available but complemented by a web API which significantly simplifies this procedure.

This manual is for existing customers and explains the breaking changes compared to previous versions of OPTICOM OEM Libraries.

3 Overview

The new license management concept consists of few simple steps, provisioning licenses, device registration and license activation.

Provisioning of licenses consists of making licenses available on the OPTICOM License Management server (LMS). For small volume customers, this is handled by OPTICOM during the order process. For large volume customers, which report their sales quarterly, provisioning is handled by logging in to the LMS and adding a bunch of licenses to a specific license pool.

During device registration, newly manufactured devices will contact the OPTICOM LMS once using an OEM customer specific user name and password. This will associate the device with a license pool and prepare it for future safe connections to the LMS by exchanging X.509 certificates.

If a device shall be equipped with a license, it needs to activate a license available in the license pool which was associated with the device during device registration. This requires another contact with the LMS. This time no user name or password are required. During this step a license file which is bound to the device will be downloaded to the device and the license on the LMS will be marked as activated (you can use this information for your license reporting as well). The downloaded license file will not work on any other device.

Device registration and license activation are supported by API functions available in our OEM Library as well as command line options of the demo executable which is provided in source code with our OEM Libraries. Of course, it is possible to perform both steps simultaneously. If you whish to integrate this process with other licensing components like e.g. FlexLM, you can easily link your activation software with our OEM library and call the OPTICOM API functions for license management in your own application.

After a perpetual license has been issued to an Android device, no more access to the LMS is required. All other licenses however, especially all Time limited or transferable licenses as well as Cat. D licenses however, will require occasional access to the LMS for license verification. This happens automatically during normal operation. If a contact to the LMS is overdue a warning will be issued long before the license expires, leaving the end user enough time to react.

1. Log on to LMS using web browser and GUI, allocate some licenses to your own pool. Log off again.

2. **Register** device: Execute a small program on the device, requires user name, password and a pool name for the LMS. This will register the device on the LMS and install a certificate on the device. Typically, this will happen during manufacturing.

3. **Activate** a license: The device connects to the LMS using its certificate. A license is drawn from the pool and saved on the device. Can happen at end-user site.

Registration program must be deleted once the device is registered (it contains sensitive information).

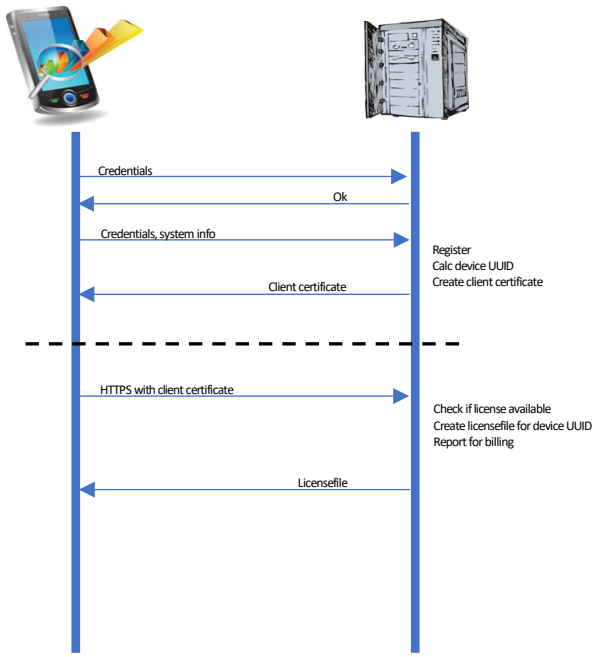

Figure 1, Outline of license provisioning, device registration and license activation.

4 Advantages

For small volume licensees, this new method is much easier than the old concept of the 1st generation, where a device specific file had to be copied manually to our LMS and subsequently a licensefile had to be copied back to the device. This process is now fully automatic.

For large volume customers, the new concept may appear a bit more laborious than a hard-coded key at first, but it allows for more flexible licensing models in the future (e.g. transferable licenses). In addition, it is much safer than the old concept and avoids fraud (be intentional fraud or oversight). This leads to more fairness amongst our licensees. Also, the risk and liability inherent to a lost or extracted unlimited license key was significant in the past (and yes, this risk was on the licensees' side!). With the new concept, the risk is very small and the only use case which is a bit riskier is device registration at end user premises (see section 8), but even in this situation, the liability is significantly lower than before.

Also, the integration with existing license management software e.g. Flexnet and the like, is very easy and requires very few additional lines of code in your applications. Essentially, all it takes are two simple POLQA API function calls after activating a license using Flexnet.

Another advantage for large volume customers is that they may want to use the reporting feature of our LMS to verify the quarterly reporting of sold licenses. Numbers will typically not match exactly due to accounting periods, but the LMS reporting can give the accounting a solid basis and gross errors (e.g. differences between internal accounting and the LMS reported numbers) should be easily detected. Please note, that this is a feature which one may use if desired, but there is not need to change any established reporting solution.

5 Quick Start: Using the POLQA OEM Demo Program

All readers of this manual are probably familiar with the demo program "PolqaOemDemo.exe" (resp. PolqaOemDemo64.exe") which is supplied with the POLQA OEM Library. We extended this program to make it usable for device registration and license retrieval as well. The easiest way to get started with POLQA V3 and to acquire a new license is through this demo program. Except for the demo program you will also need the user credentials for the license management server, which you should have received together with the new library and this manual.

NOTE: In a larger production environment it is typically required to fully automate installation of software and licenses on devices. This requires a lot of testing as well as usually some trial and error. To allow our large volume customers to test their production environment, access to a special test LMS is available on request. See section 12 for details.

To register a device, type the following from a command prompt (replace <…> by your user credentials):

```
PolqaOemDemo -Register <username> <password> <pool>
```

The machine / device on which you executed this line is now registered at the LMS and added to the pool <pool>. The address of the LMS is currently hard-coded in the demo program. Currently, there is exactly one pool per user, but this will change and users shall be able to manage their own pools.

After registering the device, the following line can be used to retrieve a license from the pool:

```
PolqaOemDemo -Activate
```

Again, the license type is hard-coded in the demo program.

The demo program is now fully activated and you can execute POLQA like e.g.:

```
PolqaOemDemo -Ref RefSWB.wav -Test TestSWB+1s.wav -LC SWB -Version 3
```

Certainly, it is also possible to perform all steps at once:

```
PolqaOemDemo -Register <username> <password> <pool> -Activate
```

Or:

```
PolqaOemDemo -Ref RefSWB.wav -Test TestSWB+1s.wav -LC SWB -Version 3 -Register
<username> <password> <pool> -Activate
```

Please note, that the sequence of the arguments doesn't matter, but of course, the keywords starting with '-' must be directly followed by the proper values.

Since the demo program is provided in source code, you may modify all hard-coded parameters or use it as a starting point for developing your own application.

6 License Administration Using the LMS GUI

Currently, no interaction with the LMS administrative interface is required. It is planned however, that large volume customers will be able to manage their own pools using the graphical web user interface. Documentation regarding this feature will be added accordingly.

For users which don't want devices to talk directly to the LMS, it is still possible to upload a licenseinfofile to the LMS and receive a package with all required files for downloading. The process is exactly the same as with 1st generation licenses, the only difference being that instead of a single licensefile, a whole bundle of files packed in a zip archive will be downloaded. This archive must be extracted to the license folder on the target machine. The licensefolder can be the default location (see section 8.4) or any other folder on the local disc. In the later case, the folder must be specified in the call to PolqaLiblnit().

7 Caveats

There are some important points to be observed with the new license protection scheme, which may not be obvious to new users:

7.1 When do Renewal Attempts Occur?

Typically, each license type has a specific renewal period. This period may vary between 1 day for server licenses up to several months for runtime licenses. Whenever the end of such a renewal period is reached, the software will try to automatically renew the license. If this renewal fails, new attempts will be made with each measurement, but at the earliest ten minutes after the last failed attempt. This implies that in order to automatically renew the license on a device, the device should be connected to the internet for at least ten minutes after the last measurement and then a new measurement should be performed.

7.2 Not Matching License Types During Registration

Please note that all parameters of the license specification in the call to PolqaLibActivateLicenseX() must match exactly the type of license available on the LMS. This also includes the POLQA version. If the demo program is started with the option "-version 2", but the LMS has a license for POLQA 3 only, then the activation will fail.

7.3 Memory leak

When debugging your own software, you may observe, that there is a memory leak caused by the POLQA OEM Library. This is due to one of the libraries used by POLQA itself, which manages its own memory. The memory will be released at termination and the amount will not accumulate (it may vary though). It is not possible, to release this memory, while the POLQA OEM Library is still loaded.

8 License Management API Details

8.1 License Provisioning (Large Volume Customers only)

Before licenses can be activated on devices, they must be provisioned on the License Management Server (LMS). For the moment, provisioning is possible via the web interface only and not through API calls. During the trial phase, one license type is provided only, which is a "Evaluation License 7 days", 2 channels, class C license. These licenses are enabled as "self generated", which means that all requests will be fulfilled with either an existing or a newly generated license. The number of generated licenses can be viewed and monitored via the web GUI of the LMS. This license type is also hard-coded as default in the demo program.

8.2 Device Registration

Device registration is performed by calling the POLQA API function

```
OEMPOLQADLL_API POLQA_ERRORCODE PolqaLibRegisterDeviceUTF8(
       INSTDATA* pInstanceData,
       const char* Username,
       const char* Password,
       const char* Pool,
       const char* OEMDeviceID,
       const char* Hostname,
       const char* Licensefolder=0);
```

Or on Windows based systems its UNICODE equivalent

```
OEMPOLQADLL_API POLQA_ERRORCODE PolqaLibRegisterDeviceW(
       INSTDATA* pInstanceData,
       const wchar_t* wUsername,
       const wchar_t* wPassword,
       const wchar_t* wPool,
       const wchar_t* wOEMDeviceID,
       const wchar_t* wHostname,
       const wchar_t* wLicensefolder=0);
```

The parameters in the call to these functions have the following meaning:

pInstanceData	A pointer to an object of type INSTDATA which you must provide. The object may be initialized to 0 and shall be passed on to all other functions requiring this pointer.
Username	Your username for accessing the OPTICOM License Management Server (LMS)
Password	Your password for accessing the OPTICOM License Management Server (LMS)
Pool	The pool on the LMS from which you want to pull licenses for this device.
OEMDeviceID	An arbitrary name which you may give to the device.
Hostname	The hostname of the OPTICOM LMS (typically lmsapi.opticom.de)

Licensefolder	Leave this empty to store all licensing related stuff in a default location on the local device. See also section 1.4.

The return code will be one of the error codes from the `POLQA_ERRORCODE enum.`

After calling this function, the device will receive a client certificate which is issued by the LMS. All further attempts to contact the LMS will use this certificate for authentication. Username and password are nor needed anymore. Be careful when choosing the correct license pool since the device will be associated with that pool forever. Please note that this function can be called without calling any other POLQA API function first (a call to PolqaLibInit() would even fail without a valid license..

8.3 License Activation

Prerequisites: *- The device must be registered with the LMS*

After a device has been registered with the LMS, licenses can be activated by calling the POLQA API function

```
OEMPOLQADLL_API POLQA_ERRORCODE PolqaLibActivateLicenseUTF8(
       INSTDATA* pInstanceData,
       const char* LicenseType,
       const char* LicenseClass,
       int EffectiveChannels,
       const char* Version,
       bool UseExisting,
       const char* Hostname,
       const char* Licensefolder=0);
```

Or on Windows based systems its UNICODE equivalent

```
OEMPOLQADLL_API POLQA_ERRORCODE PolqaLibActivateLicenseW(
       INSTDATA* pInstanceData,
       const wchar_t* wLicenseType,
       const wchar_t* wLicenseClass,
       int EffectiveChannels,
       const wchar_t* wVersion,
       bool UseExisting,
       const wchar_t* wHostname,
       const wchar_t* wLicensefolder=0);
```

The parameters in the call to these functions have the following meaning:

pInstanceData	A pointer to an object of type INSTDATA which you must provide. The object may be initialized to 0 and shall be passed on to all other functions requiring this pointer.
LicenseType	The type of license which you request. This must be a string which looks exactly like the license description in the LMS Web GUI, e.g. "Evaluation License 7 days".

LicenseClass	The category for which the license shall be valid ("A", "B", "C" or "D"). See your license agreement for details.
EffectiveChannels	The number of effective channels for which the license shall be valid.
Version	The POLQA version for which this license shall be valid (“2” or “3”).
UseExisting	If true, a matching, a free license will be searched in the pool for which the device is registered. If a matching license was found, a license-file will be stored in the local license folder on the device, otherwise an error code will be returned. If false, the LMS will check if “self-generated” licenses of the specified type are enabled for the pool for which the device is registered. If they are enabled, a matching license will be stored in the local license folder on the device, otherwise an error code will be returned.
Hostname	The hostname of the OPTICOM LMS (typically lmsapi.opticom.de)
Licensefolder	Leave this empty to store all licensing related stuff in a default location on the local device. See also section 1.4.

The return code will be one of the error codes from the `POLQA_ERRORCODE enum.`

Calling this function, a licensefile will be retrieved from the LMS and stored locally on the device in the specified Licensefolder. If the Licensefolder is 0 or a 0-length string, the library will choose a default location (recommended).

8.4 Specifying the Licensefolder

The three functions `PolqaLibRegisterDeviceX()`, `PolqaLibActivateLicenseX` and `PolqaLibInitX()` allow the specification of a “Licensefolder”. For use in combination with new licenses we clearly recommend to leave this parameter empty (pass 0, or a zero-length string). This will choose a system dependent default location. You may however decide to pass a string to an existing folder which will then be used. This folder must also contain the file root-ca.crt, which you received together with the POLQA OEM Library. The root_ca.crt file will also be generated by the POLQA OEM Library if it does not exist `PolqaLibRegisterDeviceX()` is called.

It is also possible to use POLQA with older licensefiles (see section 6). In this case, the full path of the licensefile has to be specified and root-ca.crt is not required. Please make sure, that such older licensefiles use the extension “.txt”.

The default license folder on different platforms is located as indicated below:

On Windows this is: %LocalAppData%\OPTICOM\POLQA\
On Linux this is: The current working directory
On Android this is: /sdcard/opticom/polqa/

9 Example Program

The lines below demonstrate a very simple code snippet for device registration and license activation.

```
#include "PolqaDllInterface.h"
#ifdef _WIN32
#ifndef STRING_TYPE
   #define STRING_TYPE  wchar_t
   #define STRING_PREF  L
   #define STRING_COPY     wcscpy
#endif
#else
#ifndef STRING_TYPE
   #define STRING_TYPE  char
   #define STRING_PREF
   #define STRING_COPY     strcpy
#endif
#endif


// LMS account specific
#define DEFAULT_LMS     STRING_PREF""
#define DEVICE_NAME     STRING_PREF"My devicename"
#define PASSWORD        STRING_PREF"MyPW"
#define USERNAME        STRING_PREF"Me"
#define POOLNAME        STRING_PREF"My pool"

// License specific
#define LICENSE_TYPE              STRING_PREF"Evaluation License 7 days"
#define LICENSE_CLASS             STRING_PREF"1"
#define LICENSE_CATEGORY          STRING_PREF"C"
#define LICENSE_POLQA_VERSION     STRING_PREF"3"

// For 1st gen licenses only:
#define LICENSEFILE_NAME    STRING_PREF"full path to the licensefile"

// Demo code which is listed in the manual. We compile it here to ensure correctness.
void RegisterAndActivate()
{
   // LMS account specific
   STRING_TYPE  OEMDeviceID[MAX_STRING_LEN] = DEVICE_NAME;
   STRING_TYPE  Password[MAX_STRING_LEN]    = PASSWORD;
   STRING_TYPE  UserName[MAX_STRING_LEN]    = USERNAME;
   STRING_TYPE  Pool[MAX_STRING_LEN]        = POOLNAME;

   // License specific
   STRING_TYPE  LicenseType[MAX_STRING_LEN]     = LICENSE_TYPE;
   STRING_TYPE  LicenseClass[MAX_STRING_LEN]    = LICENSE_CLASS;
   STRING_TYPE  LicenseCategory[MAX_STRING_LEN] = LICENSE_CATEGORY;
   STRING_TYPE  Version[MAX_STRING_LEN] = LICENSE_POLQA_VERSION;
   int    EffectiveChannels  = 2;
   bool   UseExisting = false;          // If true, use licenses already existing in the pool,
                                        // otherwise, generate a new license ("self-generated
                                        // license")

   POLQA_ERRORCODE PolqaErrorcode=POLQA_OK;    // Returncode of Polqa modules
   INSTDATA         InstanceData=0;

   printf("Registering device...\n");
   PolqaErrorcode = PolqaLibRegisterDevice(&InstanceData, UserName, Password, Pool, OEMDeviceID, 0);
   if (POLQA_OK == PolqaErrorcode)
   {
      printf("Activating license...\n");
      PolqaErrorcode = PolqaLibActivateLicense(&InstanceData, LicenseType, LicenseClass,
                           LicenseCategory, EffectiveChannels, Version, UseExisting, 0);
      if (POLQA_OK == PolqaErrorcode)
      {
         printf("License activation successful\n");
      }
      else printf(" PolqaLibActivateLicense() failed: %d\n", PolqaErrorcode);
   }
   else printf(" PolqaLibRegisterDevice() failed: %d\n", PolqaErrorcode);
```

```
    if (PolqaErrorcode != POLQA_OK)
        throw((PolqaErrorcode));
}
```

10 Backward Compatibility

Old large volume keys as well as older licensefiles can still be used, but functionality will be restricted to using POLQA 2 and POLQA 1. POLQA 3 will require device registration and new, device specific keys of the 2nd generation licensing method.

If used in combination with old keys, the only change to your program consists in adding one parameter in the call to PolqaLiblnit() and recompiling/linking the program with the new header files and the new library. You should also check your reporting of error codes since some error codes returned by the POLQA OEM Library may have changed.

The new parameter in the call to PolqaLibInit () is the second parameter, "INSTDATA* InstanceData", which may be 0.

Please note, that by default dongles are not supported anymore. Please consult product specific documentation (e.g. the POLQA or PEVQ-S manual) for details, exceptions and legacy dongle support.

It is also possible to manually retrieve a license from the LMS, by uploading a "licenseinfofile", which is generated by the demo program when started with the option "-Mac". The licensefile can subsequently be downloaded from the LMS in a very similar way as with older versions of the POLQA OEM Library. The big difference however is that now the download consists of a bunch of files which are stored in a zip archive. This archive must be unzipped and all files must be stored in the same location. We recommend the locations specified in 1.7.

11 Upgrading / Installing Devices at End-User Premises

Some of our customers require that devices can be upgraded at end-user premises and of course, this is possible. In this scenario, two cases need to be distinguished.

1. The device is already registered with the LMS and the license needs to be activated only.

2. The device needs to be registered with the LMS first.

The first scenario is very simple. Since the device is already registered, it contains all the information required to contact the LMS and to activate a license. All it takes is to make sure the device has a working internet connection and then follow the steps from section 1.3.

The second scenario is a bit more complex, since device registration requires the specification of a user account on the LMS in plain text (but of course, it is not transmitted unencrypted). We therefore recommend to register such devices with a special license pool which contains very few licenses which drastically limits the risk of abuse. All API calls required to register and activate the device should be contained in your installation program. The installation routines which contain the LMS access credentials must be automatically deleted immediately after the device registration.

12 The trial LMS

We have set up a small demo server which can serve only one type of very short-lived licenses in order to allow for playing with the new concept and to test integration with manufacturing processes. Of course, you will not be charged for licenses retrieved from the test LMS. The licenses available are so-called "self-generated licenses", which means that the LMS will generate a new license whenever you request one for a device for which no valid license exists already. This is the typical scenario for large volume customers.

You can access it using the following credentials:

Host (GUI):	https://OpticomExtTestServer1.ext1.opticom.de
Host (API):	OpticomExtTestServer1api.ext1.opticom.de:443
Username:	provided via separate email on request
Password:	provided via separate email on request
Poolname:	provided via separate email on request

For the moment, this account is set up to provide solely "Evaluation License 3 days" type licenses for Cat. C and two effective channels for POLQA V3. These licenses are always self-generated. The renewal and grace periods are also set very short:

Renewal Period	1 h
Warning Period	2 h
Grace Period	3h

13 Contact Information

OPTICOM

Dipl.-Ing. M. Keyhl GmbH

Nägelsbachstrasse 38

D - 91052 Erlangen

GERMANY

Phone: +49 (0) 91 31 - 5 30 20 - 0

Fax: +49 (0) 91 31 - 5 30 20 - 20

EMail: info@opticom.de

Webseite: http://www.opticom.de

Further information:

http://www.polqa.info

http://www.opticom.de