objective
SYSTEMS, INC.

ASN1C

ASN.1 Compiler
Version 6.5
Installation Guide

Objective Systems, January 2012

Author's Contact Information:

Comments, suggestions, and inquiries regarding this document may be submitted via electronic mail to info@obj-sys.com.

Table of Contents

Installing the ASN1C Compiler Software

Version 6.x of the ASN1C compiler software is packaged in two or three separate distribution units, depending on how ASN1C is licensed:

- A *System Development Kit* (SDK) unit that contains the compiler and development libraries, and

- One or more run-time deployment units containing optimized binary libraries for deployment of a finished application.

- A floating license server application (available for Windows and various UNIXes).

The following sections contain installation instructions for Windows and UNIX versions of the ASN1C distribution files. This procedure is the same for all of the different configurations.

The last section describes techniques to help port code generated by earlier versions of the compiler to version 6.0.x.

Microsoft Windows Distribution

The Microsoft Windows version of ASN1C is distributed either on a CD and/or electronically over the Internet. The distribution files are self-extracting, executable setup files. The format of the filename of the SDK unit is as follows:

> **ac<L>v65xw32sdk.exe**

where *<L>* would be replaced with a single-letter language code. Possible values are *p* for C/C++, *j* for Java, or *s* for C#.

Run-time deployment packages have the following format:

> **rt<L>v65xw32<TYP>.exe**

where *<L>* would be replaced with a single-letter language code and *<TYP>* would be replaced with a code for the package type. Possible values for the language code are *p* for C/C++, *j* for Java, or *s* for C#. The type code would contain information on whether the library is licensed per-host (limited) or unlimited, whether source code is included or not, and what encoding rules are supported.

For example: the following would be an ASN1C v6.5.0 run-time deployment kit for Java:

> **rtjv650w32ubb.exe**

In this case, the *ubb* on the end stands for unlimited, binary, BER/DER.

The floating license package is versioned separately from ASN1C and is distributed in an executable self-extracting installation file (like other Windows kits). The package is named with the following format:

> **rlmsvrw32.exe**

RLM Floating License Server Installation

This section briefly describes the installation of a RLM floating license server; users are encouraged to consult the README.html file included with the package for additional details. The README is also available on-line at http://www.obj-sys.com/docs/RLM/README_w32.html.

The server can be started from a command line or run as a service. It requires an license file (asn1c.lic) to run.

The *rlm* command is:
```
% rlm [-c license_file] [-dlog [+]logfile]
    [-nows] [-ws port] [-x [rlmdown|rlmremove]]
    [-install_service] [-service_name sname]
```

The **-c license_file** option specifies which license file to use. This option overrides the setting of the RLM_LICENSE environment variable. The **license_file** parameter can be a directory containing license files, all of which will be processed.

The **-dlog logfile** specifies the pathname for the server debug log. If **logfile** is preceded by the '+' character, the logfile will be appended, otherwise it is overwritten. (Note: all ISV servers will write their output to the same logfile specified in the -dlog option.)

The **-nows** and **-ws port** options control the operation of the embedded Web Server. The *-nows* option instructs the *rlm* server to not start the embedded web server. The *-ws port* option instructs the *rlm*server to use *port* as the port number for the web server.

The **-x [rlmdown | rlmremove]** option controls whether the rlmdown and/or rlmremove commands will be processed by the server. Specifying only **-x** will disable both commands. Specifying either command name after the -x will disable just that command.

The **-install_service** and **-service_name sname** options are used to run the *rlm* server as a service under windows. See the description of running the *rlm* server as a service below.

These options can appear in any order on the command line.

Note that if the *rlm* server cannot bind the web server port (5054 by default), it will exit.

You can install RLM as a service in a command window. To do this, use the rlm program itself (in a command window), with special arguments:

rlm -install_service -dlog [+]logfile [-service_name sname] <rlm runtime args>

where:
- *logfile* is the pathname for the server debug log. This parameter is required. If preceded by the '+' character, the logfile will be appended, rather than created.
- *sname* is an optional name for the installed service. If not specified, *sname* defaults to "rlm". If *sname* contains embedded whitespace, it must be enclosed in double quotes.
- <rlm runtime args> are any other command line arguments to be passed to rlm when it is started.

Example:
> *rlm -install_service -service_name rlm-xyz -dlog c:\logs\server.log -c*
> *c:\licenses\xyz.lic*

This installs rlm as a service under the name "rlm-xyz". When started via the Services control panel or at boot time, rlm will be passed the "-c c:\licenses\xyz.lic" args, and it will write it's debuglog information to the file c:\logs\server.log

Installed RLM services are also deleted with the rlm program. Services must be stopped via the service control panel before they can be deleted. Note that deleting a service deletes it from the Windows service database; it does not delete the rlm executable or associated license file(s):

rlm -delete_service [-service_name sname]

where:

   sname is an optional name for the installed service. If not specified, service_name defaults to "rlm". If service_name contains embedded whitespace, it must be enclosed in double quotes.

ASN1C System Development Kit (SDK) Installation

An ASN1C distribution that uses the floating license server works by contacting the server to obtain a license key. A license file (asn1c.lic) is supplied with this installation. It facilitates communication with the floating license server.

The procedure to install the ASN1C compiler and run-time libraries is as follows:

1. Double-click the SDK installation program kit executable filename. This is the filename in the format described above.

2. Follow the setup program instructions.

3. You should have received a license file (asn1c.*lic*) or a license activation key to enable the compiler to run on a given node or connect to a license server. If you have received the license file (asn1c.lic), go to step 4. The license activation key can be used in two ways:

    1) From a command line, use option '-lickey <license key>'

    2) From asn1c GUI, enter the key in the 'License key:' field:

    Note: If there is a proxy server in your environment that the ASN1C is unaware of, you may see error message. You can use the web activation interface at http://obj-sys.com/rlm_activation.php. Enter the license key and your host ID which is the MAC address or physical address without the '-', for example 0019d1e7802f . The license will be generated and displayed in the web page. Save it into a license file with an extension .lic and copy it into one of the following locations listed in step 4.

4. Copy the license file into one of the following locations:

    a. One of the directories specified in your PATH environment variable, or

b. In a different directory and create a new environment variable name OSLICDIR that points to this directory, or

c. Into the same directory as the ASN1C compiler executable file (asn1c.exe)

If installation was successful, you should have both a graphical user interface (GUI) compilation wizard available as well as a command line version of the compiler. The GUI wizard can be tested by starting the application and entering the data requested in each of the dialog prompts. The command-line version of the compiler can be tested as follows:

1. Open an MS-DOS or other command shell window.

2. Change directory (cd) to the compiler root directory. The default directory in the setup script is **c:\acv<version>** where **<version>** is the version number of the compiler. For example, **c:\acv650** is the default root directory for version 6.5.0 of the compiler.

3. Enter **.\bin\asn1c** from the command line prompt. You should see a usage display of compiler command line options (these options are discussed later). This indicates the compiler is properly installed and working.

If you get a message indicating the license file could not be found, please review the procedure in step 3 above to make sure it is installed in the proper location.

You should include the <target>\bin (e.g. c:\acv650\bin) directory in your PATH environment variable in order to run the compiler from anywhere.

ASN1C Run-time Deployment Kit Installation

The deployment run-time packages can be installed at any time after the SDK is installed. They are not necessary for basic program development. They should be used when an application is ready to be deployed. To install, do the following:

1. Double-click the installation program kit executable filename. This is the filename in the format described above.

2. Follow the setup program instructions. Note that the root directory for the installation should be the same as was specified for installation of the SDK package described earlier.

The result will be additional library subdirectories of the form *lib_opt* added to the directory hierarchy. These contain the optimized libraries. To link with these libraries, either the makefile(s) or project files used to link the application must be changed, or the subdirectories must be renamed. For example, the existing *lib* subdirectory could be renamed *lib_nonopt* and *libopt* could then be renamed to *lib*.

ASN1C runtime deployment kits cannot work with a floating runtime license. When a limited runtime kit is installed, a binary license key (*rtkey.dat*) must be used to deploy the libraries.

Testing the C or C++ Run-time Components

The default C or C++ run-time libraries for Windows were built with Microsoft Visual C++ V9.0. Other libraries are available that have been built with the Borland C++ compiler (v5.5) and with the Microsoft Visual C++ v6.0, v8.0, and v10.0 (.NET) compilers. If you have the version of the product that includes run-time source code, you can rebuild the run-time libraries using any ANSI-standard C or C++ compiler.

You can verify operation of any of the different run-time libraries by executing the sample programs. These can be found in the *sample_ber*, *sample_der*, *sample_per, sample_xer* and *sample_xml* subdirectories.

For example, we will assume that you installed ASN1C for C/C++. To test the BER C++ encode/decode capabilities, do the following:

1. Change directory (cd) to the **.\cpp\sample_ber\employee** subdirectory. Execute the **nmake** command to build the writer and reader sample programs. **nmake** is utility program that comes with Visual C++. It may also be necessary to execute a Microsoft batch file named **VCVARS32.BAT** to set the path information so that the **nmake** utility can be found. (Important note: this assumes you are using Microsoft Visual C++ on your PC. Some PC specific include and library directories in the makefile may need to be changed to get the samples to work on your system. See the README.txt file for further details).

2. Execute the **writer.exe** program to encode the sample record. The results of the encoding will be dumped to the screen and saved in a file called *message.dat*.

3. Execute the **reader.exe** program to read and decode the contents of the *message.dat* file. This program will read the encoded record into memory, decode it, and then print the contents of the generated structure variable to standard output.

Testing PER is similar:

1. Change directory (cd) to the **.\sample_per\employee** subdirectory. Execute the **nmake** command to build the writer and reader sample programs.

2. Execute the **writer.exe** program to encode the sample record. The **–a** switch can be used to encode a record using aligned PER. The **–u** switch encodes a record using unaligned PER. The results of the encoding will be dumped to the screen and saved in a file called *message.dat*.

3. Execute the **reader.exe** program to read and decode the contents of the *message.dat* file. The **–a** or **–u** switch must be the same as that specified when the writer program was executed. This program will read the encoded record into memory, decode it, and then print the contents of the generated structure variable to standard output.

This test can be repeated for XER as well by going to the **sample_xer\employee** subdirectory and repeating the above sequence of steps.

UNIX Distribution

Floating License Server Installation

RLM server for Linux and MacOSX distributions

Refer to the [RLM server for Windows](#) section.

objsys server (licServer) for other UNIX/Linux distributions

This section briefly describes the installation of a floating license server on a UNIX system. More detail may be found in the README.txt file that accompanies the installation.

The floating license server is an application that should be run as a long-running process. In UNIX, this may be done by running the license server in the background:

./licServer -url http://[host IP] &

The *host IP* specified is the host's IP address on which the license server runs.

The floating license server is itself a licensed application: it requires an *osyslic.txt* file in order to run. This file is provided by Objective Systems when the floating license server is purchased. Without the license, the server will fail to start.

The license server is not configured by default to be installed as a run-level service. As different UNIXes may treat run-level services differently, users are encouraged to consult their Operating System-specific documentation to determine how to do this if desired.

SDK and Runtime Installation

Installation of the various UNIX versions of ASN1C is similar to the Windows version except that the distribution files are packaged as gzipped tar files.

To install, do the following:

1. Copy the distribution file <distfile> to the top-level directory where the compiler is to be installed.

2. Unzip using the GNU unzip tool:

   gunzip <distfile>

3. Untar the file using the following command:

   tar xvf <distfile>

   This will create a directory tree structure with 'asn1c-v<version>' as the root.

4. You should have received a license file (asn1c.*lic* or *osyslic.txt*) or a license activation key on Linux 32-bit or MacOSX to enable the compiler to run on a given node or connect to a license server. If you have received the license file, go to step 5. The license activation key can be used in two ways:

   1) From a command line, use option '-lickey <license key>'

   2) From asn1c GUI, enter the key in the 'License key:' field:

Note: If there is a proxy server in your environment that the ASN1C is unaware of, you may see error message. You can use the web activation interface at http://obj-sys.com/rlm_activation.php. Enter the license key and your host ID which is the MAC address or physical address without the '-', for example 0019d1e7802f . The license will be generated and displayed in the web page. Save it into a license file with an extension .lic and copy it into one of the following locations listed in step 5.

5. You should have received a node-locked license file (*asn1c.lic* for Linux 32-bit or MacOSX, or *osyslic.txt* for other platforms) to enable the compiler on a given node. Copy this file into one of the following locations:

   a. One of the directories specified in your PATH environment variable, or

   b. In a different directory and create a new environment variable name OSLICDIR that points to this directory, or

   c. Into the same directory as the ASN1C compiler executable file (asn1c.exe)

To test if the compiler installation was successful, do the following:

1. Change directory (cd) to the compiler root directory. The default directory in the setup script is **./asn1c-v<version>** where **<version>** is the version number of the compiler. For example, **./asn1c-v650** is the default root directory for version 6.5.0 of the compiler.

2. Enter **./bin/asn1c** from the command line prompt. You should see a usage display of compiler command line options (these options are discussed later). This indicates the compiler is properly installed and working.

After installing the compiler, you can modify your operating environment to access the compiler executable files from anywhere. To do this, you will need to add the path to the compiler to your PATH environment variable or add a link to it from a standard binary directory (such as /opt/bin). This is optional and not required for any of the run-time sample programs to work. They are all set up to use relative directory paths to access the compiler and libraries.

Deployment run-time packages should be installed by repeating steps 1 through 3 above. The packages should be unpacked in the same base root directory as the original SDK files. This will cause *lib_opt* subdirectories to be added to the various C and C++ directories in the installation.

Testing the C or C++ Run-time Components

The basic C or C++ run-time libraries for UNIX are typically built with the GNU gcc/g++ compiler and/or the standard native compiler provided by the manufacturer of a particular type of UNIX (for example, aCC for HP-UX). Two symbolic links are used within the c or cpp subdirectory to select the version of the run-time libraries to be used. They are as follows:

- lib
- platform.mk

By default, these are set to point at the GNU gcc/g++ version of the run-time libraries for a particular platform. This is easily changed by deleting the links and setting them to point at another run-time library. For example, on Solaris, to use the native compiler libraries one would set lib -> libCC and platform.mk -> platform.CC:

```
ln –s libCC lib
ln –s platform.CC platform.mk
```

You can verify operation of any of the different run-time kits by executing the sample programs. These can be found in the different sample directories (*sample_ber*, *sample_der*, *sample_per* and/or *sample_xer* depending on what run-time kits were installed).

To test the encode/decode capabilities for any of the encoding rules, execute the sample programs as described in the section on Windows installation.

Compiling and Linking ASN1C Generated Code

When building code generated by the ASN1C compiler, you will need to have one or more run-time source directories in your include path to compile the generated code with a C or C++ compiler. The run-time source directories are *rtsrc* and *rtxsrc* (common), *rtbersrc* (BER/DER), *rtpersrc* (PER), *rtxersrc* (XER), and *rtxmlsrc* (XML). To link, you will need the *lib* or *lib_opt* subdirectory in your library path. This is where all of the library files are located.

Per-host License Deployment Issues

If you purchased run-time libraries that allow for unlimited redistribution, then all run-time license checking would have been removed from these libraries and all that must be done to deploy is to make certain that the code is linked with these libraries.

If per-host run-time licensing was selected, then there are two choices as to how applications are deployed:

1. The node-locked run-time license information can be directly compiled into the application and then the application deployed without any external files (this does not work for the floating license server), or

2. An external binary license file can be deployed with the application to allow it to run on the licensed hosts.

The first choice is done by default every time ASN.1 source files are generated and the resulting code compiled and linked into an application. Information from the *osyslic.txt* or the asb1c.lic file is transferred to a C header file called *rtkey.h* and this information is included in the generated code. This allows the application to run on all licensed hosts.

The second option is applicable mainly in situations when a finished application is to be run on different hosts than were originally licensed. This can happen if host names are changed, or additional hosts are added at a later date. It might not be practical in these situations to rebuild the application in this case. The alternative is therefore available to create a binary license file and deploy it with the application to allow it to run on the newly licensed hosts. The procedure to do this is as follows:

1. Use ASN1C to generate an *rtkey.dat* file by issuing the following command:

   asn1c -genlic

   The *rtkey.dat* file will be created in the directory where you issued the command.

2. Copy the *rtkey.dat* file onto the computers on which you want to run your application.

3. Set the *ACLICFILE* environment on these computers to point at the full path to the *rtkey.dat* file.

In the case of Java, the *rtkey.dat* file must be located somewhere on the classpath of the application.

Documentation Installation

Up-to-date documentation is always available online at: http://www.obj-sys.com/docs/documents.shtml

If a CD was purchased with the software, all applicable PDF document files for a particular configuration of the software will be available on the CD.

Backwards Compatibility

ASN1C underwent significant changes between version 5.8x and 6.0x, including some that affect code generation. Version 6.1 includes more modifications that affect Java and C# handling of enumeration types. In version 6.3, Java code generation was changed to embed the runtime license key in the generated sources in the same way that is done in C#.

These changes represent an API change for all languages. For C and C++, two files have been packaged with the kit to help in porting code generated with prior versions: rtport.pl and asn1compat.h. The former will process a specified file and rename the types contained in it. The latter contains macro definitions that convert old names to the new naming scheme.

Changes for Java and C# will be mostly transparent to the user, except that enumerated values are now retrieved through the use of accessor methods. These changes are discussed in detail in the Java and C# User's Guides. Please consult the language-specific user's manual for information on changes that may affect backwards compatibility when upgrading to new versions of the software.

In version 6.5.x, RLM is used to license the products on Windows, Linux and MacOSX. For other platforms, the osyslic.txt license file is still used.

Questions may be directed to support@obj-sys.com.