



APIs by Example: Cryptographic Key Management - Testing and Clearing Master Keys

[System iNetwork Programming Tips Newsletter](#)

[Carsten Flensburg](#)

Carsten Flensburg

Thu, 12/13/2007 (All day)

In a previous newsletter, I introduced a new suite of master key management APIs called the Cryptographic Services APIs and explained the Load Master Key Part (LODMSTKP) and Set Master Key (SETMSTK) commands, which provide interfaces to their identical API counterparts. This time around, I complete my introduction to master key management APIs with a discussion of the Test Master Key (TSTMSTK) and Clear Master Key (CLRMSTK) commands.

I also briefly discuss the technology available to establish the secure infrastructure required to keep the transmission of data between Telnet or iSeries Access terminal sessions and the System i away from prying eyes. This is particularly of interest when you perform cryptographic operations from a command line.

As I mentioned last time, V5R4 provides built-in support of cryptographic key management, so you need it to take advantage of the master key commands and APIs presented in this article.

Because this and other APIs by Example articles provides CL command interfaces to cryptographic functions and APIs, the infrastructure transmitting these commands and their parameters from a session terminal to the system that eventually processes them plays a vital role in establishing the security and confidentiality that by definition is required for cryptographic operations.

Often the terminal sessions are run from PCs using either the Telnet protocol or iSeries Access PC5250 terminal emulation product. In a traditional setup, all data submitted from such a terminal session flows unencrypted over the network to the System i server being connected to. The network can include both public segments and company internal routes that are monitored by both authorized and unauthorized parties, and actually, neither should be able to access this sensitive information.

To protect such data transmissions, Secure Socket Layer (SSL) technology offers some very interesting, readily available facilities to System i terminal clients. It's beyond the scope of this article to describe SSL in greater detail, but what it does, in essence, is provide a transparent layer of cryptographic protection for data being transmitted between a client and server. There are, however, some challenges related to the SSL configuration and the certificates required to make this happen.

So for the benefit of those wanting to research and approach this topic further, I thought I'd mention that I recently came across yet another fine Redbook that describes both the configuration and deployment issues of the SSL setup for Telnet as well as iSeries Access PC5250 emulation terminals. The Redbook is called *iSeries Access for Windows V5R2 Hot Topics: Tailored Images, Application Administration, SSL, and Kerberos*.

Chapter 2 focuses on the distribution and deployment issues, and chapter 4 documents the steps required to configure SSL for Telnet and iSeries Access PC5250 terminals. I provide links to this Redbook and to the sections on the V5R4 Infocenter documenting Telnet and iSeries Access PC5250 emulation terminals via SSL at the end of this article. The Redbook documents the aforementioned configuration at V5R2, but all the considerations and the screens involved still apply to V5R4, except for a few details documented in the more recent Redbook *IBM i5/OS Network Security Scenarios A Practical Approach*, chapter 7.

Let me begin my explanation of the new master key commands with the Clear Master Key (CLRMSTK) command. The CLRMSTK command allows you to clear either the new or old version of any specified master key. Note that the current version of a master key by definition is operational, and therefore cannot be cleared. Following a successful execution of the CLRMSTK command, the specified master key version will have all 32 of its bytes set to the hexadecimal value of x'00'. As mentioned, all 8 master keys can exist in the following three versions:

- New version -- this is the version that holds the master key parts loaded with the Load Master Key Part (LODMSTKP) command and API. When all parts are loaded and the Set Master Key (SETMSTK) command or API has been run, the new version is finalized and moved to the current version. At this point, the new version is set back to the hexadecimal value x'00', ready for a new update.
- Current version -- this is the currently active and operational version of a master key, and the one that is used by default if no key verification value (KVV) is submitted to a cryptographic operation (such as encrypt or decrypt data). The KVV is generated when the new master key version is promoted to the current master key version and, for the purpose hinted at above, should accompany the master key in cryptographic operations from there on.
- Old version -- this is the previously active but now retired version of a master key. It is kept as a safeguard until all keys or data encrypted under this version of the master key have been re-encrypted (translated) under the current version.

Submitting the KVV together with the master key to a cryptographic operation lets the system locate the right version of the master key to use for the operation:

- If the KVV belongs to the current master key version, the current version is used and the operation completes normally.
- If the KVV belongs to the old master key version, the old version is used, and a diagnostic message is issued both to the cryptographic service API in question and the QSYSOPR message queue. This is done to ensure that attention is drawn to the requirement of translating from the old to the current master key.
- If a submitted KVV belongs to neither the current nor old master key version, the cryptographic operation will fail.

Clearing the new version of a master key lets you re-enter the passphrases required to create the master key in case an error occurred or you are uncertain of the current state of the new master key. Clearing the old version of a master key is recommended once all keys encrypted under the old master key version have been translated (re-encrypted) under the new master key version. The CLRMSTK command prompt looks like this:



```

                                Clear Master Key (CLRMSTK)

Type choices, press Enter.

Master key ID . . . . . 1-8

Master key version . . . . . *NEW          *NEW, *OLD

```

If the CLRMSTK command runs successfully, a completion message is returned to the caller of the command to verify the outcome:

```

Message ID . . . . . : CBX0011      Severity . . . . . :
00
Message type . . . . . : Completion

Date sent . . . . . : 09-12-07      Time sent . . . . . :
13:04:58
Message . . . . . : Master key 8 version 2 has been successfully
cleared.
Cause . . . . . : The master key ID 8 version 2 was successfully
cleared.
The master key version may have the following values: 0 - New
version, 2 - Old
version.

```

Now on to the Test Master Key (TSTMSTK) command. The verb test in this conjunction points to the two main functions provided by this command (and API): It lets you verify the existence of the specified master key ID and version, and it provides a convenient option to retrieve the key verification value of any existing master key at a point in time following the actual generation of a master key. Here's the TSTMSTK command prompt:

```

                                Test Master Key (TSTMSTK)

Type choices, press Enter.

Master key ID . . . . . 1-8

Master key version . . . . . *CURRENT      *CURRENT, *OLD

Output . . . . . *          *, *PRINT, *STMF,
*MSG
Key verification value file . .

```

Specify the ID and version of the master key to test. The output parameter defines where the key verification value (KVV) associated with the new master key will be returned. The following options apply:

```
'*'      Displays the 20 byte KVV value in hexadecimal format in a
display panel. Use F11 to toggle
          between hexadecimal and character format of the KVV. From
the display panel function key F21
          allows you to print the information.

*PRINT    Prints the 20 byte KVV value in hexadecimal format to a
spooled file and places the file
          in the job's current default output queue. Due to the
formatting taking place the KVV will
          occupy 40 bytes.

*STMF     Writes the 20 byte KVV to stream file specified in the key
verification value file path.

*MSG      Returns the 20 byte KVV as message data in a completion
message sent to the caller of the
          command. The message will display the KVV in hexadecimal
format, while the message data
          remains unformatted.
```

Below is an example of how the display panel would look, following a successful testing of the current version of master key 1:

```

                                     Test Master Key

WYNDHAMW                                     09-12-07

12:20:08
Master key ID . . . . . :    1

Master key version . . . . :    *CURRENT

Key verification value . . :
AD98C57C67DC80DF87FABF381B246E183AB70922
Press Enter to continue

F3=Exit   F11=Display character KVV  F12=Cancel

F21=Print master key information

Master key 1 version 1 has been successfully tested.
```

The initial display shows the 20 byte key verification value formatted as 40 hex nibbles in order to display a recognizable value. Function key F11 enables you to toggle the KVV format between character and hexadecimal. Look up the command and display panel help text to learn about all the details.

The Test Master Key and Clear Master Key APIs both require *ALLOBJ and *SECADM special authority to run successfully. For this reason, the service program calling these APIs on behalf of the equivalent CL commands are configured to adopt user profile QSECOFR's authority. The CL commands are restricted from being run by unauthorized users by means of function usage authorization.

The CBX183M CL program provided to build the TSTMSTK and CLRMSTK commands will register a special and individual user function for the TSTMSTK and CLRMSTK commands, respectively. The user running the CBX183M CL program will be authorized to run both commands. Follow the instructions below to create all command objects correctly. Use the following command to locate and change the two function usage registrations in accordance with your requirements:

```
WRKFCNUSG FCNID(CBX_CRYPT0_MASTERKEY*)
```

If you loaded and installed the commands and usage registrations provided with this and the previous APIs by Example, the following screen should display, once you've run the above command:

```

                                Work with Function Usage

Type options, press Enter.

    2=Change usage    5=Display usage

Opt  Function ID                                Function Name

                                CBX_CRYPT0_MASTERKEY_CLEAR      Clear cryptographic master key
load                                CBX_CRYPT0_MASTERKEY_LOAD      Cryptographic master key part
                                CBX_CRYPT0_MASTERKEY_SET          Set cryptographic master key
                                CBX_CRYPT0_MASTERKEY_TEST          Cryptographic master key test

Bottom
Parameters for option 2 or command

===>

F3=Exit    F4=Prompt    F5=Refresh    F9=Retrieve    F12=Cancel
F17=Top
F18=Bottom

```

Use option 2 to add and/or remove users' function usage. The next section has more information on the function usage registration prerequisites and requirements.

Be sure that any person given access to the CLRMSTK command fully comprehends the scope and possible consequences of clearing the new or old version of a master key before he or she is set loose. Here's an excerpt from the command's help text that is intended to prevent mistakes from happening:

- Do not attempt to run this command unless you have been explicitly authorized to do so by the proper authority in your organization.
- Do not attempt to run this command unless you have received thorough instructions in performing this function.
- Failing to comply with the above recommendations could lead to a major loss of critical production data.

Also note that each of the Master Key APIs, Load Master Key Part, Set Master Key, Clear Master Key and Test Master Key, creates a security audit record of type CY with a detailed entry value of M (Master Key function). If you are interested in reading more about master keys, key management, and derived CL commands, check out upcoming issues of APIs by Example.

I'd also like to point you to a new IBM Redbook that is currently in a draft stage, but is already available at the link found at the end of this article. The Redbook is called "IBM System i Security: Protecting i5/OS Data with Encryption," and it provides a wealth of information as well as great code examples covering the very interesting and very challenging topic of cryptography in a practical context. Happy reading!

This APIs by Example includes the following sources:

```
CBX180  -- RPGLE  -- Cryptographic Key Management - Services
CBX180B -- SRVSRC -- Cryptographic Key Management - Binder source

CBX183  -- RPGLE  -- Test Master Key - CPP
CBX183E -- RPGLE  -- Test Master Key - UIM Exit Program
CBX183H -- PNLGRP -- Test Master Key - Help
CBX183P -- PNLGRP -- Test Master Key - Panel Group
CBX183V -- RPGLE  -- Test Master Key - VCP
CBX183X -- CMD    -- Test Master Key

CBX184  -- RPGLE  -- Clear Master Key - CPP
CBX184H -- PNLGRP -- Clear Master Key - Help
CBX184X -- CMD    -- Clear Master Key

CBX183M -- CLP    -- Cryptographic Key Management II - Build commands
```

To create all above objects, compile and run CBX183M. Compilation instructions are also found in the source headers as always. Note that the two previously published commands Add Function Registration (ADDFCNREG) and Change User Function Usage (CHGUSRFCNU) are required for the master key commands to run successfully – and the CBX183M program to compile.

The sources for the two aforementioned user function commands were included with my previous APIs by Example article of November 8. In case you missed that article, you can find it following the link below. Successfully compiling and running the CBX180M CL setup program included with that article is also prerequisite to running the CBX183M setup program included today.

Previously published related articles:

APIs by Example: Cryptographic Key Management - Loading and Setting Master Keys:

<http://www2.systeminetwork.com/article.cfm?id=55862>

Other related documentation:

IBM System i Security: Protecting i5/OS Data with Encryption (Redbook):

<http://www.redbooks.ibm.com/Redbooks.nsf/RedpieceAbstracts/sg247399.html?Open>

The above redbook is still at its draft status, the currently planned publish date is December 28, 2007.

Application Security with Secure Sockets Layer (SSL):

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/rzain/rzainsecapps.htm>

- Secure Sockets Layer (SSL) Administration (iSeries Access for Windows):

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/rzaii/rzaiissladm.htm>

- Telnet scenario: Securing Telnet with SSL:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/rzaiw/rzaiwscenariosl.htm>

iSeries Access for Windows V5R2 Hot Topics: Tailored Images, Application Administration, SSL, and Kerberos:

<http://www.redbooks.ibm.com/redbooks/pdfs/sg246939.pdf>

IBM i5/OS Network Security Scenarios A Practical Approach:

<http://www.redbooks.ibm.com/redpieces/abstracts/sg247374.html?Open>

This article demonstrates the following Cryptographic Services API:

Test Master Key Part (QcTestMasterKey) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/qc3tstmk.htm>

Clear Master Key (Qc3ClearMasterKey) API:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/qc3clrmk.htm>

Key Management APIs:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/catcrypt6.htm>

Cryptographic Services APIs:

<http://publib.boulder.ibm.com/infocenter/iseriess/v5r4/topic/apis/catcrypt.htm>

You can retrieve the source code for this API example from:

http://www.pentontech.com/IBMContent/Documents/article/56035_456_TestAndClear.zip.

Source URL: <http://iprodeveloper.com/rpg-programming/apis-example-cryptographic-key-management-testing-and-clearing-master-keys>