

[print](#) | [close](#)

## If You're Using Random Numbers, You Really Need This New Command

[System iNetwork Systems Management Newsletter](#)

[Carsten Flensburg](#)

Carsten Flensburg

Wed, 02/15/2006 (All day)

I've been exploiting the cryptographic capabilities of the i5/OS operating system in a number of utilities and business systems. One of the facilities involved is the Generate Pseudorandom Numbers API. For OPM, this is QC3GENRN, for ILE it is Qc3GenPRNs. The API basically returns a pseudorandom binary stream of the specified length, which then can be used as "salt" for a cryptographic encrypt or decrypt operation. For example, it can be used as a random number in a program function requiring a feature such as choosing 100 random customers for statement verification during a financial audit process.

A salt or initialization vector is an initial chaining value used in the encryption process to prevent the same clear text from producing the same cipher string. Which, of course, could compromise the confidentiality of the encrypted data.

You can find more information about the Generate Pseudorandom Numbers function [here](#).

To ensure that the Generate Pseudorandom Numbers API is using a truly unpredictable seed, which is essential to arrive at cryptographically secure pseudorandom numbers, it is very important the system seed digest is initialized correctly as described in the manual as follows.

"Cryptographically-secure pseudorandom numbers rely on good seed. The FIPS 186-1 key and seed values are obtained from the system seed digest. The server automatically generates seed using data collected from system information or by using the random number generator function on a cryptographic coprocessor, such as a 4758, if one is available. System-generated seed can never be truly unpredictable. If a cryptographic coprocessor is not available, you can use the Add Seed for Pseudorandom Number Generator (OPM, QC3ADDSD; ILE, Qc3AddPRNGSeed) API to add your own random seed to the system seed digest. **This should be done as soon as possible any time the Licensed Internal Code is installed.**"

To ease the process of initializing the system seed digest, I have written the Add Pseudorandom Number Seed (ADDPRNSEED) command, as shown here.

### Add Pseudorandom Number Seed (ADDPRNSEED)

Type choices, press Enter.

Seed block 1 . . . . .	01110110110101011110111011101101
Seed block 2 . . . . .	11101010101111001010010010100101
Seed block 3 . . . . .	00010101010100010100100110110010
Seed block 4 . . . . .	11010010010101010010010101110101
Seed block 5 . . . . .	00000101001001010010010010101001

---

For each seed block 1 through 5, specify the seed bits as a string of binary digits '0' and '1'. The string must have all 32 bits specified.

For more detail, refer to the help panel group.

The following source code is included. As always, check the source code headers for compile instructions and additional documentation.

CBX951	RPGLE	Add PRN seed - ILE RPG Source code
CBX951H	PNLGRP	Add PRN seed - Help Panel group
CBX951X	CMD	Add PRN seed - Command Definition

You can download a zip file containing all the source code from the URL

<http://www2.systeminetwork.com/noderesources/code/clubtechcode/addprn.zip>

**Note:** As with all new programs, test these routines thoroughly before placing them into a production environment. No warranty is expressed or implied.

**Source URL:** <http://iprodeveloper.com/application-development/if-youre-using-random-numbers-you-really-need-new-command>