Base Language - Feature #6421

finish GENERATE-RANDOM-KEY support

05/26/2022 03:30 PM - Greg Shah

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Theodoros Theodorou	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**version:**	
vendor_id:	GCD		

Description

History

#1 - 05/26/2022 03:32 PM - Greg Shah

Currently only AES_CBC_128 is supported.

Add and test support for all other algorithms which OE supports. We know that at least one customer application needs support for AES_OFB_128.

#2 - 01/13/2023 02:22 PM - Greg Shah

- Assignee set to Theodoros Theodorou

#3 - 01/16/2023 11:41 AM - Theodoros Theodorou

- Status changed from New to WIP

#4 - 01/16/2023 11:49 AM - Greg Shah

The runtime implementation of this is in SecurityOps.generateRandomKey(). There should be no conversion changes needed (the conversion occurs in rules/convert/builtin_functions.rules, search on prog.kw_gen_rndk to find where we define the conversion mapping.

#5 - 01/26/2023 08:52 AM - Theodoros Theodorou

I read the encryption algorithm by using SecurityPolicyManager.getSymmetricEncryptionAlgorithm().toStringMessage().

I checked the behaviour of our application and theirs and I noticed that we had a different result for the cases of ZZZ_ZZZ_56 and ZZZ_ZZZ_168.

Instead of 56 / 8 = 7 bytes, I had to use 8 bytes to match their results.
Instead of 168 / 8 = 21 bytes, I had to use 24 bytes to match their results.

So I modified the method SecurityOps.getAlgKeyLen() to return

```
return (int) Math.ceil(byteLen / 4d) * 4;
```

instead of

```
return byteLen;
```

After that, I checked my results with theirs and they looked the same.

I checked all the algorithms mentioned in [[

https://docs.progress.com/bundle/abl-reference/page/SYMMETRIC-SUPPORT-attribute.html#SYMMETRIC-SUPPORT-attribute]] and the results were the same.

The only algorithm that doesn't work is AES_GCM_* because it throws an exception.

Should I push my solution?

#6 - 01/26/2023 09:04 AM - Greg Shah

Create a task branch (6421a) and you can commit changes there.

#7 - 01/31/2023 02:05 AM - Theodoros Theodorou

I have created the branch 6421a

#8 - 01/31/2023 02:06 AM - Theodoros Theodorou

- % Done changed from 0 to 100

- Status changed from WIP to Review

I have committed my changes and the task is ready for review

#9 - 01/31/2023 05:41 AM - Greg Shah

Igor: Please review.

#10 - 01/31/2023 06:20 AM - Greg Shah

Code Review Task Branch 6421a Revision 14484

1. generateRandomKey() needs complete error handling. Please write test cases to see what failures can be created. I presume that they might come from:

- An invalid cipher being selected in SECURITY-POLICY:SYMMETRIC-ENCRYPTION.
- A cipher selected in SECURITY-POLICY:SYMMETRIC-ENCRYPTION which is valid in OpenEdge and invalid in Java.

2. getAlgKeyLen() needs complete error handling. Parsing failures will cause severe problems in the current code.

In both cases, we need to match the error handling behavior in the original 4GL environment, so testcases that show the issues are important. The result will be some kind of ErrorConditionException raised by ErrorManager.

3. The history entry is missing in the file.

#11 - 01/31/2023 09:03 AM - Theodoros Theodorou

How to write test cases? I thought that the p2j test cases do not run.

#12 - 01/31/2023 09:12 AM - Greg Shah

Theodoros Theodorou wrote:

> How to write test cases?

Writing 4GL Testcases
Testcase Projects (use the "new hotness")

> I thought that the p2j test cases do not run.

You write testcases in the 4GL and run them in OpenEdge, then observe the outputs/behavior. That let's you define the functionality that is our "specification".

Then you implement the feature in FWD, convert and run the testcases. The results should be the same or you are not done.

#13 - 01/31/2023 09:18 AM - Theodoros Theodorou

History entry means to add a comment in the beginning of the file?

Like the following?

```
** -#- -I- --Date-- --JPRM-- ----------------------------------Description--
...
**     TT  20230131          Finished the GENERATE-RANDOM-KEY support
```

#14 - 01/31/2023 09:30 AM - Greg Shah

Yes, except it would be useful to put slightly more detail in the comment. Something like this seems appropriate:

> generateRandomKey() now honors the cipher suite defined by the SECURITY-POLICY:SYMMETRIC-ENCRYPTION attribute instead of being hard coded to AES_CBC_128.

#15 - 02/01/2023 08:45 AM - Theodoros Theodorou

Do I need to checkout 6421a from the testcases project or use the main branch?

Then I should include my test-genrnd-key.p under security/security-policy/test-genrnd-key.p?

The above code stops if an error occurs. Should I make one big file or break it into many smaller files which will handle one algorithm at a time?

```
DEFINE VARIABLE i AS INTEGER NO-UNDO.
DEFINE VARIABLE strList AS CHARACTER NO-UNDO EXTENT 24.

ASSIGN strList[1] = "DES_CBC_56".
```

```
ASSIGN strList[2] = "DES_ECB_56".
ASSIGN strList[3] = "DES_OFB_56".
ASSIGN strList[4] = "DES_CFB_56".
ASSIGN strList[5] = "AES_CBC_128".
ASSIGN strList[6] = "AES_OFB_128".
ASSIGN strList[7] = "AES_CFB_128".
ASSIGN strList[8] = "RC4_ECB_128".
ASSIGN strList[9] = "AES_ECB_128".
ASSIGN strList[10] = "AES_GCM_128".
ASSIGN strList[11] = "DES3_CBC_168".
ASSIGN strList[12] = "DES3_CFB_168".
ASSIGN strList[13] = "DES3_ECB_168".
ASSIGN strList[14] = "DES3_OFB_168".
ASSIGN strList[15] = "AES_CBC_192".
ASSIGN strList[16] = "AES_OFB_192".
ASSIGN strList[17] = "AES_CFB_192".
ASSIGN strList[18] = "AES_GCM_192".
ASSIGN strList[19] = "AES_ECB_192".
ASSIGN strList[20] = "AES_CBC_256".
ASSIGN strList[21] = "AES_OFB_256".
ASSIGN strList[22] = "AES_CFB_256".
ASSIGN strList[23] = "AES_ECB_256".
ASSIGN strList[24] = "AES_GCM_256".

DO  i = 1 TO extent(strList):
  SECURITY-POLICY:SYMMETRIC-ENCRYPTION-ALGORITHM = strList[i].

  MESSAGE strList[i].
  MESSAGE GENERATE-RANDOM-KEY.
END.
```

#16 - 02/01/2023 12:26 PM - Greg Shah

> Do I need to checkout 6421a from the testcases project or use the main branch?

I'm not exactly sure what you are asking here. Presumably you are wanting to test your changes in 6421a. Testin trunk doesn't seem too useful as we know it only supports one cipher. If I understand correctly, then in the testcases project would should create a symbolic link (named p2j/) that points to your 6421a branch checkout.

> The above code stops if an error occurs. Should I make one big file or break it into many smaller files which will handle one algorithm at a time?

If you just add error handling, then it will work in the 4GL. Having an inner block labeled inner that uses ON ERROR, UNDO LEAVE inner can make it so that you can keep going on an error. When an error occurs you should check to see that it is matching your expectations of what the error number and message are. You do need to test this in the 4GL first. And it seems important to test values that will cause errors in the 4GL. I would try things that are purposely misformatted with spaces, unexpected _ chars or missing _ chars, hyphens instead of underscores, empty string, unknown value and so forth. I'd organize these into 2 arrays. The working ones and the failing ones.

For examples of this, see testcases/library_calls/.

#17 - 02/01/2023 12:45 PM - Igor Skornyakov

Greg Shah wrote:

> Igor: Please review.

Sorry, I do not understand. As far as I remember and can see from the code, we support all ciphers supported by 4GL for a long time (see #3810). It was a problem with encryption with short IV (for the modes that use IV), but I cannot recall any other problems regarding encyptions.
Have I missed something?
Thank you.

#18 - 02/01/2023 12:47 PM - Greg Shah

The original code in SecurityOps.generateRandomKey() was forced to AES_CBC_128. Take a look at trunk.

#19 - 02/01/2023 12:55 PM - Igor Skornyakov

Greg Shah wrote:

> The original code in SecurityOps.generateRandomKey() was forced to AES_CBC_128. Take a look at trunk.

Oh, I see. Sorry.

#20 - 02/01/2023 01:26 PM - Igor Skornyakov

Theodoros Theodorou wrote:

I read the encryption algorithm by using SecurityPolicyManager.getSymmetricEncryptionAlgorithm().toStringMessage().

I checked the behaviour of our application and theirs and I noticed that we had a different result for the cases of ZZZ_ZZZ_56 and ZZZ_ZZZ_168.

Instead of 56 / 8 = 7 bytes, I had to use 8 bytes to match their results.
Instead of 168 / 8 = 21 bytes, I had to use 24 bytes to match their results.

So I modified the method SecurityOps.getAlgKeyLen() to return

[...]

instead of

[...]

After that, I checked my results with theirs and they looked the same.

I checked all the algorithms mentioned in [[https://docs.progress.com/bundle/abl-reference/page/SYMMETRIC-SUPPORT-attribute.html#SYMMETRIC-SUPPORT-attribute]] and the results were the same.

The only algorithm that doesn't work is AES_GCM_* because it throws an exception.

Should I push my solution?

Indeed, the version of the SecurityOps.getAlgKeyLen() in the trunk provides incorrect results for DES and DES3. However I suggest using SecurityPlicyManager.getSymmetricCiperParams().getKeySize() instead of implementing non-obvious logic in two separate places. This will provide the key size for the current symmetric cipher. See CryptoUtils.CyperParams c'tor.

Apart of this, the changes look good.

#21 - 02/02/2023 10:41 AM - Theodoros Theodorou

Hello Igor and thank you for your review.

I replaced getAlgKeyLen() with SecurityPlicyManager.getSymmetricCiperParams().getKeySize() because as you stated because it is redundant. I also deleted getAlgKeyLen() because it is not used anywhere else and committed the changes under #6421a.

Is this test below a good test? Should I store it at testcases/security/security-policy/test-genrnd-key.p? What should I do next?

```
DEFINE VARIABLE i AS INTEGER NO-UNDO.
DEFINE VARIABLE list AS CHARACTER NO-UNDO EXTENT 24.

ASSIGN list[1] = "AES_CBC_128".
ASSIGN list[2] = "AES_CBC_192".
ASSIGN list[3] = "AES_CBC_256".
ASSIGN list[4] = "AES_CFB_128".
ASSIGN list[5] = "AES_CFB_192".
ASSIGN list[6] = "AES_CFB_256".
ASSIGN list[7] = "AES_ECB_128".
ASSIGN list[8] = "AES_ECB_192".
ASSIGN list[9] = "AES_ECB_256".
ASSIGN list[10] = "AES_GCM_128".
ASSIGN list[11] = "AES_GCM_192".
ASSIGN list[12] = "AES_GCM_256".
ASSIGN list[13] = "AES_OFB_128".
ASSIGN list[14] = "AES_OFB_192".
ASSIGN list[15] = "AES_OFB_256".
ASSIGN list[16] = "DES_CBC_56".
ASSIGN list[17] = "DES_CFB_56".
ASSIGN list[18] = "DES_ECB_56".
ASSIGN list[19] = "DES_OFB_56".
ASSIGN list[20] = "DES3_CBC_168".
ASSIGN list[21] = "DES3_CFB_168".
ASSIGN list[22] = "DES3_ECB_168".
ASSIGN list[23] = "DES3_OFB_168".
ASSIGN list[24] = "RC4_ECB_128".

DO i = 1 TO EXTENT(list):

  DO ON ERROR UNDO, LEAVE:

    SECURITY-POLICY:SYMMETRIC-ENCRYPTION-ALGORITHM = list[i].

    MESSAGE list[i].
    MESSAGE GENERATE-RANDOM-KEY.
  END.
END.
```

#22 - 02/02/2023 11:10 AM - Greg Shah

Do you have test code for the error cases that I mentioned in #6421-16?

In your testcase #6421-21, you should avoid interactive code (like MESSAGE) and instead write the code to detect non-interactively if the test passes or not. Then make sure it passes in both OpenEdge and FWD.

#23 - 02/02/2023 11:15 AM - Igor Skornyakov

Theodoros Theodorou wrote:

> Is this test below a good test? Should I store it at testcases/security/security-policy/test-genrnd-key.p?

I would add an encyption/decryption of the sample data using the generated key and comparing the decryption result with the original data.
Just looking at the generated keys does not look reliable - it is very easy to overlook some minor problems.

#24 - 02/06/2023 09:30 AM - Theodoros Theodorou

Please review the following test. The rest was developed and run on the AWS VM. I also included encryption/decryption tests. The length and prefix/suffix of the generated key (in string format) are also verified. I also added some purposely misformatted cases to produce and catch the errors.

```
{common/log_hlp.i}

function test-encode-decode returns logical  (encryptionKey as raw):
    def var textToEncrypt       as  longchar no-undo.
    def var encryptMptrResponse as  memptr   no-undo.
    def var decryptMptrResponse as  memptr   no-undo.
    def var decryptLongResponse as  longchar no-undo.

    textToEncrypt = 'Lorem ipsum'.
    encryptMptrResponse = encrypt(textToEncrypt, encryptionKey) no-error.
    decryptMptrResponse = decrypt(encryptMptrResponse, encryptionKey) no-error.

    copy-lob from decryptMptrResponse to decryptLongResponse no-error.
    return string(textToEncrypt) = string(decryptLongResponse).
end function.

function startsWith returns logical (s as character, prefix as character):
    return substr(s, 1, length(prefix)) = prefix.
end function.

function endsWith returns logical (s as character, suffix as character):
    return substr(s, length(s) - length(suffix) + 1) = suffix.
end function.

def var i             as int       no-undo.
def var s             as character no-undo.
def var encryptionKey as raw       no-undo.

run log-it('generate-random-key : type - start').

&scoped-define list_working_56 "DES_CBC_56,DES_CFB_56,DES_ECB_56,DES_OFB_56"

do i=1 to num-entries({&list_working_56}):
  security-policy:symmetric-encryption-algorithm = entry(i, {&list_working_56}).
  encryptionKey = generate-random-key.
  s = string(encryptionKey).

  assert-true(startsWith(s, '020008'), 'check prefix',  {&line-number}).
  assert-true(endsWith(s, '='), 'check suffix',  {&line-number}).
```

```
  assert-true(length(s) = 18, 'check length',  {&line-number}).
  assert-true(test-encode-decode(encryptionKey), 'check if the key can encrypt and decrypt', {&line-number}).
end.

&scoped-define list_working_128 "AES_CBC_128,AES_OFB_128,AES_CFB_128,RC4_ECB_128,AES_ECB_128"

do i=1 to num-entries({&list_working_128}):
  security-policy:symmetric-encryption-algorithm = entry(i, {&list_working_128}).
  encryptionKey = generate-random-key.
  s = string(encryptionKey).

  assert-true(startsWith(s, '020010'), 'check prefix',  {&line-number}).
  assert-true(endsWith(s, '=='), 'check suffix',  {&line-number}).
  assert-true(length(s) = 30, 'check length',  {&line-number}).
  assert-true(test-encode-decode(encryptionKey), 'check if the key can encrypt and decrypt', {&line-number}).
end.

&scoped-define list_working_168 "DES3_CBC_168,DES3_CFB_168,DES3_ECB_168,DES3_OFB_168"

do i=1 to num-entries({&list_working_168}):
  security-policy:symmetric-encryption-algorithm = entry(i, {&list_working_168}).
  encryptionKey = generate-random-key.
  s = string(encryptionKey).

  assert-true(startsWith(s, '020018'), 'check prefix',  {&line-number}).
  assert-true(length(s) = 38, 'check length',  {&line-number}).
  assert-true(test-encode-decode(encryptionKey), 'check if the key can encrypt and decrypt', {&line-number}).
end.

&scoped-define list_working_192 "AES_CBC_192,AES_CFB_192,AES_ECB_192,AES_OFB_192"

do i=1 to num-entries({&list_working_192}):
  security-policy:symmetric-encryption-algorithm = entry(i, {&list_working_192}).
  encryptionKey = generate-random-key.
  s = string(encryptionKey).

  assert-true(startsWith(s, '020018'), 'check prefix',  {&line-number}).
  assert-true(length(s) = 38, 'check length',  {&line-number}).
  assert-true(test-encode-decode(encryptionKey), 'check if the key can encrypt and decrypt', {&line-number}).
end.

&scoped-define list_working_256 "AES_CBC_256,AES_CFB_256,AES_ECB_256,AES_OFB_256"

do i=1 to num-entries({&list_working_256}):
  security-policy:symmetric-encryption-algorithm = entry(i, {&list_working_256}).
  encryptionKey = generate-random-key.
  s = string(encryptionKey).

  assert-true(startsWith(s, '020020'), 'check prefix',  {&line-number}).
  assert-true(endsWith(s, '='), 'check suffix',  {&line-number}).
  assert-true(length(s) = 50, 'check length',  {&line-number}).
  assert-true(test-encode-decode(encryptionKey), 'check if the key can encrypt and decrypt', {&line-number}).
end.

&scoped-define list_AES_GCM "AES_GCM_128,AES_GCM_192,AES_GCM_256"

do i=1 to num-entries({&list_AES_GCM}):
  security-policy:symmetric-encryption-algorithm = entry(i, {&list_AES_GCM}) no-error.
  assert-err-num(12223, substitute({&msg_attr_set_invalid}, "SYMMETRIC-ENCRYPTION-ALGORITHM"), {&line-number})
.
end.

&scoped-define list "DES_CBC_83,_DES_CBC_56,DES_CBC__56,DES_CBC56,DES_CBC 56,DES-CBC-56,test, "

do i=1 to num-entries({&list}):
  security-policy:symmetric-encryption-algorithm = entry(i, {&list}) no-error.
  assert-err-num(12223, substitute({&msg_attr_set_invalid}, "SYMMETRIC-ENCRYPTION-ALGORITHM"), {&line-number})
.
end.

run log-it('generate-random-key : type - end').
```

#25 - 02/06/2023 09:56 AM - Greg Shah

I think it is good. Just add a test for unknown value. You can't do that one with a preprocessor define, so just hard code it. The the code will need some tweaks to move to ABLUnit but that can wait until 3827a is merged to trunk.

How does FWD respond to the code? I suspect some changes are needed.

#26 - 02/07/2023 02:33 AM - Theodoros Theodorou

By unknown value, you mean non-ASCII characters? I wasn't able to run the test with FWD yet. I will try to run it today and I will let you know.

#27 - 02/07/2023 08:04 AM - Greg Shah

No, I am referring to the Progress 4GL concept of unknown value (which can be represented with the literal constant ?). This is similar but not the same as null in Java. Please read up about unknown value in the 4GL docs.

#28 - 02/07/2023 10:33 AM - Theodoros Theodorou

I need help running the test with FWD. I tried some things but it didn't work. My project tree is:

```
~/projects/p2j
~/projects/hotel_gui
~/projects/testcases
```

The commands I use to run the test are:

```
rm -rf ~/projects/hotel_gui/abl/*
echo $'{log_hlp.i}\n' > ~/projects/hotel_gui/abl/test-genrnd-key.p
cp /home/tt/projects/testcases/common/* ~/projects/hotel_gui/abl

sed -i 's/db.h2=false/db.h2=true/g' ~/projects/hotel_gui/build.properties
sed -i 's/db.postgresql=true/db.h2=false/g' ~/projects/hotel_gui/build.properties
sed -i 's/com.goldencode.hotel.Start.execute/com.goldencode.hotel.TestGenrndKey.execute/g' ~/projects/hotel_gu
i/deploy/server/directory.xml
cd ~/projects/hotel_gui
ant deploy.all  < /dev/null

cd ~/projects/hotel_gui/deploy/server/
./server.sh
```

To run the client I use:

```
cd ~/projects/hotel_gui/deploy/client/
./client.sh client:cmd-line-option:startup-procedure=./test-genrnd-key.p
```

With the above script I try to load the dependency log_hlp.i but I keep getting the following error in a java swing window saying ** "./test-genrnd-key.p" was not found. (293).

I would really appreciate a complete script that can execute my test using FWD if you have some spare time.

#29 - 02/07/2023 11:51 AM - Greg Shah

> rm -rf ~/projects/hotel_gui/abl/*

I think you should not do this, in general. Try ant clean.all instead.

```
sed -i 's/db.h2=false/db.h2=true/g' ~/projects/hotel_gui/build.properties
sed -i 's/db.postgresql=true/db.h2=false/g' ~/projects/hotel_gui/build.properties
```

These changes should actually be the default in Hotel GUI. Please make the edits and commit the change.

> ./client.sh client:cmd-line-option:startup-procedure=./test-genrnd-key.p

Did you try abl/test-genrnd-key.p or ./abl/test-genrnd-key.p? You can also modify the propath in the directory.xml but I think that is not needed if you add abl/ to the relative path.

#30 - 02/08/2023 12:33 PM - Theodoros Theodorou

> > sed ...
>
> These changes should actually be the default in Hotel GUI. Please make the edits and commit the change.

I agree. I pushed the change into the trunk.

I decided to follow the guide found in https://proj.goldencode.com/projects/p2j/wiki/Testcases#Conversion-Configuration-and-Runtime-details again. After some tries, I was able to set it up! I executed the test and it looks like it was successful!

Now I should copy my test-genrnd-key.p under testcases/security/security_policy/test-genrnd-key.p and push it to trunk, right?

#31 - 02/08/2023 01:06 PM - Greg Shah

Yes. At this time we don't branch testcases, so we have no trunk concept there. Instead of saying pushing to trunk for testcases, I would just say "committing to testcases". The two projects we use branching for are FWD and our internal version of H2. Right now all other projects don't have enough activity to justify the more complex branching setup.

Did you add the unknown value test code?

> After some tries, I was able to set it up!

Great!

> I executed the test and it looks like it was successful!

By this, you mean that all the FWD behavior was the same as OpenEdge?

#32 - 02/09/2023 04:01 AM - Theodoros Theodorou

> Did you add the unknown value test code?

I added this:

```
security-policy:symmetric-encryption-algorithm = ? no-error.
assert-err-num(12223, substitute({&msg_attr_set_invalid}, "SYMMETRIC-ENCRYPTION-ALGORITHM"), {&line-number}).
```

> > I executed the test and it looks like it was successful!

> By this, you mean that all the FWD behavior was the same as OpenEdge?

Yes, it behaves the same and it catches the same error codes. I also tried modifying the error codes to see if it works and it seems ok. Please test it as well if you have time to make sure it is correct. You can find the test under security/secutiry_policy/test-genrnd-key.p.

#33 - 02/09/2023 09:55 AM - Theodoros Theodorou

- Status changed from Review to Feedback

#34 - 02/15/2023 10:42 AM - Theodoros Theodorou

I have recreated the branch 6421a after the branch 6129c was merged to trunk and committed all changes. Please review it so we can merge it because the ticket #6399 depends on this to be able to pass the tests.

#35 - 02/15/2023 01:44 PM - Greg Shah

- Status changed from Feedback to Review

Igor: Please review.

#36 - 02/15/2023 01:45 PM - Greg Shah

Code Review Task Branch 6421a Revision 14485

1. Didn't you have other changes needed to make the error handling correct?

2. Did you retest all of the 4GL testcases?

3. The copyright date needs updating to 2023.

#37 - 02/15/2023 02:21 PM - Igor Skornyakov

Greg Shah wrote:

> Igor: Please review.

The changes look good.

#38 - 02/16/2023 03:25 AM - Theodoros Theodorou

> 1. Didn't you have other changes needed to make the error handling correct?

In the beginning I modified the method SecurityPolicy.getAlgKeyLen() (#6421-5) to behave the same with FWD. After Igor's suggestion (#6421-20) to use SecurityPlicyManager.getSymmetricCiperParams().getKeySize() the method getAlgKeyLen() was redundant so I completely deleted it.

> 2. Did you retest all of the 4GL testcases?

Yes, I executed the test under security/security_policy/test-genrnd-key.p again and it passed.

> 3. The copyright date needs updating to 2023.

Sorry, I forgot to update that. I pushed the change today.

#39 - 02/16/2023 07:17 AM - Greg Shah

- Status changed from Review to Test

#40 - 03/23/2023 12:39 PM - Theodoros Theodorou

I noticed that in my branch 6421a I have accidentally pushed changes from another task and, therefore I recreated the branch based on the latest version of trunk. I followed the rebase process (although it was not needed).

I didn't notice that I had to ask for a rebase review and I proceeded with merging it. 6421a was merged to trunk as revision 14509 and is now archived. The only file which has changes is SecurityOps.java. Please review it.

#41 - 03/23/2023 12:42 PM - Greg Shah

- Status changed from Test to Review

I've uncommitted rev 14509 from trunk. Let's follow the normal process.

#42 - 03/23/2023 01:49 PM - Theodoros Theodorou

6421a has been rebased to trunk 14508

#43 - 03/23/2023 01:53 PM - Constantin Asofiei

SecurityOps is missing a history number, like this:

```
** 032 TT  20230215          generateRandomKey() now works correctly with the *_56 and *_168 algorithms
```

#44 - 03/23/2023 01:58 PM - Igor Skornyakov

Constantin Asofiei wrote:

> SecurityOps is missing a history number, like this:
> [...]

I do believe that saying just with the DES and DES3 ciphers is less confusing and more accurate.

#45 - 03/23/2023 01:59 PM - Constantin Asofiei

Igor Skornyakov wrote:

> Constantin Asofiei wrote:

> SecurityOps is missing a history number, like this:
> [...]

> I do believe that saying just with the DES and DES3 ciphers is less confusing and more accurate.

OK, please fix the the history message, and also use GENERATE-RANDOM-KEY instead of 'generateRandomKey()'

#46 - 03/23/2023 02:18 PM - Theodoros Theodorou

Done

#47 - 03/23/2023 02:22 PM - Constantin Asofiei

Thanks, go ahead with merging 6421a to trunk. Please follow the merge steps in #7027-48

#48 - 03/24/2023 10:00 AM - Theodoros Theodorou

6421a was merged to trunk as revision 14509. It was archived.

#49 - 03/27/2023 03:36 AM - Greg Shah

- Status changed from Review to Closed