
From: Danilo Gligoroski <danilog@ntnu.no>
Sent: Thursday, December 28, 2017 1:00 AM
To: pqc-comments
Cc: pqc-forum@list.nist.gov
Subject: OFFICIAL COMMENT: LAKE

Dear LAKE designers,

Can you give a short explanation why the ciphertext in LAKE KAT files appears so regularly non-random and in most cases it has as a sub-string "0000....0000" (a string with 18 zeros for LAKE-I, a string with 24 zeros for LAKE-II and a string with 28 zeros for LAKE-III)? The number of zeroes (18 or 24 or 28) is in connection with the parameter m in your design, but why those strings appear so frequently in the ciphertexts? I have tried to find some note in your documentation, but I could not find any.

For example for LAKE-I let us look at examples count = 7 and count = 13:

```
count = 7
seed =
AEFB28FDD34E0AB403A703B535296E3A545CA479C1D8148E2D501B3C8DD8B1034BD986F13F1A7B4671BE769359FD2AAB
pk =
1A4D09D9EA87345B008099585032D68EF9006C9A694A051B738600EFBAC9DD42AD43290212F5C3D10BE8129401575B11
B470B3EF720515731B40A35CC5FE07EE525237CFB0D468078B5F07A63B2D9C8F0488653C52331624E0043A66DEF377C6
14C303CD759FBF16C8AEA50551FDC6F1E58B837B059A0CA7E1F40AC50E05E67F9B9DDCED0E9007CE023ECEEDB95D8F00
AD5A3F6F03158901027B00EA65E3ACCD3A009FA639BE49B41C5D078F28ED83A7BD2587055D9724AED5BF05980584574F
BD1C50A584073A2E0D004B39087F07884B97E35791CA1C03D8215BF50BF6CE7506FCBE5C04B73955BD000E27C2904FF4
82C605D295B5BA5B6CCEA00003FC0FA175B339E50500FC0EAA1F7C69E506CCE432CC1654ABF500DD7B76855B2BEAA605
9D99C85E516BCD9606D37ED19C68676535007AB9FA7EBE3D5773005A5B7188CB1334AD065D25A41EC0A989D601109785
C86D6466D506956CEAC109A7A37E054FB7B28467534F33063995F0D282D3CABD0703CAE4EA27C0739707FC7EC8D22F03
9DA60137CCE90028BD361906D201BE1A6A26BB9B00A04CE16B3117CE8D04CA9600ED3DFFC8FE03
sk = D86634ECF96CC2603761E284C0E36734CEDEC64E7FF486469E38539C71141C5A0472C3D61E4CAE91
ct =
97938F14C48EBC2505D903B47FAAE20F9C03A20909F01D4A825401121D4F636F94673501BD6B48CFDA0443C501121D4F
636F9467350138E588B8711E98D5054E903B6B6E6CB3B90643EF3537C6B6151D07D903B47FAAE20F9C03CB1EFB1CC576
68A902D47CBA230238A93802C661F5406DACCE0D0388F1CE2B03C07DB405E1E63CC7DBFC974906F3FB73A4B468F07C07
AF7607ACB59024F000F3FB73A4B468F07C077B0ABD8FB7A88DC80297938F14C48EBC2505D47CBA230238A938022AF8C7
DB1E8AFFE004F3FB73A4B468F07C07F3FB73A4B468F07C07CB1EFB1CC57668A90297938F14C48EBC2505121D4F636F94
673501D903B47FAAE20F9C037675B3D31F722B6C03B014469372DEE561001F62413FC74EC1910038E588B8711E98D505
88F1CE2B03C07DB405C661F5406DACCE0D0338E588B8711E98D5057B0ABD8FB7A88DC802359A86E4D9C43E710443EF35
37C6B6151D07C661F5406DACCE0D032787C987B650594405FE847DF81CB256D80651F27A54A92272280688F1CE2B03C0
7DB405EC99329B732631ED076468FCB070E64C59020D7F0E5CA8DAA6A40143EF3537C6B6151D07
ss =
C6CD973D27C8701EBE23FEE9620DC9E7F7799A800F3102974E2ACF71449A6A18C2EFDB393074EFE10FE7FD45CA456463D5B23
D922793F64FE619A7A9C4609672
```

If we see the ciphertext output for count = 7 it seems fairly randomly distributed.
This is what I would expect for all other KATs.

From: Danilo Gligoroski <danilog@ntnu.no>
Sent: Thursday, December 28, 2017 6:28 AM
To: pqc-comments
Cc: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] OFFICIAL COMMENT: LAKE

More about the non-randomness in LAKE ciphertexts:

I must correct myself that the count = 7 KAT looks fairly randomly distributed. It is also pretty non-random. If we represent ct in count = 7 in chunks of 18 nibbles we have the following formatting:

```
ct = {  
"97938F14C48EBC2505", "D903B47FAAE20F9C03", "A20909F01D4A825401", \  
"121D4F636F94673501", "BD6B48CFDA0443C501", "121D4F636F94673501", \  
"38E588B8711E98D505", "4E903B6B6E6CB3B906", "43EF3537C6B6151D07", \  
"D903B47FAAE20F9C03", "CB1EFB1CC57668A902", "D47CBA230238A93802", \  
"C661F5406DACCE0D03", "88F1CE2B03C07DB405", "E1E63CC7DBFC974906", \  
"F3FB73A4B468F07C07", "AF7607ACB59024F000", "F3FB73A4B468F07C07", \  
"7B0ABD8FB7A88DC802", "97938F14C48EBC2505", "D47CBA230238A93802", \  
"2AF8C7DB1E8AFFE004", "F3FB73A4B468F07C07", "F3FB73A4B468F07C07", \  
"CB1EFB1CC57668A902", "97938F14C48EBC2505", "121D4F636F94673501", \  
"D903B47FAAE20F9C03", "7675B3D31F722B6C03", "B014469372DEE56100", \  
"1F62413FC74EC19100", "38E588B8711E98D505", "88F1CE2B03C07DB405", \  
"C661F5406DACCE0D03", "38E588B8711E98D505", "7B0ABD8FB7A88DC802", \  
"359A86E4D9C43E7104", "43EF3537C6B6151D07", "C661F5406DACCE0D03", \  
"2787C987B650594405", "FE847DF81CB256D806", "51F27A54A922722806", \  
"88F1CE2B03C07DB405", "EC99329B732631ED07", "6468FCB070E64C5902", \  
"0D7F0E5CA8DAA6A401", "43EF3537C6B6151D07"  
}
```

and we can see that in the list of 47 sub-strings in ct, there are
"only" 27 different sub-strings

```
{  
"0D7F0E5CA8DAA6A401", "121D4F636F94673501", "1F62413FC74EC19100", \  
"2787C987B650594405", "2AF8C7DB1E8AFFE004", "359A86E4D9C43E7104", \  
"38E588B8711E98D505", "43EF3537C6B6151D07", "4E903B6B6E6CB3B906", \  
"51F27A54A922722806", "6468FCB070E64C5902", "7675B3D31F722B6C03", \  
"7B0ABD8FB7A88DC802", "88F1CE2B03C07DB405", "97938F14C48EBC2505", \  
"A20909F01D4A825401", "AF7607ACB59024F000", "B014469372DEE56100", \  
"BD6B48CFDA0443C501", "C661F5406DACCE0D03", "CB1EFB1CC57668A902", \  
"D47CBA230238A93802", "D903B47FAAE20F9C03", "E1E63CC7DBFC974906", \  
"EC99329B732631ED07", "F3FB73A4B468F07C07", "FE847DF81CB256D806"  
}
```

Will this non-randomness affect the security claims of LAKE?

Regards,

Danilo!

On 28/12/2017 06:59, Danilo Gligoroski wrote:

Dear LAKE designers,

Can you give a short explanation why the ciphertext in LAKE KAT files appears so regularly non-random and

in most cases it has as a sub-string "0000....0000" (a string with 18 zeros for LAKE-I, a string with 24 zeros for LAKE-II and a string with 28 zeros for LAKE-III)? The number of zeroes (18 or 24 or 28) is in connection with the parameter m in your design, but why those strings appear so frequently in the ciphertexts?

I have tried to find some note in your documentation, but I could not find any.

For example for LAKE-I let us look at examples $\text{count} = 7$ and $\text{count} = 13$:

$\text{count} = 7$

seed =

AEFB28FDD34E0AB403A703B535296E3A545CA479C1D8148E2D501B3C8DD8B1034BD986F13F1A7B467
1BE769359FD2AAB

pk =

1A4D09D9EA87345B008099585032D68EF9006C9A694A051B738600EFBAC9DD42AD43290212F5C3D10
BE8129401575B11

B470B3EF720515731B40A35CC5FE07EE525237CFB0D468078B5F07A63B2D9C8F0488653C52331624E00
43A66DEF377C6

14C303CD759FBF16C8AEA50551FDC6F1E58B837B059A0CA7E1F40AC50E05E67F9B9DDCED0E9007CE02
3ECEEDB95D8F00

AD5A3F6F03158901027B00EA65E3ACCD3A009FA639BE49B41C5D078F28ED83A7BD2587055D9724AED
5BF05980584574F

BD1C50A584073A2E0D004B39087F07884B97E35791CA1C03D8215BF50BF6CE7506FCBE5C04B73955BD
000E27C2904FF4

82C605D295B5BA5B6CCEA00003FC0FA175B339E50500FC0EAA1F7C69E506CCE432CC1654ABF500DD7B
76855B2BEAA605

9D99C85E516BCD9606D37ED19C68676535007AB9FA7EBE3D5773005A5B7188CB1334AD065D25A41EC
0A989D601109785

C86D6466D506956CEAC109A7A37E054FB7B28467534F33063995F0D282D3CABD0703CAE4EA27C0739
707FC7EC8D22F03

9DA60137CCE90028BD361906D201BE1A6A26BB9B00A04CE16B3117CE8D04CA9600ED3DFFC8FE03

sk =

D86634ECF96CC2603761E284C0E36734CEDEC64E7FF486469E38539C71141C5A0472C3D61E4CAE91

ct =

97938F14C48EBC2505D903B47FAAE20F9C03A20909F01D4A825401121D4F636F94673501BD6B48CFDA
0443C501121D4F

636F9467350138E588B8711E98D5054E903B6B6E6CB3B90643EF3537C6B6151D07D903B47FAAE20F9C
03CB1EFB1CC576

68A902D47CBA230238A93802C661F5406DACCE0D0388F1CE2B03C07DB405E1E63CC7DBFC974906F3FB
73A4B468F07C07

AF7607ACB59024F000F3FB73A4B468F07C077B0ABD8FB7A88DC80297938F14C48EBC2505D47CBA2302

From: Nicolas Aragon <nicolas.aragon@etu.unilim.fr>
Sent: Thursday, December 28, 2017 6:42 AM
To: Danilo Gligoroski; pqc-comments
Cc: pqc-forum@list.nist.gov
Subject: RE: [pqc-forum] OFFICIAL COMMENT: LAKE

Dear Danilo,
Thanks for pointing out this issue.

This problem comes from an error in the implementation of LAKE, we are currently working on a fix. Ciphertexts are indeed supposed to be randomly distributed

We will provide updated KATs when we have fixed the implementation

Regards,
Nicolas Aragon

De : Danilo Gligoroski <daniilog@ntnu.no>
Envoyé : jeudi 28 décembre 2017 12:27:53
À : pqc-comments@nist.gov
Cc : pqc-forum@list.nist.gov
Objet : Re: [pqc-forum] OFFICIAL COMMENT: LAKE

More about the non-randomness in LAKE ciphertexts:
I must correct myself that the count = 7 KAT looks fairly randomly distributed. It is also pretty non-random. If we represent ct in count = 7 in chunks of 18 nibbles we have the following formatting:

```
ct = {  
"97938F14C48EBC2505", "D903B47FAAE20F9C03", "A20909F01D4A825401", \  
"121D4F636F94673501", "BD6B48CFDA0443C501", "121D4F636F94673501", \  
"38E588B8711E98D505", "4E903B6B6E6CB3B906", "43EF3537C6B6151D07", \  
"D903B47FAAE20F9C03", "CB1EFB1CC57668A902", "D47CBA230238A93802", \  
"C661F5406DACCE0D03", "88F1CE2B03C07DB405", "E1E63CC7DBFC974906", \  
"F3FB73A4B468F07C07", "AF7607ACB59024F000", "F3FB73A4B468F07C07", \  
"7B0ABD8FB7A88DC802", "97938F14C48EBC2505", "D47CBA230238A93802", \  
"2AF8C7DB1E8AFFE004", "F3FB73A4B468F07C07", "F3FB73A4B468F07C07", \  
"CB1EFB1CC57668A902", "97938F14C48EBC2505", "121D4F636F94673501", \  
"D903B47FAAE20F9C03", "7675B3D31F722B6C03", "B014469372DEE56100", \  
"1F62413FC74EC19100", "38E588B8711E98D505", "88F1CE2B03C07DB405", \  
"C661F5406DACCE0D03", "38E588B8711E98D505", "7B0ABD8FB7A88DC802", \  
"359A86E4D9C43E7104", "43EF3537C6B6151D07", "C661F5406DACCE0D03", \  
"2787C987B650594405", "FE847DF81CB256D806", "51F27A54A922722806", \  
"88F1CE2B03C07DB405", "EC99329B732631ED07", "6468FCB070E64C5902", \  
"0D7F0E5CA8DAA6A401", "43EF3537C6B6151D07"  
}
```

and we can see that in the list of 47 sub-strings in ct, there are

From: Gaborit <gaborit@unilim.fr>
Sent: Saturday, December 30, 2017 9:20 PM
To: Danilo Gligoroski
Cc: pqc-forum@list.nist.gov
Subject: Fwd: RE: [pqc-forum] OFFICIAL COMMENT: LAKE

Dear Danilo,

thank you for your remarks, this is the same minor implementation problem
than for LAKE, which does not modify the inherent theoretical security of the proposals
nor the protocols in themselves, nor the parameters and performances,
the fixed reference implementation and KAT can be found at:

LAKE : <http://filex.unilim.fr/get?k=PoPX0bJhkzfQfiW97HA>
LOCKER : <http://filex.unilim.fr/get?k=2hYbKuJeKDrc2m6D1HI>

best,

the LAKE and LOCKER team.

--

You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
Visit this group at <https://groups.google.com/a/list.nist.gov/group/pqc-forum/>.

From: Danilo Gligoroski <danilog@ntnu.no>
Sent: Sunday, January 07, 2018 6:30 PM
To: pqc-comments
Cc: pqc-forum@list.nist.gov
Subject: OFFICIAL COMMENT: LAKE

Dear LAKE and LOCKER designers,

I found the following 2 properties of your schemes that are not discussed in your documentation. For the sake of short presentation I will describe the procedures for LAKE, but the same properties hold for LOCKER too.

1. For a given pair (pk, sk) it is easy for everyone to produce millions of new equivalent public keys pk_1, pk_2, \dots such that $K = \text{Decap}(sk, \text{Encap}(pk)) = \text{Decap}(sk, \text{Encap}(pk_1)) = \text{Decap}(sk, \text{Encap}(pk_2)) = \dots$ 2. For a given ciphertext $c = e_1 + e_2 \cdot h$ it is easy for everyone who captures c , to produce millions of equivalent ciphertexts c_1, c_2, \dots , such that $K = \text{Decap}(sk, c) = \text{Decap}(sk, c_1) = \text{Decap}(sk, c_2) = \dots$

The procedure for producing these equivalent public keys and ciphertexts is quite simple:

1. Let B^n be the n -dimensional vector space $\langle 0, 1 \rangle^n$ (which is a subspace of $F^n_{\{2^m\}}$) 2. Generate a random vector $s \in B^n$, such that s is invertible mod P 3. Produce an equivalent $pk_1 = s \cdot pk$ or equivalent $c_1 = s \cdot c$

The equivalent pk_1 and c_1 have the properties 1 and 2.

The correctness of the above procedure comes from a trivial property of B^n : For any vector space E , $E \cdot B^n = B^n \cdot E = E$.

In my opinion, the first property can be seen as an advantage of the scheme, since it gives some possibilities to anonymise the public key, but it is in conjunction with the second property. And the second property is quite unusual for a public key scheme, since it gives opportunity to the adversaries to scramble the ciphertext, but the decrypting party will not notice that. Maybe that is not a serious problem for an IND-CPA scheme, but still it is quite unusual for a public key scheme.

Best regards,
Danilo!

P.S. My guess is that similar properties hold for the patented scheme RQC, but once I read that it is patented I stop my analysis for it.

From: Olivier Blazy <gloupin@gmail.com>
Sent: Monday, January 08, 2018 5:13 AM
To: pqc-forum
Cc: pqc-comments; danilog@ntnu.no
Subject: Re: OFFICIAL COMMENT: LAKE

Thank you Danilo for your remark. It is indeed an interesting property and indubitably a strength of our proposals.

As for your comments concerning the decryption, we'd like to point out that you do not seem to have taken into account the various steps of the

HHK transform that prevents this kind of attack.

The basic PKE scheme that we propose is indeed proven IND-CPA and not IND-CCA2, it is however just the inner block to be used in

the HHK transform. (Which allows to generically transform IND-CPA schemes into IND-CCA one).

In some ways, what you say is equivalent to saying that ElGamal is malleable on the keys, however it does not mean that Cramer Shoup is.

Same thing goes for HQC. Now yes HQC is patented but as NIST proposes there will be a free general license on the protocol, as it is written in the document.

Best,

The LAKE, LOCKER and HQC teams.