Dear authors, deal all,

The current specification (and implementation) of LEDAkem seems to fail to achieve CCA security.
LEDAkem tries to construct an IND-CCA-secure KEM by applying the conversion in [30] to a OW-CPA-secure deterministic PKE.
The authors would not notice the chosen-ciphertext attacks in [A1] and [A3, Appendix K] against KEM/Hybrid PKE in [30].

LEDAkem
==========
* The public key is $M$ in $\mathbb{F}_2^{p \times n}$.
* The encapsulation algorithm chooses $e \in \mathbb{F}_2^n$ with $\text{HW}(e) = t$, and outputs a ciphertext $s = M e^T$ and a session key $K = \text{KDF}(e)$.
* The decapsulation algorithm recovers $e$ from $s$ by using the secret key and outputs $k_s = \text{KDF}(e)$. If $s$ is invalid, the decapsulation algorithm returns a "pseudorandom" key $k_s = \text{KDF}(s)$.

The footnote 1 of [30] suggests $k_s = \text{KDF}(s)$, which is not pseudorandom.

Chosen-Ciphertext Attack against the current LEDAkem ========= The following CCA exists even if the scheme is perfectly correct. See [A1] and [A3, Appendix K].
For $i = 0, \ldots, n-1$, let $u_i$ be the i-th unit vector of dimension $n$.

* Assume that a ciphertext $s = M e^T$ is given and assume that $e[0] = 0$.
* For $i = 1, \ldots, n-1$, we query $s_i = s + M \{u_0 + u_i\}^T$ and obtain the result.
* Set $e[i] = 0$ if $k_s = \text{KDF}(s_i)$; else set $e[i] = 1$.
* Compute $K = \text{KDF}(e)$

If $e[i] = 1$, then $\text{HW}(e + u_0 + u_i) = t$. On the other hand, if $e[i] = 0$, then $\text{HW}(e + u_0 + u_i) = t + 2 > t$.
This breaks the onewayness of KEM.

Note
=======
If DFR is 0, it is easy to fix the problem.

* Persichetti's thesis suggests to use $\text{KDF}(s')$, where $s' = l_{\{n_0-1\}^{\{-1\}}} s$ in the LEDAkem context.
* [A1] and [A2] suggests to use $\text{KDF}(\pi(s))$, where $\pi$ is a random permutation. Notice that this $\pi$ should be pseudorandom. Otherwise, one can still check if a ciphertext is valid or invalid by checking the answer is random or deterministic.
* [HHK17] and [SXY17] suggests to use $\text{Hash}(\text{secret-seed}, s)$ (or $\text{KDF}(\text{secret-seed}, s)$).
[30]: Edoardo Persichetti:
"Secure and Anonymous Hybrid Encryption from Coding Theory" in PQCrypto 2013
[A1]: Pierre-Louis Cayrel, Cheikh Thiecoumba Gueye, El Hadji Modou Mboup, Ousmane Ndiaye, and Edoardo Persichetti:
"Efficient Implementation of Hybrid Encryption from Coding Theory" in C2SI 2017
[A2]: Edoardo Persichetti:
"Code-based Key Encapsulation from McEliece's Cryptosystem" in MACIS2017
[A3]: Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, Christine van Vredendaal:
(http://eprint.iacr.org/2016/461)

Regards,
Keita Xagawa
Dear Keita,

thank you for the observations provided and the clarity in pointing them out.

In the LEDAkem supporting documentation Sect. 2.4, we claimed that LEDAkem provides the NIST required IND-CPA security and that it is possible to adapt it to achieve IND-CCA security employing a KDF to hide the case where a decoding failure (in your observation, induced by the attacker) takes place.

We acknowledge that the KDF to provide such a guarantee requires the addition of a secret bitstring of some kind as a parameter to the KDF, as pointed out by Persichetti’s thesis, [A1], [A2] (can you provide references for [HHK17] and [SXY17] to us privately?), and this is not pointed out in the specification and reference implementation.

While the presence of a secret bitstring has no impact on the IND-CPA security of the scheme in a case where ephemeral keys are used, it provides a useful resiliency feature in case of accidental key reuse, in addition to allowing to achieve IND-CCA.

We are thus planning to provide an up-to-date specification document including this clarification, and adapt the implementation accordingly on our website as soon as possible.

Kind Regards,
--the LEDAkem team

Il 01/10/18 05:42, Keita Xagawa ha scritto:
> Dear authors, deal all,
> > The current specification (and implementation) of LEDAkem seems to
> > fail to achieve CCA security.
> > LEDAkem tries to construct an IND-CCA-secure KEM by applying the
> > conversion in [30] to a OW-CPA-secure deterministic PKE.
> > The authors would not notice the chosen-ciphertext attacks in [A1] and
> > [A3, Appendix K] against KEM/Hybrid PKE in [30].
> > > LEDAkem
> > ========
> > * The public key is M in \( F_{2^n} \times \).
> > * The encapsulation algorithm chooses \( e \in F_{2^n} \) with \( HW(e) = t \), and
> > outputs a ciphertext \( s = Me^T \) and a session key \( K = KDF(e) \).
> > * The decapsulation algorithm recovers \( e \) from \( s \) by using the secret
> > key and outputs \( k_s = KDF(e) \). If \( s \) is invalid, the decapsulation
> > algorithm returns a "pseudorandom" key \( k_s = KDF(s) \).
> > > The footnote 1 of [30] suggests \( k_s = KDF(s) \), which is not pseudorandom.
> > >
> > > Chosen-Ciphertext Attack against the current LEDAkem ========= The
Dear all,

we prepared a document as an official comment on LEDA kem and LEDAPkc (attached to this message and available at the url reported below) in which:

** We provide our quantification of the quantum and classic computational effort levels we considered as the computational requirements to break AES. We relied on classical circuit design estimates for the classical computing complexity, and on the work by Grassl et al. at PQCrypto 2016 for the quantum computing complexity.

** We delineate an automatic procedure to find an optimal set of parameters for LEDA kem/LEDAPkc matching the aforementioned security margin. We pair this procedure with the release of the sources of a parameter computation tool for LEDA kem/LEDAPkc. The sources are hereby placed in the public domain, and we welcome contributions and suggestions.

** Our approach to the estimation of the computational effort required to perform Information Set Decoding attacks was to evaluate their complexity in the finite regime (as opposed to employing asymptotic bounds), and to perform an exhaustive search in the parameter space of the algorithms.

We report that, concerning the parameter sizes of the LEDA cryptosystems, the advantages offered by more recent ISD algorithms over the proposal by Stern in 1988 are lower than a factor of 2^4.

** We report new running time and key size figures for the optimal parameter sets, showing a x3.5--x6.8 speedup on the reference implementation and a ~x2 key size reduction w.r.t. the submission parameters.

** We propose a novel technique allowing us to design a set of QC-LDPC code parameters for use in LEDA kem/LEDAPkc deriving an upper bound to the code DFR in closed-form. This allows to include a bound on the DFR as a parameter design criterion.

** We report sample sets of parameters targeting an upper bound for the DFR of 2^-64 for long term keys in LEDAPkc.

The full document is both attached to this message, and hosted at [https://www.ledacrypt.org/archives/official_comment.pdf](https://www.ledacrypt.org/archives/official_comment.pdf)

The public domain software implementation of the automated procedure for the design of tight and optimal sets of parameters for the LEDA cryptosystems is available at [https://github.com/LEDAcrypt/LEDAtools](https://github.com/LEDAcrypt/LEDAtools)

At the same address, we also provide a software tool to compute the complexity of the considered ISD attacks, given a set of code parameters.

The header files containing the revised parameter sets which tightly match the security requirements are both hosted at
https://www.ledacrypt.org/archives/new_parameter_headerfiles.zip, and available on our github repositories, tagged as version 1.1.0 of the codebase.

The revised codebase also includes the appropriate modifications to cope with an artificially higher number of errors being inserted in the message (i.e. a check for the number of errors has been added) and the modifications suggested in the previous official comments.

Best regards,
--LEDA team