Limonnitsa: making Limonnik-3 post-quantum (with isogenies)

Sergey Grebnev

TC 26

Classic Diffie-Hellman

Diffie-Hellman-Merkle, 1976

$x, A = g^x$

$\xrightarrow{A}$

$\xleftarrow{B}$

$y, B = g^y$

$B^x = g^{xy} = A^y$

Limonnik-3

Introduced in 2014, officially accepted in 2017.

- Built upon MTI/A0, KEA+C ideas.
- Two ephemeral-to-static DH.
- Uses (optionally) two distinct elliptic curves.
- UKS- and KCI-secure.
- Security argument by reduction to GDHP.

Limonnik-3

$A:$	$k_A \in_R [1, q_B - 1]$
$A \to B$	$\mathtt{Id}_A, \mathtt{Cert}_A, k_A P_B$
$B:$	$k_B \in_R [1, q_A - 1], Q = c_A k_B S_A, R = c_B s_B k_A P_B$
	$K \parallel M = \mathtt{kdf}(\pi(Q), \pi(R), \mathtt{Id}_A \parallel \mathtt{Id}_B[\parallel OI])$
	$tag_B = \mathtt{mac}_M(h_2, k_B P_A, k_A P_B, \mathtt{Id}_B, \mathtt{Id}_A)$
$B \to A$	$\mathtt{Id}_B, \mathtt{Cert}_B, k_B P_A, tag_B$
$A:$	$Q = c_A s_A k_B P_A, R = c_B k_A S_B$
	$K \parallel M = \mathtt{kdf}(\pi(Q), \pi(R), \mathtt{Id}_A \parallel \mathtt{Id}_B[\parallel OI])$
	If $tag_B \neq \mathtt{mac}_M(h_2, k_B P_A, k_A P_B, \mathtt{Id}_B, \mathtt{Id}_A)$, terminates the session with an error
	$tag_A = \mathtt{mac}_M(h_3, k_A P_B, k_B P_A, \mathtt{Id}_A, \mathtt{Id}_B)$
$A \to B$	tag_A
$B:$	If $tag_A \neq \mathtt{mac}_M(h_3, k_A P_B, k_B P_A, \mathtt{Id}_A, \mathtt{Id}_B)$, terminates the session with an error

Limonnik-3 and quantum threat

Limonnik-3 is not quantum-secure.

Classical security: $O(\sqrt{\min(q_A, q_B)})$ by Pollard's ρ.
Quantum security: $O(\ln^2 \min(q_A, q_B))$ by Schor's method.

Can we replace the basic Diffie-Hellman key exchange by a post-quantum primitive?
Consider Supersingular Isogeny Diffie-Hellman (L. De Feo, D. Jao, J. Plût, 2011-2014).

$$\begin{array}{ccc} E & \xrightarrow{\varphi} & E/\langle P\rangle \\ {\scriptstyle\psi}\downarrow & & \downarrow \\ E/\langle Q\rangle & \longrightarrow & E/\langle P, Q\rangle \end{array}$$

SIDH

Public parameters: $p = l_A^{e_A} l_B^{e_B} \cdot f \pm 1$, l_A, l_B – distinct small primes, $(l_A, f) = (l_B, f) = 1$, a supersingular elliptic curve $E_0(GF(p^2))$ and bases $\{P_A, Q_A\}$ и $\{P_B, Q_B\}$, generating, resp., $E_0[l_A^{e_A}]$ and $E_0[l_B^{e_B}]$, that is, $\langle P_A, Q_A \rangle = E_0[l_A^{e_A}]$ and $\langle P_B, Q_B \rangle = E_0[l_B^{e_B}]$.

A chooses $n_A \in_R \mathbb{Z}/l_A^{e_A}\mathbb{Z}$, constructs $\varphi_A : E_0 \to E_A$ with the kernel $K_A := \langle P_A + [n_A]Q_A \rangle$. A also computes вычисляет образ $\{\varphi_A(P_B), \varphi_A(Q_B)\}$ and sends them to B together with E_A.

Having received from B the tuple $E_B, \varphi_B(P_B), \varphi_B(Q_B)$, A constructs $\varphi'_A : E_B \to E_{AB}$ with the kernel $\langle \varphi_B(P_A) + [n_A]\varphi_B(Q_A) \rangle$.

B proceeds simultaneously. The secret key is the j-invariant of

$$E_{AB} = \varphi' B(\varphi_A(E_0)) = \varphi'_A(\varphi_B(E_0)) = E_0/\left\langle P_A + [n_A]Q_A, P_B + [n_B]Q_B \right\rangle .$$

Introducing Limonnitsa

A post-quantum version of Limonnik-3.

- Built upon Limonnik-3 structure.
- Two ephemeral-to-static SIDH.
- Uses (optionally) two distinct parameters sets.
- UKS- and KCI-secure.
- Security argument by weaker reduction to SSDHP.

Limonnitsa

Fix public parameters for the parties A and B.

- $p_A = 2^{e_{a2}} 3^{e_{a3}} - 1$,
- $E_{A0}(GF(p^2))$;
- linearly independent points $P_{A2}, Q_{A2} \in E_{A0}[2^{e_{a2}}]$ (that is, $|\langle P_{A2}, Q_{A2} \rangle| = 2^{2e_{a2}}$)

For the party B, we have:

- $p_B = 2^{e_{b2}} 3^{e_{b3}} - 1$,
- $E_{B0}(GF(p_B^2))$;
- linearly independent points $P_{B2}, Q_{B2} \in E_{B0}[2^{e_{b2}}]$ (that is, $|\langle P_{B2}, Q_{B2} \rangle| = 2^{2e_{b2}}$)

Limonnitsa

Now, the party A selects its secret static key as an integer s_A such that $0 < s_A < 2^{e_{a2}}$, constructs the isogeny $\varphi_A : E_A \to E_A / \langle P_{A2} + [s_A]Q_{A2} \rangle$, calculates $E_A = E_{A0} / \langle P_{A2} + [s_A]Q_{A2} \rangle$, $P_A = \varphi_A(P_{A3})$, $Q_A = \varphi_A(Q_{A3})$, sets its static public key to $\{E_A, P_A, Q_A\}$, and acquires a certificate $\texttt{Cert}_A$.

B selects its static key as an integer s_B such that $0 < s_B < 2^{e_{b2}}$, constructs the isogeny $\varphi_B : E_B \to E_B / \langle P_{B2} + [s_B]Q_{B2} \rangle$, calculates $E_B = E_{B0} / \langle P_{B2} + [s_B]Q_{B2} \rangle$, $P_B = \varphi_B(P_{B3})$, $Q_B = \varphi_B(Q_{B3})$, sets is static public key as $\{E_B, P_B, Q_B\}$, and acquires a certificate $\texttt{Cert}_B$ as well.

Limonnitsa

$A:$	$k_A \in_R [1, 3^{e_{b3}}], S_{AB} = P_{B3} + [k_A]Q_{B3},$
	$\varphi_{AB} : E_B \to E_B / \langle S_{AB} \rangle$ – an isogeny with the kernel $\langle S_{AB} \rangle$
	$E_{AB} = E_{B0} / \langle S_{AB} \rangle$ (that is, $E_{AB} = \varphi_{AB}(E_{B0})$)
	$\mathcal{K}_A = \{E'_A, \varphi_{AB}(P_{B2}), \varphi_{AB}(Q_{B2})\}$ – A's ephemeral public key
$A \to B$	$\mathtt{Id}_A, \mathtt{Cert}_A, \mathcal{K}_A$
$B:$	$k_B \in_R [1, 3^{e_{a3}}], S_{BA} = P_{A3} + [k_B]Q_{A3},$
	$\varphi_{BA} : E_A \to E'_B / \langle S_{BA} \rangle$ – an isogeny with the kernel $\langle S_{BA} \rangle$
	$E_{BA} = E_{A0} / \langle S_{BA} \rangle$ (that is, $E_{BA} = \varphi_{BA}(E_{A0})$)
	$\mathcal{K}_B = \{E'_B, \varphi_{BA}(P_{A2}), \varphi_{BA}(Q_{A2})\}$ – B's session public key
	$T_{AB} = P_A + [k_B]Q_A$
	$T'_{AB} = \varphi_{AB}(P_{B2}) + [s_B]\varphi_{AB}(Q_{B2})$
	$\psi_{AB} : E'_A \to E'_A / \langle T_{AB} \rangle$ – an isogeny with the kernel $\langle T_{AB} \rangle$
	$\psi'_{AB} : E_B \to E_B / \langle T'_{AB} \rangle$ – an isogeny with the kernel $\langle T'_{AB} \rangle$
	$E_{AB} = \psi_{AB}(E'_A); E'_{AB} = \psi'_{AB}(E_B)$
	$K \parallel M = \mathtt{kdf}(j(E_{AB}) \parallel j(E'_{AB}) \parallel \mathtt{Id}_A \parallel \mathtt{Id}_B [\parallel OI])$
	$tag_B = \mathtt{mac}_M(h_2, \mathcal{K}_B, \mathcal{K}_A, \mathtt{Id}_B, \mathtt{Id}_A)$
$B \to A$	$\mathtt{Id}_B, \mathtt{Cert}_B, \mathcal{K}_B, tag_B$

Limonnitsa

$A:$	$T_{BA} = \varphi_{BA}(P_{A2}) + [s_A]\varphi_{BA}(Q_{A2})$
	$T'_{BA} = P_B + [k_A]Q_B$
	$\psi'_{BA} : E'_B \to E'_B / \langle T_{BA} \rangle$ – an isogeny with the kernel $\langle T_{BA} \rangle$
	$\psi_{BA} : E_A \to E_A / \langle T'_{BA} \rangle$ – an isogeny with the kernel $\langle T'_{BA} \rangle$
	$E'_{BA} = \psi'_{BA}(E'_B); E_{BA} = \psi'_{AB}(E_A)$
	$K \parallel M = \mathtt{kdf}(j(E'_{BA}) \parallel j(E_{BA}) \parallel \mathtt{Id}_A \parallel \mathtt{Id}_B[\parallel OI])$
	If $tag_B \neq \mathtt{mac}_M(h_2, \mathcal{K}_B, \mathcal{K}_A, \mathtt{Id}_B, \mathtt{Id}_A)$, terminates the session with an error
	$tag_A = \mathtt{mac}_M(h_3, \mathcal{K}_A, \mathcal{K}_B, \mathtt{Id}_A, \mathtt{Id}_B)$
$A \to B$	tag_A
$B:$	If $tag_A \neq \mathtt{mac}_M(h_3, \mathcal{K}_A, \mathcal{K}_B, \mathtt{Id}_A, \mathtt{Id}_B)$, terminates the session with an error

Security properties

- Secret key recovery: $\sqrt[3]{p}$ quantum and $\sqrt{p}$ classical.
- Parties' authentication: PKI.
- UKS-attacks: by tags structure similar to Limonnik-3.
- KCI-атаки: immune by the basic design of MTI/A0.
- Forward secrecy: for A, B, but not for A and B.
- Parameters: 902-bit prime, to keep up with Kuznyechik.

Problem 1.

Computational isogeny Diffie-Hellman, SSCDH: let $\varphi_A : E_0 \to E_A$ – an isogeny with kernel $\langle P_A + [n_A]Q_A \rangle$, and $\varphi_B : E_0 \to E_B$ – an isogeny with kernel $\langle P_B + [n_B]Q_B \rangle$, where n_A is chosen uniformly randomly from $\mathbb{Z}/l_A^{e_A}\mathbb{Z}$ and n_B is chosen uniformly randomly from $\mathbb{Z}/l_B^{e_B}\mathbb{Z}$. Given E_A, E_B and the images $\varphi_A(P_B), \varphi_A(Q_B), \varphi_B(P_A), \varphi_B(Q_A)$, find the j-invariant of the curve $E_0/\langle P_A + [n_A]Q_A, P_B + [n_B]Q_B \rangle$.

Security reductions

Problem 2.

Decisional isogeny Diffie-Hellman, SSDDH: Given a tuple sampled with probability $1/2$ from one of the following two distributions

- $(E_A, E_B, \varphi_A(P_B), \varphi_A(Q_B), \varphi_B(P_A), \varphi_B(Q_A), E_{AB})$, *where* $(E_A, E_B, \varphi_A(P_B), \varphi_A(Q_B), \varphi_B(P_A), \varphi_B(Q_A)$ – *as before,* $E_{AB} \cong E_0/\langle P_A + Q_A, [m]P_B + [n]Q_B\rangle$;
- $(E_A, E_B, \varphi_A(P_B), \varphi_A(Q_B), \varphi_B(P_A), \varphi_B(Q_A), E_C)$, *where* $(E_A, E_B, \varphi_A(P_B), \varphi_A(Q_B), \varphi_B(P_A), \varphi_B(Q_A)$ – *as before, and* $E_C \cong E_0/\langle P_A + [n']Q_A, P_B + [n']Q_B\rangle$ *where* m', n' *are chosen at random from from* $\mathbb{Z}/l_B^{e_B}\mathbb{Z}$*);*

determine from which distribution the tuple is sampled.

Security reductions II

We state now a weaker version of the security definition. We allow an adversary M to perform any of the following queries.

- Initiate a session between any chosen parties.
- Send messages from a party to another, which is followed by a correct (prescribed by the protocol) response.
- Execute a correct session between any chosen parties.
- Corrupt a party (that is, to learn any secret keys, as well as all generated shared keys and any local state information).

Security reductions II

Note that *M* cannot perform any Reveal queries.
Define as $\Lambda(n)$ the set of all Limonnitsa public parameters for a chosen security parameter *n*: that is, all primes of an appropriate form with bit-lentgh *n*, all possible supersingular elliptic curves defined over those primes.

Security reductions II

Definition 3.

A key agreement protocol is said to be weak-AKE-secure if the following conditions hold:

1. If two honest parties complete matching sessions then, except with negligible probability, they both compute the same session key.
2. No polynomially bounded adversary M defined above can distinguish the session key of a fresh session from a randomly chosen session key with probability greater than $1/2$ plus a negligible fraction.

Security reductions III

Theorem 4.

Let the SSDDH problem for Λ be computationally hard. Let `kdf` *be modelled by a pseudorandom function, let* `mac` *be secure against forgery attack. Then Limonnitsa is secure in the sense of Definition 3.*

Security reductions III

Proof (sketch).

The proof repeats the analogous results for Limonnik-3 (Grebnev, 2014-2019) in a weaker security model.

- We introduce L-2, a reduced version of Limonnitsa, by removing authentication tags and replacing `kdf` by a hash function.
- We consider the only possibility for an adversary to break L-2: that is, to solve the SSCDH.
- We show that there exists a polynomial-time algortith with success probability

$$\frac{1}{n^2 k} Pr[Success(\mathcal{M})],$$

  where $Pr[Success(\mathcal{M})]$ is the probability that $\mathcal{M}$ breaks weak AKE-secutity of L-2.
- We use then the UC-property to show that Limonnitsa is secure.

Attacks against static keys

The security model does not cover adaptive attacks by Galbraith, Petit, Shani, and Ti.

Suppose B has a static public key $E_B = E/\langle P_B + [\beta]Q_B\rangle$. Let φ_X be A's isogeny, $R = \varphi_X(P_B)$, $S = \varphi_X(Q_B)$. Suppose A knows K_i, $0 < K_i < l_2^i$, such that $\beta = K_i + l_2^i z$, let z_0 be guess for $z \pmod{l_2}$. The attack is to choose $R' = R + [-l_2^{m-1-i}K_i - l_2^{m-1}z_0]S$ and $S' = [1 + l_2^{m-i-1}]S$ and send $\{E_X, R', S'\}$ to B.

B computes

$$R' + [\beta]S = \cdots = (R + [\beta]S) + [(z - z_0)l_2^{m-1}]S,$$

the resulting kernel is correct iff $z \equiv z_0 \pmod{l_2}$.

After $(l_2 - 1)e_2$ sessions, the secret key is recovered.

Public key validation

We use Kirkwood's trick to counter this attack.
Instead of choosing random ephemeral secret key k_A, the party A chooses a single random seed $r_A \in V^*$ and uses a pseudo-random function $\texttt{prf}$ to output $k_A = \texttt{prf}(r_A)$. Then, tag_A is calculated as $tag_A = \texttt{encrypt}_M(h_2, r_A, K_A, K_B, \texttt{Id}_A, \texttt{Id}_B)$. The party B, having calculated the session key, recovers the seed r_A and repeats A's computations in order to verify that the keys were constructed as prescribed, otherwise, terminates the session. The parties B and A proceed vice versa.

Limonnitsa++

$A:$ $c_A \in_R V^*, k_A = H(s_A), S_{AB} = P_{B3} + [k_A]Q_{B3},$
$\varphi_{AB} : E_B \to E_B / \langle S_{AB} \rangle$ – an isogeny with the kernel $\langle S_{AB} \rangle$
$E_{AB} = E_{B0} / \langle S_{AB} \rangle$ (that is, $E_{AB} = \varphi_{AB}(E_{B0})$)
$\mathcal{K}_A = \{E'_A, \varphi_{AB}(P_{B2}), \varphi_{AB}(Q_{B2})\}$ – A's ephemeral public key

$A \to B$ $\mathtt{Id}_A, \mathtt{Cert}_A, \mathcal{K}_A$

$B:$ $s_B \in_R V^*[1, 3^{e_{a3}}], k_B = H(s_B), S_{BA} = P_{A3} + [k_B]Q_{A3},$
$\varphi_{BA} : E_A \to E'_B / \langle S_{BA} \rangle$ – an isogeny with the kernel $\langle S_{BA} \rangle$
$E_{BA} = E_{A0} / \langle S_{BA} \rangle$ (that is, $E_{BA} = \varphi_{BA}(E_{A0})$)
$\mathcal{K}_B = \{E'_B, \varphi_{BA}(P_{A2}), \varphi_{BA}(Q_{A2})\}$ – B's session public key
$T_{AB} = P_A + [k_B]Q_A$
$T'_{AB} = \varphi_{AB}(P_{B2}) + [s_B]\varphi_{AB}(Q_{B2})$
$\psi_{AB} : E'_A \to E'_A / \langle T_{AB} \rangle$ – an isogeny with the kernel $\langle T_{AB} \rangle$
$\psi'_{AB} : E_B \to E_B / \langle T'_{AB} \rangle$ – an isogeny with the kernel $\langle T'_{AB} \rangle$
$E_{AB} = \psi_{AB}(E'_A); E'_{AB} = \psi'_{AB}(E_B)$
$K \parallel M = \mathtt{kdf}(j(E_{AB}) \parallel j(E'_{AB}) \parallel \mathtt{Id}_A \parallel \mathtt{Id}_B[\parallel OI])$
$tag_B = \mathtt{encrypt}_M(h_2, c_B, \mathcal{K}_B, \mathcal{K}_A, \mathtt{Id}_B, \mathtt{Id}_A)$

$B \to A$ $\mathtt{Id}_B, \mathtt{Cert}_B, \mathcal{K}_B, tag_B$

Limonnitsa

$A:$	$T_{BA} = \varphi_{BA}(P_{A2}) + [s_A]\varphi_{BA}(Q_{A2})$
	$T'_{BA} = P_B + [k_A]Q_B$
	$\psi'_{BA} : E'_B \to E'_B / \langle T_{BA} \rangle$ – an isogeny with the kernel $\langle T_{BA} \rangle$
	$\psi_{BA} : E_A \to E_A / \langle T'_{BA} \rangle$ – an isogeny with the kernel $\langle T'_{BA} \rangle$
	$E'_{BA} = \psi'_{BA}(E'_B); E_{BA} = \psi'_{AB}(E_A)$
	$K \parallel M = \mathtt{kdf}(j(E'_{BA}) \parallel j(E_{BA}) \parallel \mathtt{Id}_A \parallel \mathtt{Id}_B[\parallel OI])$
	decrypts $c'_B, k'_B = H(c'_B)$, computes φ'_{BA}
	– an isogeny with the kernel $P_{A3} + [k'_B]Q_{A3}$,
	if $\varphi'_{BA}(P_A 2) \neq \varphi_{BA}(P_A 2)$ or $\varphi'_{BA}(Q_A 2) \neq \varphi_{BA}(Q_A 2)$,
	sets $K \parallel M = \mathtt{kdf}(j(E'_{BA}) \parallel n \parallel \mathtt{Id}_A \parallel \mathtt{Id}_B[\parallel OI]), n \in_R [1, p_B^2]$
	$tag_A = \mathtt{encrypt}_M(h_3, c_A, \mathcal{K}_A, \mathcal{K}_B, \mathtt{Id}_A, \mathtt{Id}_B)$
$A \to B$	tag_A
$B:$	decrypts $c_A, k_A = H(c_A)$, computes $\varphi' AB$
	– an isogeny with the kernel $P_{B3} + [k'_B]Q_{B3}$,
	if $\varphi'_{BA}(P_A 2) \neq \varphi_{BA}(P_A 2)$ or $\varphi'_{BA}(Q_A 2) \neq \varphi_{BA}(Q_A 2)$,
	sets $K \parallel M = \mathtt{kdf}(n \parallel j(E'_{AB}) \parallel \mathtt{Id}_A \parallel \mathtt{Id}_B[\parallel OI]), n \in_R [1, p_A^2]$

Thank you

Thanks for your attention.

grebnev_sv@tc26.ru