A Variant of Lindell17

Jianhong He, Wenping Ma, Jiajia Yang

1 Introduction

We introduce a variant of the Lindell 17 [1], denoted as **Lindell (+)**. The main difference from Lindell 17 is that the private key (kept completely unknown to any individual party) can be obtained by combining the private key shares of the two parties through addition. Specifically, if x_1 is the private key share held by party P_1 and x_2 is the private key share held by party P_2, the private key x is computed as:

$$x = (x_1 + x_2) \pmod q$$

where q is the order of the elliptic curve.

To ensure the security of **Lindell17(+)**, we introduced several improvements:

- Construct a new zero-knowledge protocol denoted as $ZK - L'_{PDL}$ based on the existing zero-knowledge protocol $ZK - L_{PDL}$.
- Optimized and revised the original KeyGen and Signing protocols to ensure the security and reliability of the protocol.
- Completed the security proof for this variant.

Paper Organization. In Section 2, we describe the details of the proposed variant *Lindell17(+)*. Section 3 focuses on the construction and details of the new zero-knowledge protocol, ZK-L'_{PDL}. In Sections 4 and 5, we present the comprehensive security proof for **Lindell17(+)**, ensuring its robustness and correctness.

2 A Variant of Lindell17 — Lindell17(+)

The **Lindell17(+)** is almost identical to the original protocol, with the differences highlighted in red for clarity.

2.1 Key Generation

Protocol 1: Key Generation Subprotocol $\mathsf{KeyGen}(\mathcal{G}, g, q)$

Given the common joint input $(\mathbb{G}, G, q)$ and security parameter 1^n, work as follows:

1. ***P_1*'s first message:**
   (a) P_1 chooses a random $x_1 \leftarrow \mathbb{Z}_q$, and computes $Q_1 = x_1 \cdot G$.
   (b) P_1 sends $(\mathsf{com\text{-}prove}, 1, Q_1, x_1)$ to $\mathcal{F}^{R_{DL}}_{\mathsf{com\text{-}zk}}$ (P_1 sends a commitment to Q_1 and a proof knowledge of it's discrete log).
2. ***P_2*'s first message:**
   (a) P_2 receives $(\mathsf{proof\text{-}receipt}, 1)$ from $\mathcal{F}^{R_{DL}}_{\mathsf{com\text{-}zk}}$.
   (b) P_2 chooses a random $x_2 \in \mathbb{Z}_{N^*}$, and computes $Q_2 = x_2 \cdot G$.
   (c) P_2 sends $(\mathsf{prove}, 2, Q_2, x_2)$ to $\mathcal{F}^{R_{DL}}_{\mathsf{zk}}$.

3. **P_1's second message:**
   (a) P_1 receives $(\mathsf{proof}, 2, Q_2)$ from $\mathcal{F}_{\mathsf{zk}}^{R_{DL}}$. If not, it aborts.
   (b) P_1 sends $(\mathsf{decom\text{-}proof}, 1)$ to $\mathcal{F}_{\mathsf{com\text{-}zk}}^{R_{DL}}$.
   (c) P_1 generates a Paillier key-pair (pk, sk) of length $max(3 \log |q| + 1, n)$ and computes $c_{key} = \mathsf{Enc}_{pk}(x_1)$. Denote $N = pk$. (Note that n denotes the minimum length of N for Paillier to be secure)
   (d) P_1 sends $pk = N$ and c_{key} to P_2.

4. **ZK Proofs:** P_1 proves to P_2 in zero knowledge that $N \in L_p$ and that $(c_{key}, pk, Q_1) \in L'_{PDL}$(refered to section 3) .

5. **P_2's verification:** P_2 aborts unless all the following hold: **(a)**it received $(\mathsf{decom\text{-}proof}, 1, Q_1)$ from $\mathcal{F}_{\mathsf{zk}}^{R_{DL}}$, **(b)**it holds that $c_{key} \in \mathbb{Z}^*_{N^2}$, **(c)**it accepted the proofs: $(1)N \in L_p$ and $(2)(c_{key}, pk, Q_1) \in L_{PDL}$, **(d)**the received key $pk = N$ is of length at least $max(3 \log |q| + 1, n)$.

6. **Output:**
   (a) P_1 computes $Q = Q_1 + Q_2$ and stores (x_1, Q).
   (b) P_2 computes $Q = Q_1 + Q_2$ and stores (x_2, Q, c_{key}).

Beyond generating Q, the protocol concludes with P_2 holding a Paillier encryption of x_1, where $Q_1 = x_1 \cdot G$. As described, this is used to obtain higher efficiency in the signing protocol, and is guaranteed via a zero-knowledge proof.

2.2 Distributed Signing

Protocol 2: Signing Subprotocol Sign(sid, m)

Inputs:

1. Party P_1 has $(x, Q, (pk, sk))$ as output from Key Generation Protocol. the message m, and a unique session id sid.

2. Party P_1 has (x, Q, c_{key}) as output from Key Generation Protocol. the message m, and the session id sid. P_1 and P_2 both locally compute $m' \leftarrow H_q(m)$ and verify that sid has not been used before (if it has been, the protocol is not executed).

1. **P_1's first message:**
   (a) P_1 chooses a random $k_1 \in \mathbb{Z}_q$, $t \in \mathbb{Z}_q$ and computes $R_1 = k_1 \cdot G$.
   (b) P_1 sends $(\mathsf{com\text{-}prove}, sid||1, R_1, k_1, t)$ to $\mathcal{F}_{\mathsf{com\text{-}zk}}^{R_{DL}}$.

2. **P_2's first message:**
   (a) P_2 receives $(\mathsf{proof\text{-}receipt}, sid||1)$ from $\mathcal{F}_{\mathsf{com\text{-}zk}}^{R_{DL}}$.
   (b) P_2 chooses a random $k_2 \in \mathbb{Z}_q$ and computes $R_2 = k_2 \cdot G$.
   (c) P_2 sends $(\mathsf{prove}, sid||2, R_2, k_2)$ to $\mathcal{F}_{\mathsf{zk}}^{R_{DL}}$.

3. **P_1's second message:**
   (a) P_1 receives $(\mathsf{proof}, sid||2, R_2)$ from $\mathcal{F}_{\mathsf{zk}}^{R_{DL}}$; if not, it aborts.
   (b) P_1 sends $(\mathsf{decom\text{-}proof}, sid||1)$ to $\mathcal{F}_{\mathsf{com\text{-}zk}}^{R_{DL}}$.

4. **P_2's second message:**

(a) P_2 receives $(\mathsf{decom\text{-}proof}, sid||1, R_1, t)$ from $\mathcal{F}^{R_{DL}}_{\mathsf{com\text{-}zk}}$; if not or $t = 0$, it aborts.

(b) P_2 computes $R = t \cdot (k_2 \cdot R_1)$. Denote $R = (r_x, r_y)$. Then, P_2 computes $r = r_x \mod q$.

(c) P_2 chooses a random $\rho \leftarrow \mathbb{Z}_{q^2}$ and random $\tilde{r} \in \mathbb{Z}_N^*$ (verifying explicitly that $gcd(\tilde{r}, N) = 1$), and computes $c_1 = \mathsf{Enc}_{pk}(\rho \cdot q + [k_2^{-1} \cdot m' + k_2^{-1} \cdot r \cdot x_2 \pmod q); \tilde{r}])$. Then, P_2 computes $v = k_2^{-1} \cdot r \pmod q$, $c_2 = v \odot (c_{key} \oplus \mathsf{Enc}_{pk}(q; 1))$ and $c_3 = c_1 \oplus c_2$.

(d) P_2 sends c_3 to P_1.

5. **P_1's generates output:**

(a) P_1 computes $R = t \cdot (k_1 \cdot R_2)$. Denote $R = (r_x, r_y)$. Then, P_1 computes $r = r_x \mod q$.

(b) P_1 computes $s' = \mathsf{Dec}_{sk}(c_3)$ and $s'' = k_1^{-1} \cdot s' \pmod q$. P_1 sets $s = \min\{s'', q - s''\}$(this ensures that the signature is always the smaller of the two possible value).

(c) P_1 verifies that (r, s) is a valid signature with public key Q. If yes it outputs the signature (r, s); otherwise, it aborts. If a party aborts at any point, then all $\mathsf{Sign}(sid, m)$ executions are halted.

It is noted that in the new signing protocol, there is:

$$c_3 = Enc_{pk}([k_2^{-1} \cdot m' + k_2^{-1} \cdot r \cdot x_2 \pmod q)] + [k_2^{-1} \cdot r \pmod q)](x_1 + q) + \rho \cdot q)$$

3 Zero-Knowledge Proof of Language L'_{PDL}

I noticed that the ZK-Proof for the Language L_{PDL} actually proves a sightly relaxed variant which is that completeness holds for $x_1 \in \{\frac{q}{3}, ..., \frac{2q}{3}\}$ whereas soundness only hold for $x \in \mathbb{Z}_q$ as follows.

$$L_{PDL} = \{(c, pk, Q_1, \mathbb{G}, G, q) \mid \exists (x_1, r) \text{ such that } c = Enc_{pk}(x_1; r) \text{ and } Q_1 = x_1 \cdot G \text{ and } x_1 \in \mathbb{Z}_q\}$$

However, in Lindell(+) P_1 select $x_1 \in \mathbb{Z}_q$, which means that the completeness holds for $x_1 \in \mathbb{Z}_q$ whereas soundness hold for $x \in \{-q, ..., 2q\}$. To accomplish this objective, we propose replacing the original ZK-L_{PDL} with a newly designed ZK-L'_{PDL}, as follows.

$$L'_{PDL} = \{(c, pk, Q_1, \mathbb{G}, G, q) \mid \exists (x_1, r) \text{ such that } c = Enc_{pk}(x_1; r) \text{ and } Q_1 = x_1 \cdot G \text{ and } x_1 \in \{-q, 2q\}\}$$

3.1 ZK-L'_{PDL}

ZK-L'_{PDL} is derived from ZK-L_{PDL}(Protocol 6.1 of Lindell17), with minimal differences between them. All the differences have been highlighted in red. The proofs for the Completeness, Soundness, and Zero-Knowledge properties of this protocol are almost identical to those of ZK-L_{PDL} , and are described in detail below.

Protocol 3: Zero-Knowledge Proof for the Language L'_{PDL}

Inputs:

The joint statement is $(c, pk, Q_1, \mathbb{G}, G, q)$, and the prover has a witness (x_1, sk) with $x_1 \in \mathbb{Z}_q$.(Recall that the proof is that $x_1 = Dec_{sk}(c)$ and $Q_1 = x_1 \cdot G$ and $x_1 \in \{-q, ..., 2q\}$.)

The Protocol:

1. V chooses a random $a \leftarrow Z_q$ and $b \leftarrow \mathbb{Z}_{3q^2}$ and computes $c' = (a \odot (c \oplus Enc_{pk}(q; 1))) \oplus Enc_{pk}(b; r)$ for random $r \in \mathbb{Z}_N^*$ (verifying explicitly that $gcd(r, N) = 1$ and $gcd(r', N) = 1$), and $c'' = commit(a, b)$. V sends (c', c'') to P. Meanwhile, V computes $Q' = a \cdot Q_1 + b \cdot G$.

2. P receives (c', c'') from V, decrypts it to obtain $\alpha = Dec_{sk}(c')$, and computes $\hat{Q} = \alpha \cdot G$. P sends $\hat{c} = commit(\hat{Q})$ to V.

3. V decommits c'', revealing (a, b).

4. P checks that $\alpha = a \cdot (x_1 + q) + b$ (over the intergers). If not , it aborts. Else it decommits $\hat{c}$ revealing $\hat{Q}$.

5. Range-ZK proof: In parallel to the above, Prove in zero knowledge that $x_1 \in \{-q, ..., 2q\}$, using the proof described in section 3.2.

***V*'s output:** V accepts if and only if it accepts the range proof and $\hat{Q} = Q'$.

Theorem 3.1. *Let $N > 6q^2 + q$. Then, Protocol 6.1 is a zero-knowledge proof for the language L_{PDL} in the $\mathcal{F}$-hybrid model, with completeness 1 for $x_1 \in \mathbb{Z}_q$ and with soundness error $2/q + 2^{-t}$.*

Proof. We prove completeness, soundness and zero knowledge. Completeness follows from the fact that when $N > 6q^2 + q$ there is no reduction modulo N in the Paillier computation and thus P obtains the correct value when decrypting c'. Furthermore, the range zero-knowledge proof has completeness 1 as long as $x_1 \in \mathbb{Z}_q$. We now proceed to the other properties.
Soundness. Let $x_1 = Dec_{sk}(c)$. We consider two cases:

1. *Case 1 - $x_1 \notin \{-q, 2q\}$*: The soundness of the range proof of Step 5 guarantees that V will reject in this case except with probability 2^{-t}.

2. *Case 2 - $x_1 \in \{-q, 2q\}$ but $Q_1 \neq x_1 \cdot G$*: We claim that even an all-powerful cheating P^* cannot cause V to accept with probability greater than $6/q$, in the $\mathcal{F}$-hybrid model. In order to see this, observe that V accepts only if P^* commits to $\hat{Q} = a \cdot Q_1 + b \cdot G$ in Step 2.
P^* receive c' and can decrypt to obtain $\alpha = a \cdot (x_1 + q) + b$. Let $y \in \mathbb{Z}_q$ be such that $Q_1 = y \cdot G$; for this case, $x_1 \neq y \pmod q$. Then, V only accepts if P^* can compute $\beta = a \cdot y + b \pmod q$; to be more exact, P must have committed to $\hat{Q} = a \cdot Q_1 + b \cdot G = a \cdot (y \cdot G) + b \cdot G = [a \cdot y + b \pmod q] \cdot G$. (Note that although P commits to $\hat{Q}$, since it is all-powerful it can compute its discrete log. Thus, if $\hat{Q} = Q'$ then P can obtain $\beta = a \cdot y + b \pmod q$.) Intuitively, P^* cannot succeed since V computes a type of informationtheoretic MAC; it is not standard since the computation is over the integers. Formally, assume that P^* succeeds. This implies that it obtains $\alpha = a \cdot x_1 + b$ and $\beta = a \cdot y + b \pmod q$. Now, P can compute $a = \frac{\alpha - \beta}{x_1 - y} \pmod q$ in order to obtain a. Since $a \in \mathbb{Z}_q$, we have that this is the same value as a over the integers. Next, P^* can compute $b = \alpha - a \cdot x_1$. Thus, if P^* succeeds, then it obtains $(a, b) \in \mathbb{Z}_q \times \mathbb{Z}_{3q^2}$.

   Consider the following experiment, denoted **Expt**1:

   (a) P^* outputs x_1, y.
   (b) Values $a \leftarrow \mathbb{Z}_q$ and $b \leftarrow \mathbb{Z}_{3q^2}$ are chosen uniformly, and $\alpha = a \cdot x_1 + b$ is computed.
   (c) P^* is given α and outputs (a', b').
   (d) P^* succeeds if and only if $a' = a$ and $b' = b$

   Consider the following experiment, denoted **Expt**2:

   (a) P^* outputs x_1, y.
   (b) Values $a \leftarrow \mathbb{Z}_q$ and $b \leftarrow \mathbb{Z}_{3q^2}$ are chosen uniformly, and $\alpha \leftarrow \mathbb{Z}_{6q^2}$ is computed.
   (c) P^* is given α and outputs (a', b').
   (d) P^* succeeds if and only if $a' = a$ and $b' = b$

   The values α in both experiments are from the same range.

   We claim that :
   $$Pr[\mathbf{Expt}^2_{P*} = 1] \geq \frac{1}{6q^2} \cdot Pr[\mathbf{Expt}^1_{P*} = 1].$$

   This holds because with probability $1/6q^2$ the value α received by P^* in $\mathbf{Expt}_2$ is such that $\alpha = a \cdot x_1 + b$. Noting now that

$$Pr[\mathbf{Expt}^2_{P*} = 1] \leq \frac{1}{3q^3}$$

because P^* receives no information whatsoever on (a, b) in $\mathbf{Expt}_2$. Combining the above, we have

$$Pr[\mathbf{Expt}^1_{P*} = 1] \leq 6q^2 \cdot Pr[\mathbf{Expt}^2_{P*} = 1] \leq \frac{6q^2}{3q^3} = \frac{2}{q},$$

as required.

Zero knowledge. We construct a simulator S for a cheating verifier V^* in the $\mathcal{F}$-hybrid model. $\mathcal{S}$ works as follows:

1. $\mathcal{S}$ invokes V^* and obtains (c', c'').
2. $\mathcal{S}$ sends a simulated commitment value $\hat{c}$ to V^*(a receipt value from $\mathcal{F}\rfloor\wr\Updownarrow$).
3. $\mathcal{S}$ receives the *decommitment*(a, b) from V^*. $\mathcal{S}$ verifies that $c' = (a \odot (c \oplus Enc(q; 1))) \oplus b$. If no, then it aborts. If yes, then it sends a decommitment of $\hat{c}$ to $\hat{Q} = a \cdot Q1 + b \cdot G$.
4. $\mathcal{S}$ simulates the range zero-knowledge proof.

V^*'s view is identical to a real protocol execution, in the Fcom-hybrid model. This is because if $c' = (a \odot c) \oplus b$ then P would send the same $\hat{Q}$ as sent by $\mathcal{S}$. This is the only difference between a real execution and the simulated one.

Observe that we only need the commitment to be equivocal; extraction is actually not needed. □

3.2 Range Proof of the Appendix A of Lindell17

Here we present a different version of the ZK-proof in the appendix A of Lindell17 [1] that $x \in \mathbb{Z}_q$ where $c = Enc_{pk}(x)$. All the differences have been highlighted in red. The proof that we use proves for $x \in \mathbb{Z}_q$ that it is in the range $\{-q, ..., 2q\}$. Let $\ell = q$. Stated differently, the input is $x \in \{0, ..., \ell\}$ and the proof guarantees that $x \in [-\ell, 2\ell]$.

The proof of the modified ZK protocol are exactly the same as before and will not be elaborated on here.

4 Proof of Security - Game-Based Definition

The entire proof process remains the same as before. Here, we primarily highlight some noteworthy details.

4.1 *Proof of (1) for* $b = 1$ *- corrupted* P_1

Here, we mainly introduce the indistinguishability of c_3 between the real execution and the simulation

We therefore prove that $\mathcal{A}$'s view is indistinguishable by showing that despite this difference, the values of c_3 are actually statistically close. In order to see this, first observe that by the definition of ECDSA signing, $s = k^{-1} \cdot (m' + rx) = k_1^{-1} \cdot k_2^{-1} \cdot (m + rx) \pmod q$. Thus, $k_2^{-1} \cdot (m' + rx) = k_1 \cdot s \pmod q$, implying that there exists some $\ell \in \mathbb{N}$ with $0 \leq \ell < q$ such that $k_2^{-1} \cdot (m' + rx) = k_1 \cdot s + \ell \cdot q$. The reason that ℓ is bound between 0 and $3q$ is that in the protocol the only operations without a modular reduction are the multiplication of $[k_2^{-1} \cdot r \pmod q]$ by $(x_1 + q)$, and the addition of $[k_2^{-1} \cdot m' + k_2^{-1} \cdot r \cdot x_2 \pmod q]$. This cannot increase the result by more than $3q^2$. Therefore, the difference between the real execution and simulation with $\mathcal{S}$ is:

1. *Real*: the ciphertext c_3 encrypts $[k_1 \cdot s \pmod q] + \ell \cdot q + \rho \cdot q$.
2. *Simulated*: the ciphertext c_3 encrypts $[k_1 \cdot s \pmod q] + \rho \cdot q$.

We show that for all $k_1, s \in \mathbb{Z}_q$ and $\ell \in \{0, ..., 3q\}$, the above values are statistically close (for a random choice of $\rho \in \mathbb{Z}_{q^2}$. In order to see this, fix k_1, s, ℓ, and let v be a value. If $v \neq [k_1 \cdot s \pmod q] + \zeta \cdot q$ for some ζ, then neither the real or simulated values can equal v. Else, if $v = [k_1 \cdot s \pmod q] + \zeta \cdot q$ for some ζ, then there are three cases:

1. Case $\zeta < \ell$: in this case, v can be obtained in the simulated execution for $\rho < \ell$, but can never be obtained in a real execution.
2. Case $\zeta > q^2 - 1$): in this case, v can be obtained in the real execution for $\rho \geq q^2 - 1 = \ell$, but can never be obtained in a simulated execution.
3. Case $\ell \leq \zeta < q^2 - 1$: in this case, v can be obtained in both the real and simulated executions, with identical probability (observe that in both the real and simulated executions, ρ is chosen uniformly in $\mathbb{Z}_{q^2}$).

Recall that the statistical distance between two distributions $\mathbf{X}$ and $\mathbf{Y}$ over a domain $\mathcal{D}$ is defined to be:

$$\Delta(\mathbf{X}, \mathbf{Y}) = \max_{\mathbf{T} \subseteq \mathbf{D}} |Pr[\mathbf{X} \in \mathbf{T}] - Pr[\mathbf{Y} \in \mathbf{T}]|$$

Let $\mathbf{X}$ be the values generated in a real execution of the protocol and let $\mathbf{Y}$ be the values generated in the simulation with $\mathcal{S}$. Then, taking $\mathbf{T}$ to be set of values v for which $\zeta < \ell$, we have that $Pr[\mathbf{X} \in \mathbf{T}] = 0$ whereas $Pr[\mathbf{Y} \in \mathbf{T}] \leq \frac{3q}{q^2} = \frac{3}{q}$ (this holds since $0 \leq \ell < 3q$ and $\rho \in \mathbf{Z}_{q^2}$). Thus, $\Delta(\mathbf{X}, \mathbf{Y}) = \frac{3}{q}$, which is negligible. (Taking $\mathbf{T}$ to be the set of values v for which $\zeta > q^2 - 1$ would give the same result and are both the maximum since any other values add no difference.) We therefore conclude that the distributions over c_3 in the real and simulated executions are statistically close. This proves that (1) holds for the case that $b = 1$.

4.2 *Proof of (1) for* $b = 2$ *- corrupted* P_2

To be honest, the change in the range of x_1 has no impact on the proof process. It only has a slight effect when calculating the final probabilities. As a result, we obtain:

$$Pr[\mathbf{Expt} - \mathbf{Sign}_{\mathcal{S},\pi}(1^n) = 1] >= \frac{Pr[\mathbf{Expt} - \mathbf{DistSign}_{\mathcal{S},\pi}(1^n) = 1]}{p(n) + 1} - \mu(n).$$

Obviously, this does not affect the correctness of the proof in any way.

5 Simulation Proof of Security(With a new Assumption)

The security of this protocol can be proven in the same way as in Lindell 17. The details are omitted here.

6 Related to SID

The protocol in the paper relies on a session identifier (SID), yet no method for deriving this SID is given. We introduce the following sub-protocol to show how the two parties can securely negotiate and agree on a common SID. Note that the operator *oplus* denotes bitwise XOR.

1. P_1 samples a uniformly random $sid_1 \in \{0, 1\}^n$ and sends a commitment $\mathsf{com}_1 = \mathsf{Com}(sid_1)$ to P_2.
2. P_2 samples a uniformly random $sid_2 \in \{0, 1\}^n$ and sends sid_2 to P_1.
3. P_1 opens (decommits) com_1 to P_2, computes $sid = sid_1 \oplus sid_2$, and outputs sid.
4. Upon receiving the opening of com_1, P_2 verifies it. If the opening is invalid, P_2 aborts. Otherwise, P_2 computes $sid = sid_1 \oplus sid_2$ and outputs sid.

7 Security patch

We adopt the elegant and lightweight patch proposed in Paper [2] to address the attack techniques already identified in the literature [2] [3].

It is important to emphasize that while this patch has the advantage of not significantly increasing the complexity of Lindell 17 and effectively mitigates the known attacks, this does not imply that it can defend against all potential or future attacks.

References

[1] Y. Lindell, "Fast secure two-party ecdsa signing," in *Advances in Cryptology–CRYPTO 2017: 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20–24, 2017, Proceedings, Part II 37*. Springer, 2017, pp. 613–644.

[2] J. He and W. Ma, "Attacks on implementations of lindell 17 and its variants," in *International Conference on Data Security and Privacy Protection*. Springer, 2025, pp. 118–136.

[3] N. Makriyannis, O. Yomtov, and A. Galansky, "Practical key-extraction attacks in leading mpc wallets," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 2024, pp. 3053–3064.