

TORTL DKG

TwO-Round TrustLess Distributed Key Generation

Florian Le Mouël, Nadège Pulgar-Vidal, Thierry St-Jacques-Gagnon

KELVINZERO
WHAT DIGITAL IS MEANT TO BE

POLYTECHNIQUE MONTRÉAL

TRUSTEE

Presented by Florian Le Mouël

Presented on 2026-October-13 at TCPT3
NIST Event for Threshold Call Preview Talks #3

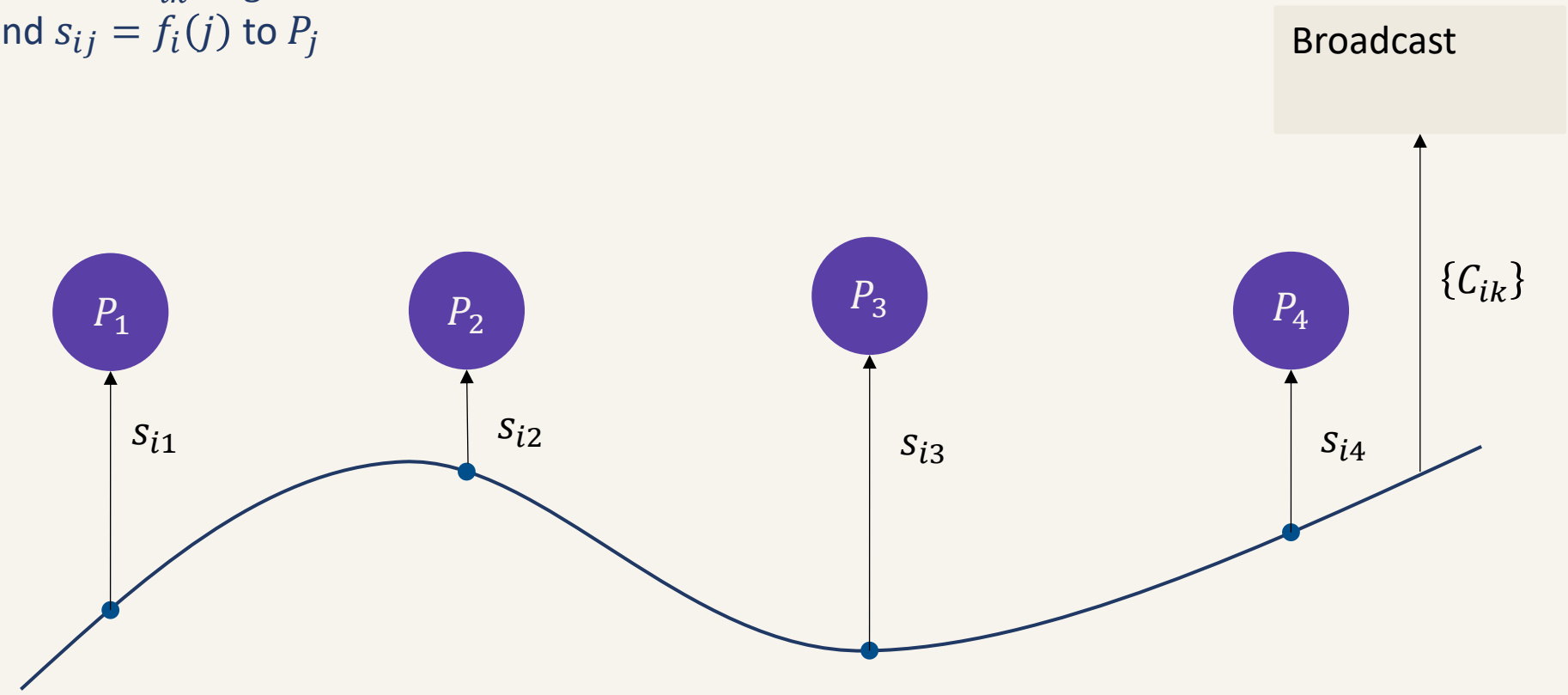
1. Specification
2. Security
3. Performance evaluation
4. Compatibility

1. Specification
2. Security
3. Performance evaluation
4. Compatibility

Complaint based DKG [1]

Round 1:

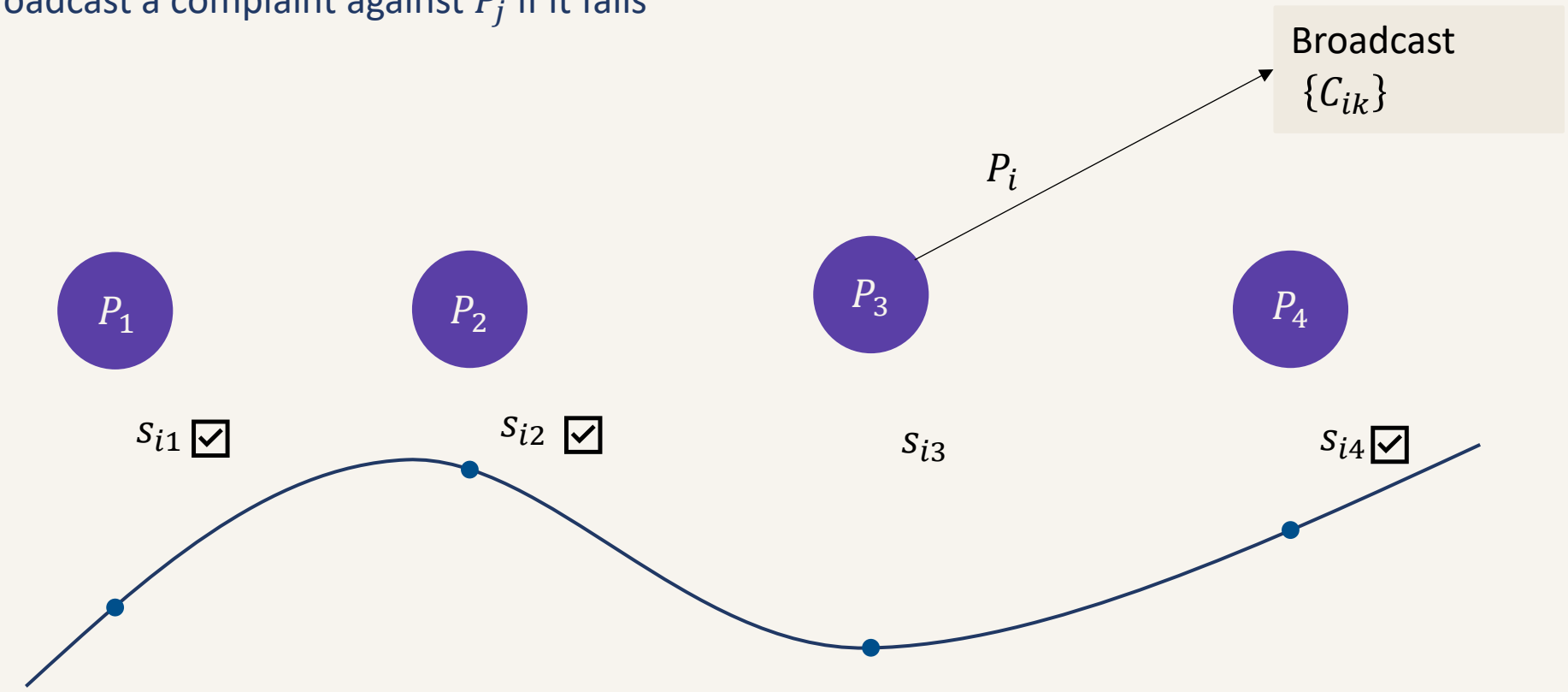
- Generate $f_i = a_{i0} + a_{i1}Z + \cdots + a_{it}Z^t$
- Broadcast $C_{ik} = g^{a_{ik}}$
- Send $s_{ij} = f_i(j)$ to P_j



Complaint based DKG [1]

Round 2:

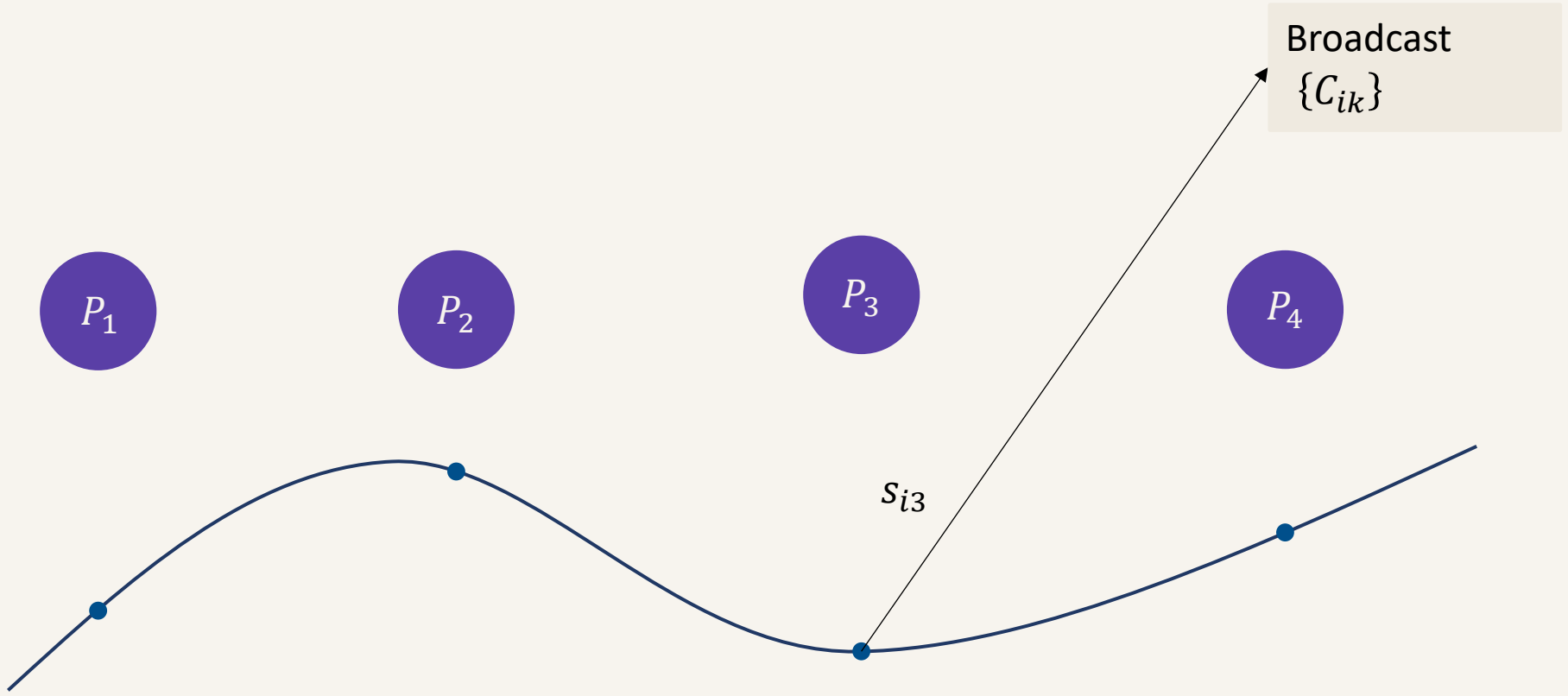
- Verify for each j that $g^{s_{ji}} == \prod C_{jk}^{i^k}$
- Broadcast a complaint against P_j if it fails



Complaint based DKG [1]

Round 3:

- If P_i receives a complaint from P_j, it broadcasts s_{ij}



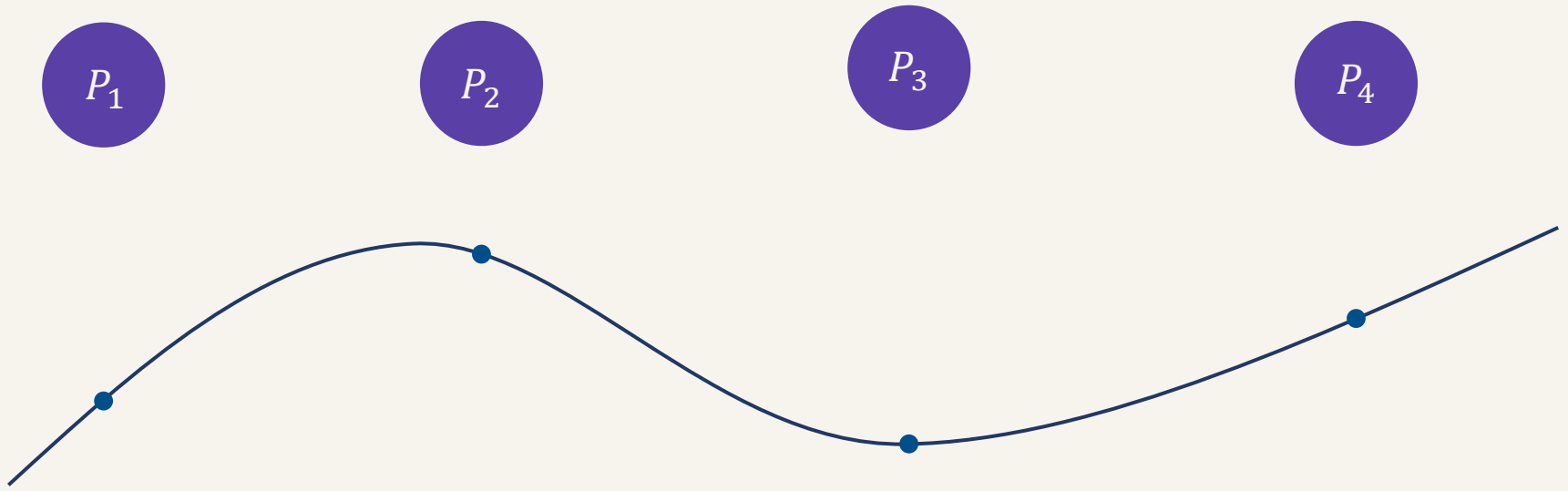
Complaint based DKG [1]

Final computations:

- Verify for each complaint that $g^{s_{ji'}} == \prod C_{jk}^{i'^k}$
- $x_i = \sum s_{ji}$

(for each j that succeeded either of the previous verifications)

Broadcast $\{C_{ik}\}, s_{i3}$ ☑



TORTL DKG



Round 1:

- Generate secret shares $psk_{i->j}$
- Distribute partial secrets $psk_{i->j}$ to P_j



Round 2:

- Combine partial secrets $sk_i = \sum psk_{j->i}$
- Commit to secret share $com_i = commit(sk_i)$
- Compute PRF with secret share $eval_i = evaluate(sk_i, b)$
- Prove relation between com_i and $eval_i$ $\pi_i = prove(eval_i, com_i, sk_i)$
- Broadcast $(com_i, eval_i, \pi_i)$



Final computations:

- Verify each π_j
- Compare threshold, threshold minus one, and all $eval_i$ reconstructions

Two members: TORTL-EC-P256 and TORTL-ML DKG

TORTL-EC-P256 · Elliptic Curve based

Round 1:

- Generate $f_i = a_{i0} + a_{i1}Z + \cdots + a_{it}Z^t$
- Send $psk_{i->j} = f_i(j)$ to P_j

Round 2:

- $sk_i = \sum psk_{j->i}$
- $com_i = sk_i \cdot g_1 + r_i \cdot g_2$
- $eval_i = sk_i \cdot b$
- $y_1, y_2 \leftarrow \mathbb{Z}_q$
- $t_{1i} = y_1 \cdot h'; t_{2i} = y_1 \cdot g_1 + y_2 \cdot g_2$
- $chal_i = h(b, eval_i, com_i, g_1, g_2, t_{1i}, t_{2i})$
- $z_{1i} = y_1 - chal_i \cdot sk_i; z_{2i} = y_2 - chal_i \cdot r_i$
- Broadcast $com_i, eval_i, (z_{1i}, z_{2i}, chal_i)$

Final computations:

- For all j
  - $t_{1j}' = z_{1j} \cdot b + chal_j \cdot eval_j$
  - $t_{2j}' = z_{1j} \cdot g_1 + z_{2j} \cdot g_2 + chal_j \cdot com_j$
  - Verify $chal_j == h(b, eval_j, com_j, g_1, g_2, t_{1j}', t_{2j}')$
- Verify $Lagrange(t) == Lagrange(n)$
- Verify $Lagrange(t) \neq Lagrange(t-1)$

TORTL-ML · Module Lattice based

Round 1:

- Generate $\{a_I\}_{|I|=n-t+1}$
- Send $psk_{i->I} = a_I$ to P_j for j in I

Round 2:

- Verify $\|psk_{j->I}\|_\infty \leq 1$
- $sk_I = \sum psk_{j->\mathrm{I}}$
- $com_{iI} = A \cdot sk_I + B \cdot r_{iI}$
- $eval_{iI} =< b, sk_I >$
- $y_{1I}, y_{2I} \leftarrow [-\beta, \beta]^k$
- $t_{1Ii} =< b, y_{1I} >; t_{2Ii} = A \cdot y_{1I} + B \cdot y_{2I}$
- $chal_{iI} = h(b, eval_{iI}, com_{iI}, A, B, t_{1Ii}, t_{2Ii})$
- $z_{1Ii} = y_{1I} - chal_{iI} \cdot sk_{iI}; z_{2Ii} = y_{2I} - chal_{iI} \cdot r_{iI}$
- Rejection sample until $\|z_{1Ii}\|_\infty$ and $\|z_{2Ii}\| \leq \beta - \kappa \cdot n$
- Broadcast $com_{iI}, eval_{iI}, (z_{1Ii}, z_{2Ii}, chal_{iI})$

Final computations:

- For all j, I
  - $t_{1Ij}' =< b, z_{1Ij} > + chal_{jI} \cdot eval_{jI}$
  - $t_{2Ij}' = A \cdot z_{1Ij} + B \cdot z_{2Ij} + chal_j \cdot com_j$
  - Verify $chal_{jI} == h(b, eval_{jI}, com_{jI}, A, B, t_{1Ii}', t_{2Ii}')$
- For all I
  - Verify $eval_{jI} == evalj'I$

Two members: TORTL-EC-P256 and TORTL-ML DKG

TORTL-EC-P256 · Elliptic Curve based

- Shamir’s secret sharing
- Pedersen commitments
- Dlog-based PRF
- Schnorr-based ZKP

TORTL-ML · Module Lattice based

- Replicated secret sharing
- Module lattice commitments
- Module lattice PRF
- Lyubashevsky-based ZKP

1. Specification
2. **Security**
3. Performance evaluation
4. Compatibility

Security of TORTL DKG



Secure e2e channels, Broadcast channels

Round 1 requires end-to-end channels that are encrypted and authenticated. Round 2 requires broadcast channels to ensure each honest party validates the same proofs as others.



Static corruption

We have yet to study the effect of dynamic corruption on TORTL.



Possible hybrid security

It is possible to combine TORTL-ML with TORTL-EC to obtain hybrid security.

Security of TORTL-EC-P256 DKG



Secrecy relies on DDH, Correctness relies on DLog

Evaluations are pseudo random assuming DDH. Breaking correctness requires breaking ECDSA soundness or Pedersen commitment's binding.



ROM assumption

Fiat-Shamir transform used to make ZKP non-interactive.



Rushing adversaries

No arbitrary shared public key. No possible brute force to enforce rare predicate, unless $t = n$.

Security of TORTL-ML DKG



Secrecy relies on MLWE, Correctness relies on MSIS

Evaluations are pseudo random assuming MLWE. Breaking correctness requires breaking MLDSA soundness or BDLOP commitment's binding.



ROM assumption

Fiat-Shamir transform used to make ZKP non-interactive. We have yet to study whether the hash function can be modeled as a QROM.



Rushing adversaries

No arbitrary shared public key. No possible brute force to enforce rare predicate, unless $t \geq n/2 + 1$.

1. Specification
2. Security
3. **Performance evaluation**
4. Compatibility

Setup timing metrics

TORTL-EC-P256 DKG

t	n	Runtime (ms)	t	n	Runtime (ms)
2	3	21	4	64	332
3	4	31	8	64	343
4	4	35	16	64	387
4	8	52	32	64	438
8	8	61	64	64	560
4	16	91	4	128	678
8	16	99	8	128	679
16	16	121	16	128	742
4	32	174	32	128	838
8	32	180	64	128	950
16	32	202	128	128	1203
32	32	249			

Payload size

TORTL-EC-P256 DKG

Artifact	Proportional to	Size
psk	$log(q)$	32 B
com	$log(p)$	33 B
z	$log(q)$	32 B
$chal$	$log(q)$	32 B
$eval$	$log(p)$	33 B
Total	-	194 B
Pedersen DKG	t	32 + 33t B

TORTL-ML DKG

Artifact	Proportional to	Size (4 of 5 example)
psk	d, k	305 B
com	$log(q), d, m$	6.42 kB
z	$log(2\beta), d, k$	3.78 kB
$chal$	-	32 B
$eval$	$log(p), d/s$	195 B
Total	-	14.5 kB
Total (redundancy accounted)	$\binom{n-1}{t-1}$	59 kB

1. Specification
2. Security
3. Performance evaluation
4. Compatibility

TORTL-EC DKG compatibility



Trout++ · threshold ECDSA

Trout ++ [2] is designed to work with Shamir secret sharing [3], allowing flexibility regarding key distribution. This is exactly the result of TORTL-EC's DKG, which is also flexible when it comes to curve choices.



KU · threshold ECDSA

KU's [6] key generation is modeled as an ideal functionality distributing the key through Shamir secret sharing, which is supported by TORTL-EC.



Twig · threshold ECDSA

Twig [4] and requires a Shamir secret sharing of the signing key on secp256k1, although they plan support for NIST standard curves. TORTL-EC DKG, while submitted on P256, also supports secp256k1, which it was initially designed on [5].



Other SSS based schemes
Other EC choices

TORTL-EC outputs Shamir secret shares of an elliptic curve coefficient, Pedersen commitments [7] to said shares, compatible with Chaum-Pedersen style ZKP [8], and the reconstructed DVRF evaluation [9] serves as a public key.

TORTL-ML DKG compatibility

Mithril · threshold MLDSA

Mithril [10] relies on replicated secret sharing of a small norm secret key over module lattices. This is exactly the use case TORTL-ML was designed to support.

Other RSS based schemes
Other parameter choices

TORTL-ML outputs replicated secret shares of a small polynomial vector, BDLOP commitments [11] to said shares, compatible with Chaum-Pedersen style ZKP, and the reconstructed DVRF evaluation [12] serves as a public key.

Other TORTL DKG compatibility

DiSE · Distributed Symmetric-key Encryption

DiSE [9] designs a generic blueprint for threshold authenticated encryption. We have successfully integrated TORTL with DiSE in previous work [5].

References

[1] Torben Pryds Pedersen. “A threshold cryptosystem without a trusted party”. In: Workshop on the Theory and Application of of Cryptographic Techniques. Springer. 1991, pp. 522–526. DOI: 10.1007/3-540-46416-6_47.

[2] Hila Dahari-Garbian, Ariel Nof, and Luke Parker. Trout++. Preview Writeup submitted to the NIST First Call for Multi-Party Threshold Schemes. May 2026. URL: https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Trout-PW02.pdf.

[3] Adi Shamir. “How to share a secret”. In: Communications of the ACM 22.11 (1979), pp. 612–613. DOI: 10.1145/359168.359176.

[4] Zengpeng Li, Yingjie Lyu, Xudong Deng, Hong-Sheng Zhou, Mei Wang, Puwen Wei, and Chen Qian. Twig: Two-Round Threshold ECDSA. Preview Writeup submitted to the NIST First Call for Multi-Party Threshold Schemes. February 2026. URL: https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Twig-PW02.pdf.

[5] Florian Le Mouël, Maxime Godon, Renaud Brien, Erwan Beurier, Nora Boulahia-Cuppens, and Frédéric Cuppens. “Trustless Distributed Symmetric-key Encryption”. In: arXiv preprint arXiv:2408.16137 (2024). DOI: 10.48550/arXiv.2408.16137.

[6] Jonathan Katz, Antoine Urban, Denis Varlakov, and Nikita Sorokovikov. KU. Preview Writeup submitted to the NIST First Call for Multi-Party Threshold Schemes. July 2026. URL: https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Dfns-KU-PW02.pdf.

[7] Torben Pryds Pedersen. “Non-interactive and information-theoretic secure verifiable secret sharing”. In: Annual international cryptology conference. Springer. 1991, pp. 129–140. DOI: 10.1007/3-540-46766-1_9.

[8] David Chaum and Torben Pryds Pedersen. “Wallet databases with observers”. In: Annual international cryptology conference. Springer. 1992, pp. 89–105. DOI: 10.1007/3-540-48071-4_7.

[9] Shashank Agrawal, Payman Mohassel, Pratyay Mukherjee, and Peter Rindal. “DiSE: distributed symmetric-key encryption”. In: Proceedings of the 2018 ACM SIGSAC conference on computer and communications security. 2018, pp. 1993–2010. DOI: 10.1145/3243734.3243774. URL: https://eprint.iacr.org/2018/727. Also at ia.cr/2018/727.

[10] Sofía Celi, Gustavo Delerue, Rafael del Pino, Thomas Espitau, Guilhem Niot, and Thomas Prest. Mithril. Preview Writeup submitted to the NIST First Call for Multi-Party Threshold Schemes. January 2026. URL: https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Mithril-PW01.pdf.

[11] Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. “More efficient commitments from structured lattice assumptions”. In: International conference on security and cryptography for networks. Springer. 2018, pp. 368–385. DOI: 10.1007/978-3-319-98113-0_20. Also at ia.cr/2017/523.

[12] Muhammed F Esgin, Veronika Kuchta, Amin Sakzad, Ron Steinfeld, Zhenfei Zhang, Shifeng Sun, and Shumo Chu. “Practical post-quantum few-time verifiable random function with applications to algorand”. In: International Conference on Financial Cryptography and Data Security. Springer. 2021, pp. 560–578. DOI: 10.1007/978-3-662-64331-0_29. Also at ia.cr/2020/1222.



TORTL DKG

Thank you.

tortl@googlegroups.com