Lattices give us KEMs and FHE, but where are the efficient lattice PETs?

By Example of (Verifiable) Oblivious PRFs

Martin R. Albrecht

24 June 2025

Outline

LWE and DH

Follow the Blueprint

Trapdoor Friendly

Noise Leakage

Noise Growth

Wrapping Up

LWE and DH

On the one hand, on the other hand

Bottom of the Stack:

- Kyber (KEM)
- Dilithium (Signature)
- Falcon (Signature)

Top of the Stack:

- FHE [Gen09; BGV12; CGGI20; GSW13]
- iO [GGHRSW13; BDGM20]
- FE [SW05; BSW11]

Somewhat efficient

RSA 2048

Key generation	≈ 130,000,000 cycles
Encapsulation	≈ 20,000 cycles
Decapsulation	≈ 2,700,000 cycles
Ciphertext	256 bytes
Public key	256 bytes

https://bench.cr.yp.to/results-kem.html

Kyber-768

Key generation	≈ 38,000 cycles
Encapsulation	≈ 49,000 cycles
Decapsulation	≈ 39,000 cycles
Ciphertext	1,088 bytes
Public key	1,184 bytes

https://bench.cr.yp.to/results-kem.html

Curve25519

Key generation	≈ 60,000 cycles
Key agreement	≈ 160,000 cycles
Public key	32 bytes
Key Share	32 bytes

https://eprint.iacr.org/2015/943

The Learning with Errors Problem (LWE)

Given $(\mathbf{A}, \mathbf{c})$ with $\mathbf{c} \in \mathbb{Z}_q^m$, $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \in \mathbb{Z}_q^n$ and small $\mathbf{e} \in \mathbb{Z}^m$ is

$$\begin{pmatrix} \\ \mathbf{c} \\ \\ \end{pmatrix} = \begin{pmatrix} \leftarrow & n & \rightarrow \\ & \mathbf{A} & \end{pmatrix} \times \begin{pmatrix} \\ \mathbf{s} \\ \end{pmatrix} + \begin{pmatrix} \\ \mathbf{e} \\ \\ \end{pmatrix}$$

or $\mathbf{c} \hookleftarrow \$ \, \mathcal{U}\left(\mathbb{Z}_q^m\right)$.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \end{pmatrix} = \begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} & a_{0,5} & a_{0,6} & a_{0,7} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & a_{1,5} & a_{1,6} & a_{1,7} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & a_{2,5} & a_{2,6} & a_{2,7} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} & a_{3,6} & a_{3,7} \\ a_{4,0} & a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} & a_{4,5} & a_{4,6} & a_{4,7} \\ a_{5,0} & a_{5,1} & a_{5,2} & a_{5,3} & a_{5,4} & a_{5,5} & a_{5,6} & a_{5,7} \\ a_{6,0} & a_{6,1} & a_{6,2} & a_{6,3} & a_{6,4} & a_{6,5} & a_{6,6} & a_{6,7} \\ a_{7,0} & a_{7,1} & a_{7,2} & a_{7,3} & a_{7,4} & a_{7,5} & a_{7,6} & a_{7,7} \end{pmatrix} \cdot \begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{pmatrix} + \begin{pmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{pmatrix}$$

Performance

Storage: $\mathcal{O}(n^2)$; Computation $\mathcal{O}(n^2)$

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \end{pmatrix} = \begin{pmatrix} a_0 & -a_7 & -a_6 & -a_5 & -a_4 & -a_3 & -a_2 & -a_1 \\ a_1 & a_0 & -a_7 & -a_6 & -a_5 & -a_4 & -a_3 & -a_2 \\ a_2 & a_1 & a_0 & -a_7 & -a_6 & -a_5 & -a_4 & -a_3 \\ a_3 & a_2 & a_1 & a_0 & -a_7 & -a_6 & -a_5 & -a_4 \\ a_4 & a_3 & a_2 & a_1 & a_0 & -a_7 & -a_6 & -a_5 \\ a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & -a_7 & -a_6 \\ a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & -a_7 \\ a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \end{pmatrix} \cdot \begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{pmatrix} + \begin{pmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \end{pmatrix}$$

Performance (n is a power of two)

Storage: $\mathcal{O}(n)$; Computation $\mathcal{O}(n \log n)$

Ring-LWE/Polynomial-LWE

$$\sum_{i=0}^{n-1} c_i \cdot X^i = \left(\sum_{i=0}^{n-1} a_i \cdot X^i\right) \cdot \left(\sum_{i=0}^{n-1} s_i \cdot X^i\right) + \sum_{i=0}^{8} e_i \cdot X^i \bmod X^n + 1$$

$$c(X) = a(X) \cdot s(X) + e(X) \bmod \phi(X)$$

We write $\mathcal{R} \coloneqq \mathbb{Z}[X]/\phi(X)$ and $\mathcal{R}_q \coloneqq \mathbb{Z}_q[X]/\phi(X)$.

Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. Efficient Public Key Encryption Based on Ideal Lattices. In: *ASIACRYPT 2009*. Ed. by Mitsuru Matsui. Vol. 5912. LNCS. Springer, Berlin, Heidelberg, Dec. 2009, pp. 617–635. DOI: 10.1007/978-3-642-10366-7_36

Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On Ideal Lattices and Learning with Errors over Rings. In: *EUROCRYPT 2010*. Ed. by Henri Gilbert. Vol. 6110. LNCS. Springer, Berlin, Heidelberg, 2010, pp. 1–23. DOI: 10.1007/978-3-642-13190-5_1

Convention

- I am going to use the Ring-LWE formulation

$$c_i(X) = a_i(X) \cdot s(X) + e_i(X)$$

  Thus, each sample corresponds to "n LWE samples"
- I will suppress the "(X)" in "$a(X)$" etc.
- I will assume s is "small" and that the product of two "small" things is "small".
- I will write e_i to emphasise that e_i is small.

TL;DR: I will write

$$c_i = a_i \cdot s + e_i$$

DH to Ring-LWE Dictionary

DH Land	Ring-LWE Land
g	a
g^x	$a \cdot s + e$
$g^x \cdot g^y = g^{x+y}$	$(a \cdot s + e_0) + (a \cdot t + e_1) = a \cdot (s + t) + e'$
$(g^a)^b = (g^b)^a$	$(a \cdot s + e) \cdot t = (a \cdot s \cdot t + e \cdot t)$ $\approx a \cdot s \cdot t \approx (a \cdot t + e) \cdot s$
(g, g^a, g^b, g^{ab}) $\approx_c (g, g^a, g^b, u)$	$(a,\ a \cdot s + e,\ a \cdot t + d,\ a \cdot s \cdot t + e')$ $\approx_c (a,\ a \cdot s + e,\ a \cdot t + d,\ u)$

ElGamal

KeyGen $h = g^s$

Encrypt $d_0,\ d_1 = (g^v,\ m \cdot h^v)$ for some random v

Decrypt $d_1/d_0^s = m \cdot (g^s)^v/(g^v)^s = m$

[LPR10]

KeyGen $c = a \cdot s + e$

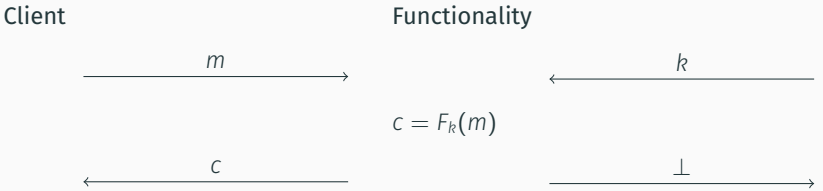
Encrypt $d_0,\ d_1 = v \cdot a + e',\ v \cdot c + e'' + \left\lfloor \frac{q}{2} \right\rfloor \cdot m$

Decrypt

$$\begin{aligned}\left\lfloor \frac{2}{q} \cdot (d_1 - d_0 \cdot s) \right\rceil &= \left\lfloor \frac{2}{q} \cdot \left(v \cdot (a \cdot s + e) + e'' + \left\lfloor \frac{q}{2} \right\rfloor \cdot m - (v \cdot a + e') \cdot s \right) \right\rceil \\ &= \left\lfloor \frac{2}{q} \cdot \left(v \cdot e + e'' + \left\lfloor \frac{q}{2} \right\rfloor \cdot m - e' \cdot s \right) \right\rceil = m\end{aligned}$$

Follow the Blueprint

Verifiable Oblivious PRFs



Example Applications: Privacy Pass

Problem:

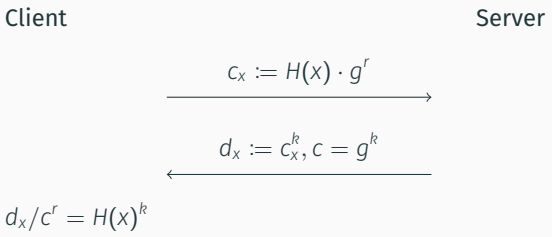
- Tor users are having a hard time on Cloudflare protected sites
- They're constantly asked to solve CAPTCHAs to prove that they're not bots
- Want a privacy-preserving way of running reverse Turing test once and re-use later

Idea:

- Solve CAPTCHA
- Evaluate a VOPRF on a bunch of random points to produce tokens $F_k(x_i)$
- Redeem token by sending $(x_i, F_k(x_i))$

Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy Pass: Bypassing Internet Challenges Anonymously. In: *PoPETs* 2018.3 (July 2018), pp. 164–180. DOI: 10.1515/popets-2018-0026

DH-Based OPRF



$$d_x / c^r = c_x^k / c^r = (H(x) \cdot g^r)^k / (g^k)^r = H(x)^k$$

"Just take Logs"

Client | Server

$$c_x := H(x) + a \cdot r + e \longrightarrow$$

$$\longleftarrow d_x := c_x \cdot k + e', c := a \cdot k + e''$$

$$\left\lfloor \frac{p}{q} \cdot d_x - c \cdot r \right\rceil$$

$$\begin{aligned} d_x - c \cdot r &= (H(x) + a \cdot r + e) \cdot k + e' - (a \cdot k + e'') \cdot r \\ &= H(x) \cdot k + a \cdot r \cdot k + e \cdot k + e' - a \cdot k \cdot r - e'' \cdot r \\ &= H(x) \cdot k + e \cdot k + e' - e'' \cdot r \\ &\approx H(x) \cdot k \end{aligned}$$

The Trouble

Trapdoor Friendly It is not safe to output $c_x \cdot k + e$ for some arbitrary c_x

Noise Leakage "$\approx$" glosses over $e \cdot k + e' - e'' \cdot r$ which depends on k

Noise Growth "$\approx$" is not $=$, how do we arrive at the same value?

Trapdoor Friendly

The Problem

The server has to output $c_x \cdot k + e'$ for some $c_x \stackrel{?}{=} H(x) + a \cdot r + e$. This may not be safe.

Point Validation

Validation of Elliptic Curve Public Keys

Adrian Antipa[1], Daniel Brown[1], Alfred Menezes[2],
René Struik[1], and Scott Vanstone[2]

[1] Certicom Research, Canada
{aantipa,dbrown,rstruik}@certicom.com
[2] Dept. of Combinatorics and Optimization, University of Waterloo, Canada
{ajmeneze,savansto}@uwaterloo.ca

Abstract. We present practical and realistic attacks on some standardized elliptic curve key establishment and public-key encryption protocols that are effective if the receiver of an elliptic curve point does not check that the point lies on the appropriate elliptic curve. The attacks combine ideas from the small subgroup attack of Lim and Lee, and the differential fault attack of Biehl, Meyer and Müller. Although the ideas behind the attacks are quite elementary, and there are simple countermeasures known, the attacks can have drastic consequences if these countermeasures are not taken by implementors of the protocols. We illustrate the effectiveness of such attacks on a key agreement protocol recently proposed for the IEEE 802.15 Wireless Personal Area Network (WPAN) standard.

Point Validation

Validation of Elliptic Curve Public Keys

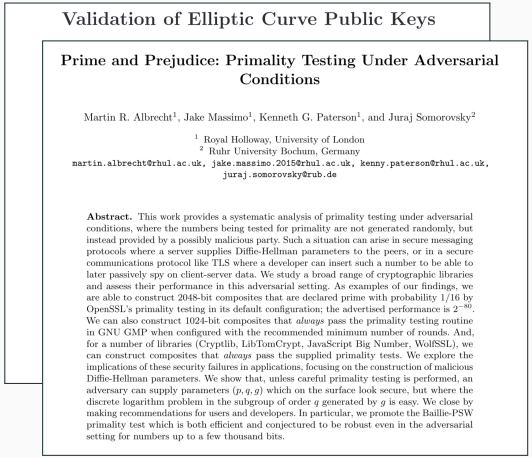
Prime and Prejudice: Primality Testing Under Adversarial Conditions

Martin R. Albrecht[1], Jake Massimo[1], Kenneth G. Paterson[1], and Juraj Somorovsky[2]

[1] Royal Holloway, University of London
[2] Ruhr University Bochum, Germany
martin.albrecht@rhul.ac.uk, jake.massimo.2015@rhul.ac.uk, kenny.paterson@rhul.ac.uk, juraj.somorovsky@rub.de

Abstract. This work provides a systematic analysis of primality testing under adversarial conditions, where the numbers being tested for primality are not generated randomly, but instead provided by a possibly malicious party. Such a situation can arise in secure messaging protocols where a server supplies Diffie-Hellman parameters to the peers, or in a secure communications protocol like TLS where a developer can insert such a number to be able to later passively spy on client-server data. We study a broad range of cryptographic libraries and assess their performance in this adversarial setting. As examples of our findings, we are able to construct 2048-bit composites that are declared prime with probability 1/16 by OpenSSL's primality testing in its default configuration; the advertised performance is 2^{-80}. We can also construct 1024-bit composites that *always* pass the primality testing routine in GNU GMP when configured with the recommended minimum number of rounds. And, for a number of libraries (Cryptlib, LibTomCrypt, JavaScript Big Number, WolfSSL), we can construct composites that *always* pass the supplied primality tests. We explore the implications of these security failures in applications, focusing on the construction of malicious Diffie-Hellman parameters. We show that, unless careful primality testing is performed, an adversary can supply parameters (p, q, g) which on the surface look secure, but where the discrete logarithm problem in the subgroup of order q generated by g is easy. We close by making recommendations for users and developers. In particular, we promote the Baillie-PSW primality test which is both efficient and conjectured to be robust even in the adversarial setting for numbers up to a few thousand bits.

Point Validation



Validation of Elliptic Curve Public Keys

Prime and Prejudice: Primality Testing Under Adversarial Conditions

Martin R. Albrecht[1], Jake Massimo[1], Kenneth G. Paterson[1], and Juraj Somorovsky[2]

[1] Royal Holloway, University of London
[2] Ruhr University Bochum, Germany
martin.albrecht@rhul.ac.uk, jake.massimo.2015@rhul.ac.uk, kenny.paterson@rhul.ac.uk, juraj.somorovsky@rub.de

Abstract. This work provides a systematic analysis of primality testing under adversarial conditions, where the numbers being tested for primality are not generated randomly, but instead provided by a possibly malicious party. Such a situation can arise in secure messaging protocols where a server supplies Diffie-Hellman parameters to the peers, or in a secure communications protocol like TLS where a developer can insert such a number to be able to later passively spy on client-server data. We study a broad range of cryptographic libraries and assess their performance in this adversarial setting. As examples of our findings, we are able to construct 2048-bit composites that are declared prime with probability 1/16 by OpenSSL's primality testing in its default configuration; the advertised performance is 2^{-80}. We can also construct 1024-bit composites that *always* pass the primality testing routine in GNU GMP when configured with the recommended minimum number of rounds. And, for a number of libraries (Cryptlib, LibTomCrypt, JavaScript Big Number, WolfSSL), we can construct composites that *always* pass the supplied primality tests. We explore the implications of these security failures in applications, focusing on the construction of malicious Diffie-Hellman parameters. We show that, unless careful primality testing is performed, an adversary can supply parameters (p, q, g) which on the surface look secure, but where the discrete logarithm problem in the subgroup of order q generated by g is easy. We close by making recommendations for users and developers. In particular, we promote the Baillie-PSW primality test which is both efficient and conjectured to be robust even in the adversarial setting for numbers up to a few thousand bits.

- However, (likely) no "point validation" for LWE by the NTRU assumption [HPS96]:

$$f, g \hookleftarrow\$ \mathcal{R}^2 : h \coloneqq f/g \approx_c \mathcal{U}(\mathcal{R}_q)$$

- Attack[a]:
  1. Sample $f, g \hookleftarrow\$ \mathcal{R}^2$ and set $\Delta \coloneqq \lceil \sqrt{q} \rceil$
  2. Submit $a \coloneqq \Delta \cdot f/g$
  3. Receive $c \coloneqq a \cdot k + e$ and compute

$$\begin{aligned} g \cdot c &= g \cdot (\Delta \cdot f/g \cdot k + e) \\ g \cdot c &= \Delta \cdot f \cdot k + g \cdot e \\ &\equiv g \cdot e \bmod \Delta \end{aligned}$$

[a] assuming f, g, k, e are sufficiently small

Workaround

- The client proves in zero-knowledge that c_x is well-formed: $c_x := H(x) + a \cdot r + e$
- This means the client needs to prove the evaluation of $H(x)$
- This is sound, we do not need to treat $H(x)$ as a Random Oracle
- This is expensive in terms of bandwidth and computation cost
  - [ADDS21]: $\approx$ 128GB per evaluation using [YAZXYW19]
  - [AG24]: $\approx$ 63kB per evaluation using [BS23]

An Aside

This NTRU "attack" can be used constructively to make proof systems online extractable (e.g. [ADDG24])

Noise Leakage

The client learns

$$e \cdot k + e' - e'' \cdot r$$

where it chooses e and r.

The Attack

Write $\mathbf{a} := (e, -r)$ and $\mathbf{s} := (k, e'')$, then we can rewrite

$$e \cdot k + e' - e'' \cdot r$$

as $\mathbf{a} \cdot \mathbf{s} + e''$ which is essentially an instance of "LWE without modular reduction" [BDEFT18] which is easy.[1]

[1] The word "essentially" does a lot of work here. That is, this is a simplification because e'' changes in each invocation and the attacks from [BDEFT18] do not apply as is.

Solutions

Statistical Noise Drowning $\|e'\| \geq \lambda^{\omega(1)} \cdot \|e \cdot k - e'' \cdot r\|$ [ADDS21]

Rényi Noise Drowning $\|e'\| \geq \mathrm{poly}(\lambda) \cdot \sqrt{Q} \cdot \|e \cdot k - e'' \cdot r\|$ [AG24]

- Q is the number of queries
- must play a search game instead of a distinguishing game (use ROM)

Computational Assumption $\|e'\| \geq \mathrm{poly}(\lambda) \cdot \sqrt{Q} \cdot \|e \cdot k - e'' \cdot r\|$ [ESTX24]

- similar to Hint-(M)LWE, but w/o reduction from (M)LWE

Cost

Since we require $q > \|e'\|$ we have that $q/\|e\|$ – the "signal to noise ratio" of the underlying RLWE samples – is quite big. A big signal to noise ratio makes decoding – i.e. solving LWE – easier. This requires us to use larger secret dimensions n to compensate. Bandwidth cost is essentially $n \log q$.

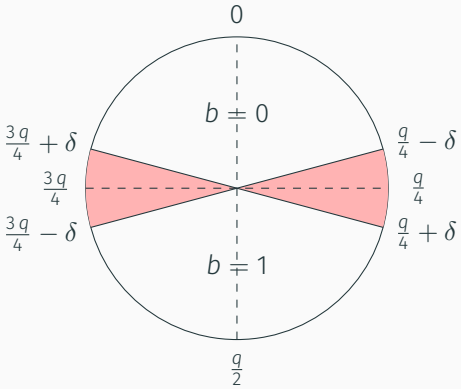
Noise Growth

The Problem

$$H(x) \cdot k + e \cdot k + e' - e'' \cdot r \approx H(x) \cdot k$$

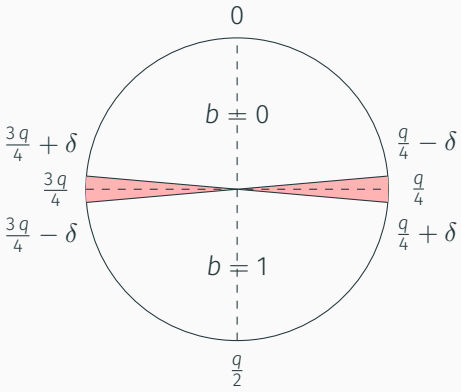
$$\left\lceil \frac{2}{q} \cdot \left(H(x) \cdot k + e \cdot k + e' - e'' \cdot r \right) \right\rfloor \stackrel{?}{=} \left\lceil \frac{2}{q} \cdot \left(H(x) \cdot k \right) \right\rfloor$$

Rounding


0
$b = 0$
$\frac{3q}{4} + \delta$
$\frac{q}{4} - \delta$
$\frac{3q}{4}$
$\frac{q}{4}$
$\frac{3q}{4} - \delta$
$\frac{q}{4} + \delta$
$b = 1$
$\frac{q}{2}$

Solution Attempt

Make $q > \text{poly}(\lambda)$ such that the red area is negligibly small.

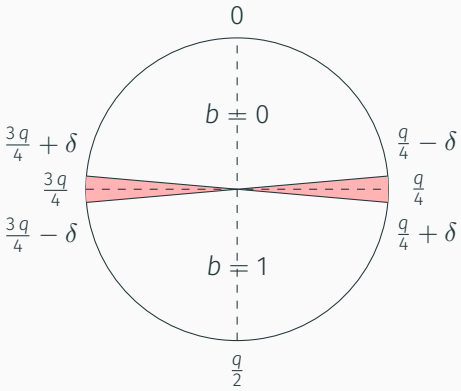


Solution Attempt

Make $q > \text{poly}(\lambda)$ such that the red area is negligibly small.

Malicious Servers

This argument works on average but does not work against adversaries that somehow pick k s.t. $H(x) \cdot k$ lands in the red area with high probability for some x.



Solution

- Plant a hard SIS instance in each coefficient: 1D-SIS [ADDS21]

Solution

- Plant a hard SIS instance in each coefficient: 1D-SIS [ADDS21]
  - Yes, seriously!
  - This requires $q \gg 2^{\lambda}$

Solution

- Plant a hard SIS instance in each coefficient: 1D-SIS [ADDS21]
  - Yes, seriously!
  - This requires $q \gg 2^{\lambda}$
- Change the PRF output to $H(x) \cdot k + H_2(x, c_0)$ [AG24]
  - $H_2()$ is some Random Oracle that randomly shifts $H(x) \cdot k$
  - $c_0 := a_0 \cdot k + e'_0$ is a commitment to k
  - The trick is from [GKQMS24]
  - $q \gg \text{poly}(\lambda)$ is sufficient

Stuck with a super-polynomial q

Big "signal-to-noise" ratio, forcing us to increase n, as above.

Related: Can't Just Do It™ — NIKE

NIKE enables Alice and Bob, who know each others' public keys, to agree on shared key without requiring any interaction [DH76]

- Deployed in WireGuard [HNSWZ20] and static DH is also used in e.g. Google's QUIC.
- For lattices there are significant barriers [GKRS20].
- Stark contrast to interactive key-exchanges or plain public-key encryption
  1. We send along some "hints" that allow to handle the noise
  2. secrets are not re-used, allowing us to avoid expensive "well-formedness" proofs
- [GKQMS24] is an instantiation that essentially accepts the super-polynomial modulus

Wrapping Up

Realisations of this Blueprint

Work	Model	1-time Offline	Online	Q
[ADDS21]	H-H	–	2MB	
[ADDS21]	M-M	–	128GB	
[AG24]	M-M	114kB	198kB	2^{32}
[ESTX24]	M-M	20kB	159kB	2^{32}

H: semi-honest, M: malicious

An Alternative from FHE [ADDG24]

We do have efficient FHE, indeed FHE ciphertexts are typically smaller than the messages exchanged in the schemes discussed above.

- Simple idea:
  1. Client FHE encrypts x as $[x]$
  2. Server homomorphically computes PRF using plaintext k and $[x]$ to obtain $[F_k(x)]$
  3. Client FHE decrypts $F_k(x)$
- Problem: PRFs need deep circuits, deep circuits are expensive
- Proposal: Use Dark Matter (weak-)PRF candidate [BIPSW18] $\sum(\mathbf{A} \cdot \mathbf{x} \bmod 2) \bmod 3$ where $\mathbf{A}$ is the secret key
- This can be computed with one level of FHE bootstrapping

Other Round-Optimal Alternatives w/o Trusted Setup

Work	Assumption	Model	1-time Offline	Online
ADDS21	(R)LWE+SIS	H-H	-	2MB
ADDS21	(R)LWE+SIS	M-M	-	128GB
AG24	(R)LWE+SIS	M-M	114kB	198kB
ADDG23	mod(2,3)+lattices	M-H	2.5MB	10KB
ADDG23	mod(2,3)+lattices	M-M	2.5MB	160KB
ESTX24	iMLWER-RU+MLE+SIS	M-M	20kB	159KB
APRR24	mod(2,3)	M-H, pp	4.75B	114.5B
FOO23	AES+Garbled Circuits	H-H	-	6.79MB
Basso24	Higher-Dimensional isogenies	M-M	-	28.9kB
HHM+23	Isogenies F_p + lattices + HE OT	H-H	-	640kB
dSP23	Isogenies F_p	M-H, pp	68.4 kB	384B
dSP23	Isogenies F_p	M-H, pp	-	16.38kB

adapted from `https://heimberger.xyz/oprfs.html`

Thank You

https://ia.cr/2019/1271

https://ia.cr/2023/232

https://ia.cr/2024/1459

References I

[ADDG24] Martin R. Albrecht, Alex Davidson, Amit Deo, and Daniel Gardham. Crypto Dark Matter on the Torus - Oblivious PRFs from Shallow PRFs and TFHE. In: *EUROCRYPT 2024, Part VI*. Ed. by Marc Joye and Gregor Leander. Vol. 14656. LNCS. Springer, Cham, May 2024, pp. 447–476. DOI: 10.1007/978-3-031-58751-1_16.

[ADDS21] Martin R. Albrecht, Alex Davidson, Amit Deo, and Nigel P. Smart. Round-Optimal Verifiable Oblivious Pseudorandom Functions from Ideal Lattices. In: *PKC 2021, Part II*. Ed. by Juan Garay. Vol. 12711. LNCS. Springer, Cham, May 2021, pp. 261–289. DOI: 10.1007/978-3-030-75248-4_10.

[AG24] Martin R. Albrecht and Kamil Doruk Gür. Verifiable Oblivious Pseudorandom Functions from Lattices: Practical-Ish and Thresholdisable. In: *ASIACRYPT 2024, Part IV*. Ed. by Kai-Min Chung and Yu Sasaki. Vol. 15487. LNCS. Springer, Singapore, Dec. 2024, pp. 205–237. DOI: 10.1007/978-981-96-0894-2_7.

[BDEFT18] Jonathan Bootle, Claire Delaplace, Thomas Espitau, Pierre-Alain Fouque, and Mehdi Tibouchi. LWE Without Modular Reduction and Improved Side-Channel Attacks Against BLISS. In: *ASIACRYPT 2018, Part I*. Ed. by Thomas Peyrin and Steven Galbraith. Vol. 11272. LNCS. Springer, Cham, Dec. 2018, pp. 494–524. DOI: 10.1007/978-3-030-03326-2_17.

[BDGM20] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Candidate iO from Homomorphic Encryption Schemes. In: *EUROCRYPT 2020, Part I*. Ed. by Anne Canteaut and Yuval Ishai. Vol. 12105. LNCS. Springer, Cham, May 2020, pp. 79–109. DOI: 10.1007/978-3-030-45721-1_4.

References II

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In: *ITCS 2012*. Ed. by Shafi Goldwasser. ACM, Jan. 2012, pp. 309–325. DOI: 10.1145/2090236.2090262.

[BIPSW18] Dan Boneh, Yuval Ishai, Alain Passelègue, Amit Sahai, and David J. Wu. Exploring Crypto Dark Matter: New Simple PRF Candidates and Their Applications. In: *TCC 2018, Part II*. Ed. by Amos Beimel and Stefan Dziembowski. Vol. 11240. LNCS. Springer, Cham, Nov. 2018, pp. 699–729. DOI: 10.1007/978-3-030-03810-6_25.

[BS23] Ward Beullens and Gregor Seiler. LaBRADOR: Compact Proofs for R1CS from Module-SIS. In: *CRYPTO 2023, Part V*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14085. LNCS. Springer, Cham, Aug. 2023, pp. 518–548. DOI: 10.1007/978-3-031-38554-4_17.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional Encryption: Definitions and Challenges. In: *TCC 2011*. Ed. by Yuval Ishai. Vol. 6597. LNCS. Springer, Berlin, Heidelberg, Mar. 2011, pp. 253–273. DOI: 10.1007/978-3-642-19571-6_16.

[CGGI20] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: Fast Fully Homomorphic Encryption Over the Torus. In: *Journal of Cryptology* 33.1 (Jan. 2020), pp. 34–91. DOI: 10.1007/s00145-019-09319-x.

References III

[DGSTV18] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. **Privacy Pass: Bypassing Internet Challenges Anonymously.** In: *PoPETs* 2018.3 (July 2018), pp. 164–180. DOI: 10.1515/popets-2018-0026.

[DH76] Whitfield Diffie and Martin E. Hellman. **New Directions in Cryptography.** In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654. DOI: 10.1109/TIT.1976.1055638.

[ESTX24] Muhammed F. Esgin, Ron Steinfeld, Erkan Tairi, and Jie Xu. **LeOPaRd: Towards Practical Post-Quantum Oblivious PRFs via Interactive Lattice Problems.** Cryptology ePrint Archive, Report 2024/1615. 2024. URL: https://eprint.iacr.org/2024/1615.

[Gen09] Craig Gentry. **A fully homomorphic encryption scheme.** crypto.stanford.edu/craig. PhD thesis. Stanford University, 2009.

[GGHRSW13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. **Candidate Indistinguishability Obfuscation and Functional Encryption for all Circuits.** In: *54th FOCS*. IEEE Computer Society Press, Oct. 2013, pp. 40–49. DOI: 10.1109/FOCS.2013.13.

[GKQMS24] Phillip Gajland, Bor de Kock, Miguel Quaresma, Giulio Malavolta, and Peter Schwabe. **SWOOSH: Efficient Lattice-Based Non-Interactive Key Exchange.** In: *USENIX Security 2024*. Ed. by Davide Balzarotti and Wenyuan Xu. USENIX Association, Aug. 2024. URL: https://www.usenix.org/conference/usenixsecurity24/presentation/gajland.

References iv

[GKRS20] Siyao Guo, Pritish Kamath, Alon Rosen, and Katerina Sotiraki. **Limits on the Efficiency of (Ring) LWE Based Non-interactive Key Exchange**. In: *PKC 2020, Part I*. Ed. by Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas. Vol. 12110. LNCS. Springer, Cham, May 2020, pp. 374–395. DOI: 10.1007/978-3-030-45374-9_13.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. **Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based**. In: *CRYPTO 2013, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. LNCS. Springer, Berlin, Heidelberg, Aug. 2013, pp. 75–92. DOI: 10.1007/978-3-642-40041-4_5.

[HNSWZ20] Andreas Hülsing, Kai-Chun Ning, Peter Schwabe, Fiona Johanna Weber, and Philip R. Zimmermann. **Post-quantum WireGuard**. Cryptology ePrint Archive, Report 2020/379. 2020. URL: https://eprint.iacr.org/2020/379.

[HPS96] Jeffery Hoffstein, Jill Pipher, and Joseph H. Silverman. **NTRU: A new high speed public-key cryptosystem**. Tech. rep. available at https://cdn2.hubspot.net/hubfs/49125/downloads/ntru-orig.pdf. Draft distributed at CRYPTO96, 1996.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. **On Ideal Lattices and Learning with Errors over Rings**. In: *EUROCRYPT 2010*. Ed. by Henri Gilbert. Vol. 6110. LNCS. Springer, Berlin, Heidelberg, 2010, pp. 1–23. DOI: 10.1007/978-3-642-13190-5_1.

References v

[SSTX09] Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. **Efficient Public Key Encryption Based on Ideal Lattices.** In: *ASIACRYPT 2009*. Ed. by Mitsuru Matsui. Vol. 5912. LNCS. Springer, Berlin, Heidelberg, Dec. 2009, pp. 617–635. DOI: 10.1007/978-3-642-10366-7_36.

[SW05] Amit Sahai and Brent R. Waters. **Fuzzy Identity-Based Encryption.** In: *EUROCRYPT 2005*. Ed. by Ronald Cramer. Vol. 3494. LNCS. Springer, Berlin, Heidelberg, May 2005, pp. 457–473. DOI: 10.1007/11426639_27.

[YAZXYW19] Rupeng Yang, Man Ho Au, Zhenfei Zhang, Qiuliang Xu, Zuoxia Yu, and William Whyte. **Efficient Lattice-Based Zero-Knowledge Arguments with Standard Soundness: Construction and Applications.** In: *CRYPTO 2019, Part I*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Vol. 11692. LNCS. Springer, Cham, Aug. 2019, pp. 147–175. DOI: 10.1007/978-3-030-26948-7_6.