

Pseudorandom Functions and Lattices

Abhishek Banerjee¹

Chris Peikert¹

Alon Rosen²

¹Georgia Institute of Technology

²IDC Herzliya

EUROCRYPT '12

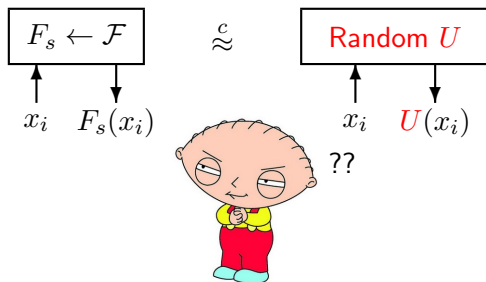
19 April 2012

- 1 Introduction
- 2 Learning with Rounding
- 3 Direct Construction

- 1 Introduction
- 2 Learning with Rounding
- 3 Direct Construction

Pseudorandom Functions [GGM'84]

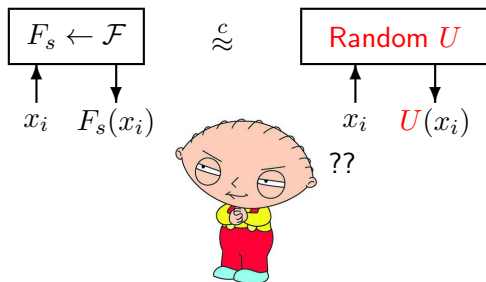
- A family of functions $\mathcal{F} = \{F_s : \{0, 1\}^k \rightarrow B\}$ such that, given adaptive query access,



(Thanks to Seth MacFarlane for the adversary)

Pseudorandom Functions [GGM'84]

- A family of functions $\mathcal{F} = \{F_s : \{0, 1\}^k \rightarrow B\}$ such that, given adaptive query access,



- Lots of applications** in symmetric key cryptography: encryption, message authentication, friend or foe identification...

(Thanks to Seth MacFarlane for the adversary)

Cooking a PRF

- ① “Man-made”: AES, Blowfish...
 - Fast and secure against known cryptanalytic techniques

- ① “Man-made”: AES, Blowfish...
 - Fast and secure against known cryptanalytic techniques
 - Want **provable** security based on a natural hard problem

Cooking a PRF

- 1 “Man-made”: AES, Blowfish...
 - Fast and secure against known cryptanalytic techniques
 - Want provable security based on a natural hard problem
- 2 Goldreich-Goldwasser-Micali [GGM'84]
 - Based on **any (doubling) PRG**. $F_s(x_1, \dots, x_k) = G_{x_k}(\dots(G_{x_1}(s))\dots)$

Cooking a PRF

- 1 “Man-made”: AES, Blowfish...
 - Fast and secure against known cryptanalytic techniques
 - Want provable security based on a natural hard problem
- 2 Goldreich-Goldwasser-Micali [GGM'84]
 - Based on any (doubling) PRG. $F_s(x_1, \dots, x_k) = G_{x_k}(\dots(G_{x_1}(s))\dots)$
 - **Sequential**: at least k iterations

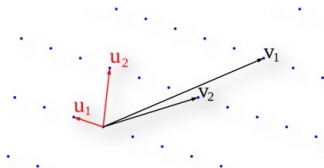
Cooking a PRF

- 1 “Man-made”: AES, Blowfish...
 - Fast and secure against known cryptanalytic techniques
 - Want provable security based on a natural hard problem
- 2 Goldreich-Goldwasser-Micali [GGM'84]
 - Based on any (doubling) PRG. $F_s(x_1, \dots, x_k) = G_{x_k}(\dots(G_{x_1}(s))\dots)$
 - Sequential: at least k iterations
- 3 Direct constructions [NR'95, NR'97, NRR'00]
 - **Parallel** and theoretically efficient
 - Security based on **number-theory** (DDH, factoring...)

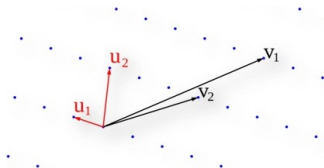
Cooking a PRF

- 1 “Man-made”: AES, Blowfish...
 - Fast and secure against known cryptanalytic techniques
 - Want provable security based on a natural hard problem
- 2 Goldreich-Goldwasser-Micali [GGM'84]
 - Based on any (doubling) PRG. $F_s(x_1, \dots, x_k) = G_{x_k}(\dots(G_{x_1}(s))\dots)$
 - Sequential: at least k iterations
- 3 Direct constructions [NR'95, NR'97, NRR'00]
 - Parallel and theoretically efficient
 - Security based on number-theory (DDH, factoring...)
 - Not practically efficient (**huge exponentiations**), lots of preprocessing
 - What about a “**post-quantum**” world?

Lattices



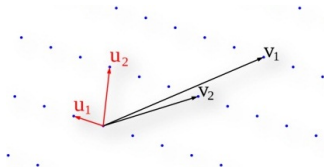
A **periodic grid** in the n -dimensional Euclidean space



A periodic grid in the n -dimensional Euclidean space

Advantages of Lattice Crypto Schemes

- Simple & efficient: linear, highly parallel operations
- Resist quantum attacks (so far)
- Secure under worst-case hardness assumptions [Ajt'96,...]



A periodic grid in the n -dimensional Euclidean space

Advantages of Lattice Crypto Schemes

- Simple & efficient: linear, highly parallel operations
- Resist quantum attacks (so far)
- Secure under worst-case hardness assumptions [Ajt'96,...]

Lattice-based Pseudorandomness?

- Only known PRF is generic GGM, no direct constructions
- Security proofs based on hard lattice problems need fresh biased errors

- ★ Low-depth, highly efficient PRFs from (ring-)LWE

- ★ Low-depth, highly efficient PRFs from (ring-)LWE
 - ★ Synthesizer-based PRF in $TC^1 \subseteq NC^2$ *a la* [NR'95]
 - ★ Direct construction in $TC^0 \subseteq NC^1$ analogous to [NR'97,NRR'00]

- ★ Low-depth, highly efficient PRFs from (ring-)LWE
 - ★ Synthesizer-based PRF in $TC^1 \subseteq NC^2$ *a la* [NR'95]
 - ★ **Direct construction** in $TC^0 \subseteq NC^1$ analogous to [NR'97,NRR'00]
- ★ Main technique - “**derandomizing**” LWE: **deterministic errors**

- ★ Low-depth, highly efficient PRFs from (ring-)LWE
 - ★ Synthesizer-based PRF in $TC^1 \subseteq NC^2$ *a la* [NR'95]
 - ★ **Direct construction** in $TC^0 \subseteq NC^1$ analogous to [NR'97,NRR'00]
- ★ Main technique - “**derandomizing**” LWE: deterministic errors
Gives more practical PRGs to feed in GGM

- ★ Low-depth, highly efficient PRFs from (ring-)LWE
 - ★ Synthesizer-based PRF in $TC^1 \subseteq NC^2$ *a la* [NR'95]
 - ★ **Direct construction** in $TC^0 \subseteq NC^1$ analogous to [NR'97,NRR'00]
- ★ Main technique - “**derandomizing**” LWE: deterministic errors
Gives more practical PRGs to feed in GGM

Full version: <http://eprint.iacr.org/2011/401>

Outline

- 1 Introduction
- 2 Learning with Rounding
- 3 Direct Construction

- For n a power of 2 (say), define “cyclotomic” polynomial rings

$$R := \mathbb{Z}[x]/(x^n + 1) \quad \text{and} \quad R_q := R/qR = \mathbb{Z}_q[x]/(x^n + 1)$$

- For n a power of 2 (say), define “cyclotomic” polynomial rings

$$R := \mathbb{Z}[x]/(x^n + 1) \quad \text{and} \quad R_q := R/qR = \mathbb{Z}_q[x]/(x^n + 1)$$

- (R)LWE Problem:

$$\boxed{(a_i, a_i \cdot s + e_i)} \stackrel{c}{\approx} \boxed{(a_i, b_i) \in R_q \times R_q}$$

$s \in R_q$ **uniform** and **fixed**

e_i **Gaussian**

Hardness based on **worst case** (ideal-)lattice problems [LPR'10]

- For n a power of 2 (say), define “cyclotomic” polynomial rings

$$R := \mathbb{Z}[x]/(x^n + 1) \quad \text{and} \quad R_q := R/qR = \mathbb{Z}_q[x]/(x^n + 1)$$

- (R)LWE Problem:

$$\boxed{(a_i, a_i \cdot s + e_i)} \stackrel{c}{\approx} \boxed{(a_i, b_i) \in R_q \times R_q}$$

$s \in R_q$ **uniform** and fixed

e_i **Gaussian**

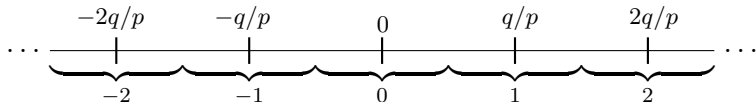
Hardness based on worst case (ideal-)lattice problems [LPR'10]

- Secret** errors e_i need fresh randomness. Can we make this deterministic?

“Learning with Rounding”: LWR [This Work]

- Generate errors deterministically by **rounding** to a “sparse” subgroup (Fundamental operation used in decryption algorithms)

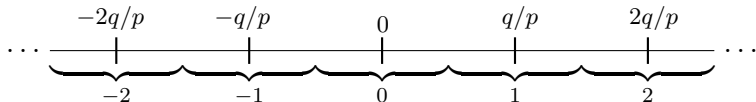
Let $p < q$ and define $\lfloor x \rfloor_p = \lfloor (p/q) \cdot x \rfloor \bmod p$ for $x \in \mathbb{Z}_q$



“Learning with Rounding”: LWR [This Work]

- Generate errors deterministically by rounding to a “sparse” subgroup (Fundamental operation used in decryption algorithms)

Let $p < q$ and define $\lfloor x \rfloor_p = \lfloor (p/q) \cdot x \rfloor \bmod p$ for $x \in \mathbb{Z}_q$



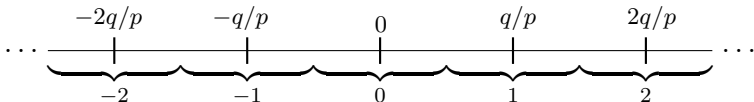
Ring-LWR Problem ($\text{RLWR}_{q,p}$)

Distinguish pairs $(a_i, \lfloor a_i \cdot s \rfloor_p) \in R_q \times R_p$ from **uniform**

“Learning with Rounding”: LWR [This Work]

- Generate errors deterministically by rounding to a “sparse” subgroup (Fundamental operation used in decryption algorithms)

Let $p < q$ and define $\lfloor x \rfloor_p = \lfloor (p/q) \cdot x \rfloor \bmod p$ for $x \in \mathbb{Z}_q$



Ring-LWR Problem ($RLWR_{q,p}$)

Distinguish pairs $(a_i, \lfloor a_i \cdot s \rfloor_p) \in R_q \times R_p$ from **uniform**

- LWE **conceals** low-order bits by adding small random noise
- LWR **discards** low-order bits instead

Theorem

Ring-LWE _{q, χ} \leq *RLWR* _{q, p} when $q \geq p \cdot n^{\omega(1)}$ and χ *short*

Theorem

Ring-LWE_{q,χ} ≤ RLWR_{q,p} when $q ≥ p · n^{ω(1)}$ and $χ$ short

$$\begin{aligned} (U(R_q), U(R_p)) &\equiv (U(R_q), \lfloor U(R_q) \rfloor_p) \\ &\stackrel{c}{\approx} (a, \lfloor a \cdot s + e \rfloor_p) \text{ (by the (R)LWE assumption)} \\ &\equiv (a, \lfloor a \cdot s \rfloor_p) \text{ (w.h.p., for short error)} \end{aligned}$$

Theorem

Ring-LWE_{q,χ} ≤ RLWR_{q,p} when $q ≥ p · n^{ω(1)}$ and $χ$ short

$$\begin{aligned} (U(R_q), U(R_p)) &\equiv (U(R_q), \lfloor U(R_q) \rfloor_p) \\ &\stackrel{c}{\approx} (a, \lfloor a \cdot s + e \rfloor_p) \text{ (by the (R)LWE assumption)} \\ &\equiv (a, \lfloor a \cdot s \rfloor_p) \text{ (w.h.p., for short error)} \end{aligned}$$

Synthesizers from LWR

- $S: R_q \times R_q \rightarrow R_p$ defined as $S(a, s) = \lfloor a \cdot s \rfloor_p$ is a synthesizer [NR'95]
- Gives a k -bit PRF through a $\log k$ depth tree of synthesizers
- Details of the construction in the paper

- 1 Introduction
- 2 Learning with Rounding
- 3 Direct Construction

Direct Construction: Shallower? More Efficient?

- Synth-PRF is $\log k$ levels of synthesizers (NC^2). Can we do better?

Direct Construction: Shallower? More Efficient?

- Synth-PRF is $\log k$ levels of synthesizers (NC^2). Can we do better?

Direct LWE-Based Construction

- Secret key is **uniform** $a \leftarrow R_q$ and **short** $s_1, \dots, s_k \in R$

Direct Construction: Shallower? More Efficient?

- Synth-PRF is $\log k$ levels of synthesizers (NC^2). Can we do better?

Direct LWE-Based Construction

- Secret key is **uniform** $a \leftarrow R_q$ and **short** $s_1, \dots, s_k \in R$
- “**Rounded subset-product**” function:

$$F_{a, s_1, \dots, s_k}(x_1 \cdots x_k) = \left[a \cdot \prod_{i=1}^k s_i^{x_i} \bmod q \right]_p$$

Direct Construction: Shallower? More Efficient?

- Synth-PRF is $\log k$ levels of synthesizers (NC^2). Can we do better?

Direct LWE-Based Construction

- Secret key is **uniform** $a \leftarrow R_q$ and **short** $s_1, \dots, s_k \in R$
- “Rounded subset-product” function:

$$F_{a,s_1,\dots,s_k}(x_1 \cdots x_k) = \left[a \cdot \prod_{i=1}^k s_i^{x_i} \bmod q \right]_p$$

Can be computed efficiently (in NC^1), via FFT/CRT and reduction to subset-sum

Direct Construction: Shallower? More Efficient?

- Synth-PRF is $\log k$ levels of synthesizers (NC^2). Can we do better?

Direct LWE-Based Construction

- Secret key is **uniform** $a \leftarrow R_q$ and **short** $s_1, \dots, s_k \in R$
- “Rounded subset-product” function:

$$F_{a,s_1,\dots,s_k}(x_1 \cdots x_k) = \left[a \cdot \prod_{i=1}^k s_i^{x_i} \bmod q \right]_p$$

Can be computed efficiently (in NC^1), via FFT/CRT and reduction to subset-sum

- [NR'97,NRR'00]: **direct** PRFs from DDH / factoring (in NC^1)

$$F_{g,s_1,\dots,s_k}(x_1 \cdots x_k) = g^{\prod s_i^{x_i}}$$

(Computing this needs a costly exponentiation or lots of preprocessing...)

Proof Outline

- Seed is **uniform** $a \in R_q$ and **short** $s_1, \dots, s_k \in R$

$$F_{a,s_1,\dots,s_k}(x_1 \cdots x_k) = \lfloor a \cdot s_1^{x_1} \cdots s_k^{x_k} \bmod q \rfloor_p$$

Proof Outline

- Seed is **uniform** $a \in R_q$ and **short** $s_1, \dots, s_k \in R$

$$F_{a,s_1,\dots,s_k}(x_1 \cdots x_k) = \lfloor a \cdot s_1^{x_1} \cdots s_k^{x_k} \bmod q \rfloor_p$$

- Like the **LWE** \leq **LWR** proof, but “souped up” to handle queries

Proof Outline

- Seed is **uniform** $a \in R_q$ and **short** $s_1, \dots, s_k \in R$

$$F_{a,s_1,\dots,s_k}(x_1 \cdots x_k) = \lfloor a \cdot s_1^{x_1} \cdots s_k^{x_k} \bmod q \rfloor_p$$

- Like the $\text{LWE} \leq \text{LWR}$ proof, but “souped up” to handle queries
Thought experiment: answer queries with

$$\begin{aligned} \tilde{F}(x) &:= \lfloor (a \cdot s_1^{x_1} + x_1 \cdot e_{x_1}) \cdot s_2^{x_2} \cdots s_k^{x_k} \rfloor_p \\ &= \left\lfloor a \prod_{i=1}^k s_i^{x_i} + x_1 \cdot e_{x_1} \cdot \prod_{i=2}^k s_i^{x_i} \right\rfloor_p \end{aligned}$$

W.h.p., $\tilde{F}(x) = F(x)$ due to “small” error and rounding

Proof Outline

- Seed is **uniform** $a \in R_q$ and **short** $s_1, \dots, s_k \in R$

$$F_{a,s_1,\dots,s_k}(x_1 \cdots x_k) = \lfloor a \cdot s_1^{x_1} \cdots s_k^{x_k} \bmod q \rfloor_p$$

- Like the $\text{LWE} \leq \text{LWR}$ proof, but “souped up” to handle queries
Thought experiment: answer queries with

$$\begin{aligned} \tilde{F}(x) &:= \lfloor (a \cdot s_1^{x_1} + x_1 \cdot e_{x_1}) \cdot s_2^{x_2} \cdots s_k^{x_k} \rfloor_p \\ &= \left\lfloor a \prod_{i=1}^k s_i^{x_i} + x_1 \cdot e_{x_1} \cdot \prod_{i=2}^k s_i^{x_i} \right\rfloor_p \end{aligned}$$

W.h.p., $\tilde{F}(x) = F(x)$ due to “small” error and rounding

- Replace $(a, a \cdot s_1 + e_{x_1})$ with (a_0, a_1) [ring-LWE]
 \Rightarrow New function $F'(x) = \lfloor a_{x_1} \cdot s_2^{x_2} \cdots s_k^{x_k} \rfloor_p$

Proof Outline

- Seed is **uniform** $a \in R_q$ and **short** $s_1, \dots, s_k \in R$

$$F_{a,s_1,\dots,s_k}(x_1 \cdots x_k) = \lfloor a \cdot s_1^{x_1} \cdots s_k^{x_k} \bmod q \rfloor_p$$

- Like the $\text{LWE} \leq \text{LWR}$ proof, but “souped up” to handle queries
Thought experiment: answer queries with

$$\begin{aligned}\tilde{F}(x) &:= \lfloor (a \cdot s_1^{x_1} + x_1 \cdot e_{x_1}) \cdot s_2^{x_2} \cdots s_k^{x_k} \rfloor_p \\ &= \left\lfloor a \prod_{i=1}^k s_i^{x_i} + x_1 \cdot e_{x_1} \cdot \prod_{i=2}^k s_i^{x_i} \right\rfloor_p\end{aligned}$$

W.h.p., $\tilde{F}(x) = F(x)$ due to “small” error and rounding

- Replace $(a, a \cdot s_1 + e_{x_1})$ with (a_0, a_1) [ring-LWE]
 \Rightarrow New function $F'(x) = \lfloor a_{x_1} \cdot s_2^{x_2} \cdots s_k^{x_k} \rfloor_p$
- Repeat inductively for s_2, s_3, \dots until we get the **Uniform** func

Summary

- 1 Derandomizing LWE: Generate errors deterministically
- 2 Efficient lattice-based PRFs from synthesizers and directly
- 3 [Zha'12]: these constructions also yield quantum PRFs

Conclusions

Summary

- 1 Derandomizing LWE: Generate errors deterministically
- 2 Efficient lattice-based PRFs from synthesizers and directly
- 3 [Zha'12]: these constructions also yield quantum PRFs

Open Questions

- 1 Get different proofs with better p/q ratios:

	LWR	Synth-PRF	Direct-PRF
Ratio (Current)	$n^{\omega(1)}$	$n^{\Theta(\log k)}$	$n^{\Theta(k)}$
Ratio (Hope)	\sqrt{n}	$\text{poly}(n)$	$\text{poly}(n)$

- 2 Efficient PRF from **parity with noise** (LPN) or **subset sum**?

```
int getRandomNumber()  
{  
    return 4; // chosen by fair dice roll.  
             // guaranteed to be random.  
}
```

(Image source: <http://xkcd.com/221/>)